CHALMERS
UNIVERSITY OF TECHNOLOGY

Master's Thesis

Iteratively Regularized Finite Element Method for Conductivity Reconstruction in a Waveguide

Carl Persson

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2016

Iteratively Regularized Finite Element Method for Conductivity Reconstruction in a Waveguide

Student/Author: Carl Persson

Supervisor/Examiner: Dr. Larisa Beilina

Department of Mathematical Sciences

Chalmers University of Technology

Gothenburg, Sweden 2016

Abstract

In this work we consider iteratively regularized conjugate gradient method to solve the inverse problem of reconstructing the unknown space dependent conductivity function in a bounded domain. The conductivity function is reconstructed by solving the hyperbolic differential equation using measured data in space and time on the boundary of the domain, where the measured data is generated by sending a plane-wave through the domain.

The inverse problem is stated as a minimization problem of the Lagrangian functional, defined using the Tikhonov functional, and approximate functions are computed by the finite element method (FEM). The stationary point that minimizes the Lagrangian is found by computing a sequence of conductivity functions that converges to the minimum point. Conjugate gradient method (CGM) and fixed point iterations (FPI) are considered to compute the minimization sequence, where the Tikhonov functional is iteratively updated.

The goal of this master's thesis is to compare the efficiency and speed of the optimization algorithms, when the regularization parameter in the Tikhonov functional is iteratively updated.

Contents

1 Introduction

Reconstruction techniques of the unknown conductivity function $a(x)$, $x \in \Omega \subset \mathbb{R}^3$, in space present increasing interest for the mathematical and physical research community. Some common techniques used today, are for example, ultrasound and x-ray, which give a good image of the object inside the material but a poor estimate of numerical values of $a(x)$. Possible areas of applications are cancer diagnostics, nondestructive testing of materials and many others.

Reconstruction of the conductivity is an ill-posed, inverse problem of finding a feasible $a(x)$ so that $A(a(x)) = b(x,t)$, where A is an arbitrary mapping operator to the system defined for all feasible $a(x)$, and $b(x,t) \in \Omega \times J$ is measured data in time interval $J \subset \mathbb{R}$ and space Ω. Since the inverse problem is ill-posed the solution might be numerically unstable for small perturbations of b, see [6, 11]. The inverse problem is solved by minimizing the Tikhonov functional:

$$F(a(x)) = \frac{1}{2}\|A(a(x)) - b\|^2_{L_2(\Omega\times J)} + \frac{\gamma}{2}\|a(x) - a_0(x)\|^2_{L_2(\Omega)},$$

where a_0 is an initial guess, assumed to be close to the exact solution, $\gamma \in (0,1)$ is the *regularization parameter*, $\|\cdot\|_{L_2(\Omega\times J)}$ and $\|\cdot\|_{L_2(\Omega)}$ are the L_2 norm over $\Omega \times J$ and Ω respectively. The regularization term in the functional ensures that solutions $\|a(x)\|_{L_2(\Omega)}$ remains bounded and close to the initial guess.

The conductivity $a(x)$ can be reconstructed using computer simulations of partial differential equations in two and three dimensions, meaning that the mapping operator $A(a(x))$ can be computed for any feasible $a(x)$, it is important to use efficient computational methods for speed and accuracy when the computational domain is very large.

This master's thesis examines the benefits of *iterative regularization*, using the Lagrangian approach to reconstruct the conductivity in a hyperbolic equation, see [1, 3] where this method is applied, which can also be used to reconstruct the coefficients in Maxwell's equations, see [5, 10]. This is done by comparing the results of computer simulations using different iterative minimization algorithms. The algorithms used in this work are fixed point iterations (FPI), conjugate gradient method (CGM) with a constant regularization parameter, and CGM with a iteratively regularized Tikhonov functional. See [5, 11] for details of iterative regularization.

In works [1, 3, 5, 10] the mapping $A(a(x))$ is computed with domain decomposition of $\Omega = \Omega_{FEM} \cup \Omega_{FDM} \in \mathbb{R}^3$, using an adaptive finite element method (FEM) on the subdomain Ω_{FEM} of the material where $a(x)$ is assumed to be unknown, and finite difference method (FDM) on Ω_{FDM} where $a(x)$ is assumed to be constant. Using FEM/FDM domain decomposition ensures that the more computationally costly FEM is only used where needed. The resulting equation systems are solved using parallel computing for speed.

However, in this work the inverse problem is solved in two dimensions using a non-adaptive FEM over the whole domain, without any parallel computing.

The mathematical equation that governs the propagation of a wave in a waveguide is the hyperbolic partial differential equation, which in its general form is defined as:

$$\frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (a\nabla u) = f \quad \text{in } \Omega \times (0,T), \tag{1}$$

where the scalar field $u(x,t)$ denotes the measured property of the wave such as voltage or displacement, $c(x) > 0$ denotes the wave speed in the domain, $a(x) > 0$ denotes the *conductivity* coefficient in the domain and $f(x,t)$ denotes some external force acting on the system. The domain $\Omega \in \mathbb{R}^3$ is assumed to be convex and bounded, the boundary to Ω is denoted Γ.

If c, a and f are known, then $u(x,t)$ can be solved using initial conditions:

$$u(x,0) = u_0(x) \quad \text{in } \Omega,$$

$$\frac{\partial u}{\partial t}(x,0) = u_1(x) \quad \text{in } \Omega,$$

and first order absorbing boundary conditions:

$$\frac{\partial u}{\partial n} = -\frac{\partial u}{\partial t} \quad \text{on } \Gamma \times (0,T).$$

The hyperbolic equation (1) is simplified for the scope of this master's thesis, which is comparison of iterative minimization methods for reconstructing the conductivity coefficient $a(x)$ in (1). It is sufficient to work in two dimensions with a rectangle domain $\Omega \in \mathbb{R}^2$, any results and conclusions are directly applicable on the problem in three dimensions. Working in two dimensions reduces computational time and makes it easier to view the conductivity $a(x)$. The coefficients in (1) are dimensionless, so that the wave speed $c = 1$ throughout the domain, and $a(x) \geq 1$ with $a(x) = 1$ on Γ. External forces $f = 0$ and initial conditions $u_0 = u_1 = 0$ are set to zero. The wave is initialized in Ω by introducing an impulse function $p(t)$ acting on a part of the boundary for a sub-interval $(0, t_1]$.

This simulates sending a *plane wave* through a material that is initially at rest with no external forces working on it, and then measuring $u(x, t)$ on the boundary over the time $t \in (0, T)$. The measurements of u on $\Gamma \times (0, T)$ gives information about the material and are used to reconstruct the conductivity inside the domain.

The data measured on the boundary where the plane wave is induced, is known as *backscatter data*, which is caused by that the wave is reflected in regions where $\nabla a \neq 0$. The data measured on the opposite boundary from where the plane wave is induced, is known as *transmitted data*, which is distorted from the induced plane wave if $\nabla a \neq 0$ in regions in the material.

The outline of this work is as follows. Section 2 of this thesis describes some general theory of ill-posed inverse problems and Tikhonov functionals that can be used to solve such problems, see [6, 11]. Section 3 describes some common iterative minimization methods when applied on real valued functions, see [4, 5]. Sections 4-10 define the classical problem described by P.D.E. with boundary and initial conditions, the variational formulation and minimization problem, who are all equivalent, that is:

$$\text{(P.D.E. Problem)} \Leftrightarrow \text{(Variational Formulation)} \Leftrightarrow \text{(Minimization Problem)},$$

where the classical P.D.E. problem have strong solutions, and variational formulations have weak solutions. The Lagrangian functional is defined in section 9, using the Tikhonov functional, and then it is shown that (Minimization Problem) $\Rightarrow$ (Variational Formulation) $\Rightarrow$ (P.D.E. Problem), in the sense that the strong solutions to the forward problem, adjoint problem and conductivity coefficient are derived from the variational formulations of the minimization problem. Then the finite element method is defined and described in detail in sections 11-13. The FEM gives approximate solutions to the variational formulation that minimizes the Lagrangian functional. Sections 14-15 define the minimization methods and algorithms used to compute a sequence $\{a_0(x),\ a_1(x),\ \ldots, a_N(x)\}$ that minimizes the Lagrangian functional. Sections 16-19 present the numerical results and discussions when algorithms of sections 13-14 are implemented in two different case studies.

The appendix presents and describes some common methods to define and compute linear basis functions in one and two dimensions, for the finite element method, and how to compute resulting FEM matrices. See [2, 7, 8, 9].

2 General theory about ill-posed inverse problems

In this section the general theory of real valued operators and functionals is presented. For further enquiry see [6, 7, 11].

2.1 Tikhonov functional

Let A be a mapping operator on a Hilbert space H_1 into a Hilbert space H_2 such that $A : H_1 \to H_2$. Also assume that the two Hilbert spaces H_1 and H_2 contain real valued elements, and let $(\cdot,\cdot)_{H_1}$, $(\cdot,\cdot)_{H_2}$ denote the *bilinear* scalar product associated with the two function spaces. Define the norms by:

$$\begin{aligned} \|x\|_{H_1}^2 &= (x,x)_{H_1} \quad x \in H_1, \\ \|y\|_{H_2}^2 &= (y,y)_{H_2} \quad y \in H_2. \end{aligned} \tag{2}$$

Assume that $A(x) = y$ is well-posed and can be computed for each $x \in H_1$, but $A(x)^{-1}$ is ill-posed. We want to find x^* such that $A(x^*) = y^*$, where y^* is known, let $X^* \neq \emptyset$ denote the set of possible solutions. Define the Tikhonov functional as:

$$F(x) = \frac{1}{2}\|A(x) - y^*\|_{H_2}^2 + \frac{\gamma}{2}\|x - x_0\|_{H_1}^2, \tag{3}$$

where $\gamma > 0$ is the regularization parameter and x_0 is the initial guess, a fixed point in H_1.

The initial guess x_0 is assumed to be chosen in the vicinity of $x^* \in X^*$, so that the chosen optimization method is able to converge to x^*, but the fixed point x_0 does not necessarily have to be the initial guess of the optimization method. It can be chosen as an arbitrary fixed point $\xi \in H_1$, that will consequently limit the set of possible minimization points to an open ball around ξ for any fixed $\gamma > 0$.

Our goal is to find the $x_\gamma \in H_1$ that minimizes F, that is:

$$F(x_\gamma) \leq F(\bar{x}) \quad \forall \bar{x} \in H_1. \tag{4}$$

The most direct way to find the minimum point is to compute the Fréchet derivative of the Tikhonov functional and find the stationary points. Denote the Fréchet derivative $F'(x) = H(x) + \gamma(x - a_0) \in H_1$ for simplicity, where the operator $H : H_1 \to H_1$, and find the stationary points x_γ that satisfies:

$$F'(x_\gamma) = H(x_\gamma) + \gamma(x_\gamma - a_0) = 0, \tag{5}$$

and chose the x_γ that is the global minimizer. Depending on γ, the point that minimizes F will differ from x^*. Clearly the Tikhonov functional satisfies:

$$\inf_{x \in H_1} F(x) \geq 0 \quad \forall \gamma \geq 0, \tag{6}$$

but the functional does not necessarily have a global minimizer. However, for any $\gamma > 0$ and for any accuracy $\varepsilon > 0$ there exists an element $x_\gamma^\varepsilon \in H_1$ such that:

$$\inf_{x \in H_1} F(x) \leq F(x_\gamma^\varepsilon) \leq \inf_{x \in H_1} F(x) + \varepsilon. \tag{7}$$

Meaning that $F(x_\gamma^\varepsilon)$ differs by not more than ε from the infimum. Take some minimum point $x^* \in X^*$, we have the estimate:

$$F(x_\gamma^\varepsilon) \leq F(x^*) + \varepsilon. \tag{8}$$

Using the definition of the Tikhonov functional, the estimate is:

$$\frac{1}{2}\|A(x_\gamma^\varepsilon) - y^*\|_{H_2}^2 + \frac{\gamma}{2}\|x_\gamma^\varepsilon - x_0\|_{H_1}^2 \leq \frac{\gamma}{2}\|x^* - x_0\|_{H_1}^2 + \varepsilon. \tag{9}$$

From equation (9) it follows that:

$$\begin{aligned} \frac{1}{2}\|A(x_\gamma^\varepsilon) - y^*\|_{H_2}^2 &\leq \frac{\gamma}{2}\|x^* - x_0\|_{H_1}^2 + \varepsilon, \\ \|x_\gamma^\varepsilon - x_0\|_{H_1}^2 &\leq \|x^* - x_0\|_{H_1}^2 + \frac{2\,\varepsilon}{\gamma}. \end{aligned} \tag{10}$$

Suppose that $\varepsilon = \varepsilon(\gamma)$ depends on the regularization parameter such that:

$$\lim_{\gamma \to 0} \frac{\varepsilon(\gamma)}{\gamma} = 0, \tag{11}$$

which implies that $\lim_{\gamma \to 0} \varepsilon(\gamma) = 0$. Then the sequence of elements $\{x_\gamma^{\varepsilon(\gamma)}\}_{\gamma \in (0,\gamma_0]}$ fulfilling the estimate, minimizes the functional as $\gamma \to 0$, since:

$$\begin{aligned} &\lim_{\gamma \to 0} \|A(x_\gamma^\varepsilon) - y^*\|_{H_2} = 0, \\ &\lim_{\gamma \to 0} \|x_\gamma^\varepsilon - x_0\|_{H_1} = \|x^* - x_0\|_{H_1}. \end{aligned} \tag{12}$$

2.2 Iteratively regularized Tikhonov functional

The general idea of iteratively regularized Tikhonov functional is to choose an iterative optimization method, such as the gradient method (GM) or conjugate gradient method (CGM), to compute a sequence of points $\{x_n\}$ that converges to x^* for an arbitrary strongly convex functional, given a sequence of regularization parameters $\{\gamma_n\}$ that converges to zero.

This implies that the Tikhonov functional tends to zero for large n:s, instead of some positive constant. The sequence $\{\gamma_n\}$ should fulfill:

$$\begin{aligned} &\gamma_0 \geq \gamma_1 \geq \dots \gamma_n \geq \gamma_{n+1} > 0, \\ &\lim_{n \to \infty} \gamma_n = 0, \end{aligned} \tag{13}$$

and be chosen to ensure fast convergence for the optimization method.

Let $\varepsilon_n = \varepsilon(\gamma_n)$ denote the accuracy and $x_{\gamma_n}^{\varepsilon_n}$ denote the point that differs by no more than ε_n from the infimum, dependent on γ_n and iteration n in the iterative scheme. Write the estimates derived from equation (9) as:

$$\begin{aligned} \frac{1}{2}\|A(x_{\gamma_n}^{\varepsilon_n}) - y^*\|_{H_2}^2 &\leq \frac{\gamma_n}{2}\|x^* - x_0\|_{H_1}^2 + \varepsilon_n, \\ \|x_{\gamma_n}^{\varepsilon_n} - x_0\|_{H_1}^2 &\leq \|x^* - x_0\|_{H_1}^2 + \frac{2\,\varepsilon_n}{\gamma_n}, \end{aligned} \tag{14}$$

and suppose that:

$$\lim_{n \to \infty} \varepsilon_n = \lim_{n \to \infty} \gamma_n = \lim_{n \to \infty} \frac{\varepsilon_n}{\gamma_n} = 0. \tag{15}$$

Let x_{γ_n} denote the stationary point that exactly minimizes the Tikhonov functional in each given iteration, with the condition $F'(x_{\gamma_n}) = 0$.

Consider using the gradient method with an iteratively regularized parameter γ_n that satisfies condition (13), to find the global minimizer. The gradient method is based on computing the steepest descent $-F'(x_n) \in H_1$ at point x_n and minimize the function $F(x_n - \alpha F'(x_n))$ by performing a line search along $x_n - \alpha F'(x_n)$. The step-size $\alpha_n > 0$ that minimizes the function along the steepest descent is then chosen to compute x_{n+1}, such that:

$$x_{n+1} = x_n - \alpha_n F'(x_n) = x_n - \alpha_n(H(x_n) + \gamma_n(x_n - x_0)), \tag{16}$$

until convergence is achieved for the sequence of $\{x_n\}$. How to perform the line search and compute $F'(x_n)$ is explained for the model problem in sections 10, 14. The gradient method implemented on general real valued functions as $f : \mathbb{R}^d \to \mathbb{R}$ is described in more detail in section 3. For now, we assume that it is possible to compute $F'(x_n)$ and α_n for the arbitrary Tikhonov functional. The step-size $\alpha_n > 0$ is an inner parameter of the gradient method that will depend on the regularization parameter γ_n.

To find a suitable sequence of $\{\gamma_n\}$ that fits the gradient method, start by subtracting the global minimizer x_{γ_n} and take the H_1 norm squared, so that:

$$\|x_{n+1} - x_{\gamma_n}\|_{H_1}^2 = \|x_n - x_{\gamma_n} - \alpha_n(F'(x_n) - F'(x_{\gamma_n}))\|_{H_1}^2. \tag{17}$$

Here we have used that $F'(x_{\gamma_n}) = 0$ on the right hand side. Assume that the H_1 norm is defined by a *bilinear* scalar product so that $\|\cdot\|_{H_1}^2 = (\cdot,\cdot)_{H_1}$, we then get:

$$\begin{aligned}\|x_{n+1} - x_{\gamma_n}\|_{H_1}^2 = \|x_n - x_{\gamma_n}\|_{H_1}^2 - 2\alpha_n(F'(x_n) - F'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} \\ + \alpha_n^2\|F'(x_n) - F'(x_{\gamma_n})\|_{H_1}^2.\end{aligned} \tag{18}$$

The sequence of regularization parameters is found by estimating equation (18) and analysing its convergence rate.

The first step is to estimate the $(F'(x_n) - F'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1}$ term. Use that $F'(x_n) = H(x_n) + \gamma_n(x_n - x_0) = \xi$, where ξ is an arbitrary element in H_1, and that $F'(x_{\gamma_n}) = H(x_{\gamma_n}) + \gamma_n(x_{\gamma_n} - x_0) = 0$. We get that $F'(x_n) - F'(x_{\gamma_n}) = H(x_n) - H(x_{\gamma_n}) + \gamma_n(x_n - x_{\gamma_n}) = \xi$. Scalar multiply $F'(x_n) - F'(x_{\gamma_n})$ by $x_n - x_{\gamma_n}$ to get:

$$\begin{aligned}(F'(x_n) - F'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} = (H'(x_n) - H'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} \\ + \gamma_n\|x_n - x_{\gamma_n}\|_{H_1}^2 = (\xi, x_n - x_{\gamma_n})_{H_1}.\end{aligned} \tag{19}$$

Use the fact that F is strongly convex in H_1, so that:

$$(H'(x_2) - H'(x_1), x_2 - x_1)_{H_1} \geq 0 \quad \forall x_1, x_2 \in H_1. \tag{20}$$

Hence we get the inequality:

$$(F'(x_n) - F'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} \geq \gamma_n\|x_n - x_{\gamma_n}\|_{H_1}^2, \tag{21}$$

which means that:

$$-(F'(x_n) - F'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} \leq -\gamma_n\|x_n - x_{\gamma_n}\|_{H_1}^2. \tag{22}$$

Substitute inequality (22) into equation (18), to get:

$$\begin{aligned}\|x_{n+1} - x_{\gamma_n}\|_{H_1}^2 \leq \|x_n - x_{\gamma_n}\|_{H_1}^2 - 2\alpha_n\gamma_n\|x_n - x_{\gamma_n}\|_{H_1}^2 \\ + \alpha_n^2\|F'(x_n) - F'(x_{\gamma_n})\|_{H_1}^2.\end{aligned} \tag{23}$$

The next step is to estimate the $\|F'(x_n) - F'(x_{\gamma_n})\|_{H_1}^2$ term. This is done by using the definitions of $F'(x_n)$, $F'(x_{\gamma_n})$ and expanding the bilinear scalar product $\|\cdot\|_{H_1}^2 = (\cdot,\cdot)_{H_1}$, so that:

$$\begin{aligned}\|F'(x_n) - F'(x_{\gamma_n})\|_{H_1}^2 = \cdots &= \|H'(x_n) - H'(x_{\gamma_n})\|_{H_1}^2 \\ &+ 2\gamma_n(H'(x_n) - H'(x_{\gamma_n}), x_n - x_{\gamma_n})_{H_1} \\ &+ \gamma_n^2\|x_n - x_{\gamma_n}\|_{H_1}^2 \\ &\leq \|H'(x_n) - H'(x_{\gamma_n})\|_{H_1}^2 + \gamma_n^2\|x_n - x_{\gamma_n}\|_{H_1}^2 \\ &\leq \gamma_n^2\|x_n - x_{\gamma_n}\|_{H_1}^2,\end{aligned} \tag{24}$$

where inequality (20) is used again. Substitute (24) into (23), divide both sides by $\|x_n - x_{\gamma_n}\|_{H_1}^2$ and take the square root of the expression, to get:

$$\frac{\|x_{n+1} - x_{\gamma_n}\|_{H_1}}{\|x_n - x_{\gamma_n}\|_{H_1}} \leq \sqrt{1 - 2\alpha_n\gamma_n + \alpha_n^2\gamma_n^2} = 1 - \alpha_n\gamma_n. \tag{25}$$

To get convergence in the gradient method, we need to choose γ_n, α_n so that $0 < 1 - \alpha_n\gamma_n < 1$ is small for at least large n:s. The sequence of $\{\gamma_n\}$ should fulfill (13) and the sequence of step-sizes $\{\alpha_k\}$ should converge to zero as the gradient method converges to the minimum point.

One example of such a sequence that slowly converges to zero is:

$$\begin{aligned} \gamma_n &= \frac{\gamma_0}{(n+1)^p}, \\ \alpha_n &= \frac{\alpha_0}{(n+1)^{p+q}}, \\ n &= 0, 1, \ldots, \end{aligned} \tag{26}$$

where $\gamma_0, \alpha_0, p \in (0,1)$ and $q > 0$. For further enquiry on the theory behind choosing a suitable sequence, see [5, 11].

3 Minimization methods, some common iterative schemes

The iterative schemes described in this section are for real valued functions $f(x) : \mathbb{R}^n \to \mathbb{R}$, continuous for $x \in \Omega$, and $f(x^*) \leq f(x)\ \forall x \in \Omega$ is the sought for minimum point.

3.1 Definition of convergence rate

For any given iterative scheme, the sequence $x_k \to x^*$ as $k \to \infty$, converges with rate $p \geq 1$, if $\exists c > 0$, such that:

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} = c. \tag{27}$$

In the computer simulations, x^* is known, and the error sequence $e_k = x_k - x^*$ can be computed and estimated by $|e_{k+1}| \approx c|e_k|^p$ for large enough k. If the exact x^* is not known, the the error can be estimated by $\tilde{e}_k = x_k - x_{k-1}$ for large enough k [4]. That is to say, c and p can be approximated by solving:

$$\varepsilon_k = |e_k| = |x_k - x^*|, \tag{28a}$$

$$\tilde{\varepsilon}_k = |\tilde{e}_k| = |x_k - x_{k-1}|, \tag{28b}$$

$$\frac{\varepsilon_{k+1}}{(\varepsilon_k)^p} \approx c, \qquad \text{(when } k \text{ is large)} \tag{28c}$$

$$\frac{\varepsilon_{k+1}}{\varepsilon_k} \approx \left(\frac{\varepsilon_k}{\varepsilon_{k-1}}\right)^p, \qquad \text{(when } k \text{ is large)} \tag{28d}$$

for large enough k:s, where you use the error e_k or $\tilde{e}_k$, depending on if x^* is known or not.

3.2 Fixed point iteration (FPI)

The point x^* is a fixed point to f if it fulfills $f(x^*) = x^*$. The sequence:

$$x_{k+1} = f(x_k) \tag{29}$$

converges to x^* for x_0 chosen in the vicinity of x^*. If the function is Lipschitz continuous, so that $|f(x_i) - f(x_j)| \leq L|x_i - x_j|$ for any points $x_i, x_j \in \Omega$, with Lipschitz constant L<1, it can be shown that $|x_k - x_{k-1}| \leq L^{k-1}|x_1 - x_0|$. Fixed point iteration converges linearly to x^*, since:

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|} = L. \tag{30}$$

3.3 Gradient method (GM)

Assume that $f(x) : \mathbb{R}^n \to \mathbb{R}$ is continuous, and that $f'(x) : \mathbb{R}^n \to \mathbb{R}^n$ exists. Then minimize $f(x)$ by computing the sequence:

$$\begin{aligned} d_k &= -f'(x_k), \\ \alpha_k &= \operatorname{argmin}\{f(x_k + \alpha d_k) : \alpha \geq 0\}, \\ x_{k+1} &= x_k + \alpha_k d_k. \end{aligned} \tag{31}$$

The step-size α_k is chosen to minimize the function $g_k(\alpha) = f(x_k + \alpha d_k)$ for each k. This can be seen as doing a line-search from the point x_k along the direction d_k, which may not be possible or feasible to do exactly, then choose a_k to "loosely" minimize the functional along the direction.

3.3.1 GM on quadratic functions

A common minimization problem is to minimize the quadratic function:

$$f(x) = \frac{1}{2}x^T Ax - x^T b, \tag{32}$$

where A is an $n \times n$ symmetric positive definite matrix. We have that $f'(x) = Ax - b$ and $f''(x) = A$. Since $f'(x) = Ax - b$ we get that $d_k = b - Ax_k$ and $g_k(\alpha) = f(x_k + \alpha d_k)$ is a second order polynomial that can be computed and minimized for each k. By expanding $g_k(\alpha)$ and computing $g'(\alpha) = 0$, we get that:

$$\begin{aligned} g_k(\alpha) &= \frac{1}{2}\Big(x_k^T Ax_k + 2\alpha d_k^T Ax_k + \alpha^2 d_k^T Ad_k\Big), \\ g'(\alpha) &= d_k^T Ax_k + \alpha d_k^T Ad_k - d_k^T b = 0, \\ \alpha &= \frac{d_k^T(b - Ax_k)}{d_k^T Ad_k} = \frac{d_k^T d_k}{d_k^T Ad_k} = \frac{\|f'(x_k)\|_2^2}{\|f'(x_k)\|_A^2}. \end{aligned} \tag{33}$$

The computing sequence for $f(x) = \frac{1}{2}x^T Ax - x^T b$ then becomes:

$$\begin{aligned} d_k &= -(Ax_k - b), \\ \alpha_k &= \frac{d_k^T d_k}{d_k^T Ad_k}, \\ x_{k+1} &= x_k + \alpha_k d_k. \end{aligned} \tag{34}$$

3.4 Conjugate gradient method (CGM)

Let $f(x)$ be as for GM. Minimize $f(x)$ by the sequence:

$$\begin{aligned} d_0 &= -f'(x_0), && \text{(first iteration)} \\ d_k &= -f'(x_k) + \beta_k d_{k-1}, \\ \beta_k &= \frac{\|f'(x_k)\|^2}{\|f'(x_{k-1})\|^2}, \\ \alpha_k &= \operatorname{argmin}\{f(x_k + \alpha d_k) : \alpha \geq 0\}, \\ x_{k+1} &= x_k + \alpha_k d_k. \end{aligned} \tag{35}$$

The major difference from the gradient method, is that a fraction of the previous direction d_{k-1} is added to d_k. As for the GM, α_k is chosen to minimize the function $g_k(\alpha) = f(x_k + \alpha d_k)$ for each k. It should also be mentioned that β_k can be chosen in many ways, the way described above is the Fletcher-Reeves method.

3.4.1 CGM on quadratic functions

If $f(x) = \frac{1}{2}x^T Ax - x^T b$, where A is S. P. D., the computing sequence becomes:

$$\begin{aligned} g_k &= f'(x_k) = Ax_k - b, \\ d_0 &= -g_0, \qquad\qquad \text{(first iteration)} \\ \beta_k &= \frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}, \\ d_k &= -g_k + \beta_k d_{k-1}, \\ \alpha_k &= \frac{g_k^T g_k}{d_k^T A d_k}, \\ x_{k+1} &= x_k + \alpha_k d_k. \end{aligned} \tag{36}$$

Here it is used that $f'(x_k) \perp f'(x_{k-1})$ when computing α_k. As before we get that:

$$\alpha = \frac{d_k^T(b - Ax_k)}{d_k^T A d_k} = \frac{(r_k + \beta_k d_{k-1})^T r_k}{d_k^T A d_k} = \frac{r_k^T r_k}{d_k^T A d_k}. \tag{37}$$

3.5 Example of GM and CGM on a quadratic function

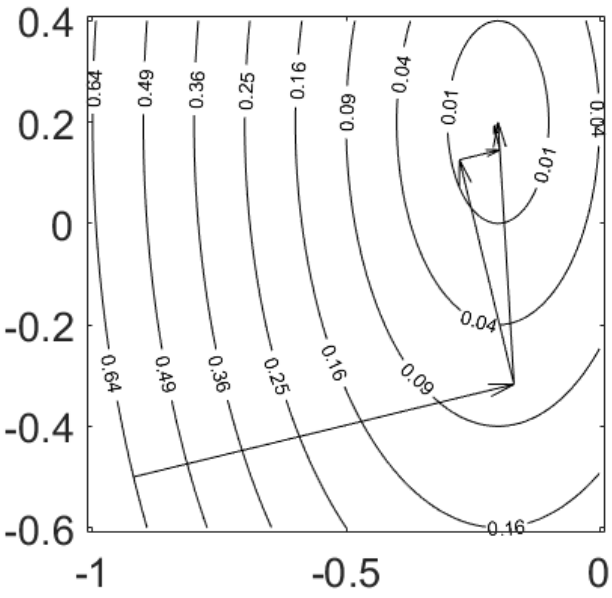


Figure 1: Example of GM and CGM on a quadratic positive definite function.

Figure 1 shows an example of GM and CGM on the quadratic function $f(x) = (x_1 + 0.2)^2 + (0.5(x_2 - 0.2))^2$. The contour of $f(x)$ is outlined and the arrows show how the GM and CGM approaches the minimum point $(-0.2, 0.2)$ from the same starting point.

CGM needs at most n iterations to find the exact minimum point for a quadratic S.P.D. function in n dimensions, given any starting point.

Depending on the starting point and if the quadratic function have elliptic contours, the GM may never reach the exact minimum point in a finite number of steps.

4 Model problem

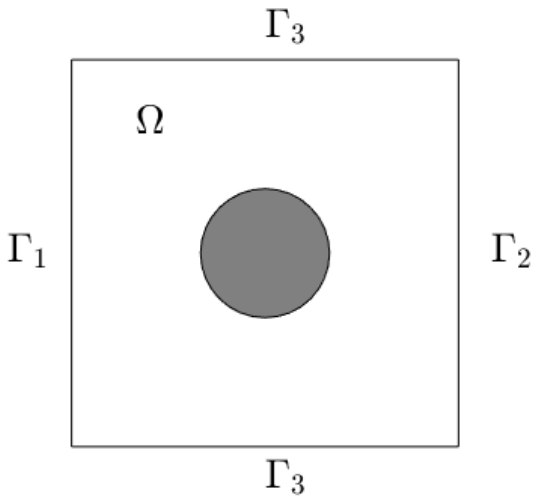


Figure 2: Example of the rectangular domain Ω in $\mathbb{R}^2$. The grey circle shows where $a(x) > 1$. Boundaries where $n \cdot \nabla u = 0$ are unified into Γ_3.

The model problem used to reconstruct the conductivity is the hyperbolic equation (1) in $d = 2, 3$ dimensions with absorbing boundary conditions and zero initial conditions. Let $\Omega \in \mathbb{R}^d$ be a bounded domain where the conductivity $a(x)$ is a function of x. Ω is assumed to be a rectangle domain in two dimensions, and a rectangular box in three dimensions. The boundaries to Ω are denoted by $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$, where Γ_1 and Γ_2 are opposite. The boundaries where $n \cdot \nabla u = 0$ are unified into Γ_3, see figure 2 for an example of Ω in two dimensions. The model problem used in computations is defined as:

$$\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (a \nabla u) = 0 \qquad \text{in } \Omega \times (0, T), \tag{38a}$$

$$u(x, 0) = 0 \qquad \text{in } \Omega, \tag{38b}$$

$$\frac{\partial u}{\partial t}(x, 0) = 0 \qquad \text{in } \Omega, \tag{38c}$$

$$\frac{\partial u}{\partial n} = p(t) \qquad \text{on } \Gamma_1 \times (0, t_1], \tag{38d}$$

$$\frac{\partial u}{\partial n} = -\frac{\partial u}{\partial t} \qquad \text{on } \Gamma_1 \times (t_1, T), \tag{38e}$$

$$\frac{\partial u}{\partial n} = -\frac{\partial u}{\partial t} \qquad \text{on } \Gamma_2 \times (0, T), \tag{38f}$$

$$\frac{\partial u}{\partial n} = 0 \qquad \text{on } \Gamma_3 \times (0, T). \tag{38g}$$

The conductivity coefficient $a(x)$ is assumed to satisfy:

$$\begin{aligned} a(x) &\in [1, a_{max}] \quad x \in \Omega, \\ a(x) &= 1 \qquad\qquad x \in \Gamma, \end{aligned} \tag{39}$$

for some constant $1 < a_{max} \leq d_1$, where d_1 is a bound we know a priori.

5 Forward problem

Determine $u(x,t)$ in (38), when $a(x)$ and $p(t)$ are known.

5.1 Definition of $p(t)$

The impulse $p(t)$ induces a plane wave propagating from Γ_1 to Γ_2 in the model problem. It is chosen as an sine wave.

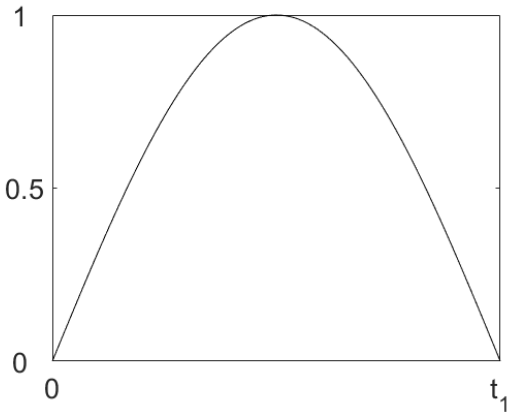


Figure 3: $p(t)$ plotted for $t \in [0, t_1]$.

Definition of $p(t)$:

$$\begin{aligned} p(t) &= \sin(\omega t) \quad \forall t \in [0, t_1], \\ t_1 &= \frac{\pi}{\omega}, \end{aligned} \tag{40}$$

for some chosen angular frequency ω [rad/s], frequency f [1/s], speed $c = 1$ [m/s], period time t_p [s] and wavelength l_p [m]:

$$\begin{aligned} \omega &= 2\pi f, \\ t_p &= \frac{1}{f}, \\ c &= f l_p \Leftrightarrow l_p = \frac{1}{f}. \end{aligned} \tag{41}$$

6 Inverse problem

Determine the unknown coefficient $a(x)$ in (38), that satisfies (39), given known measurements $\tilde{u}(x,t)$ on the boundary:

$$\begin{aligned} u(x,t) &= \tilde{u}(x,t) \quad \forall (x,t) \in \Gamma_1 \times (0,T) \ \text{(Backscatter data)}, \\ u(x,t) &= \tilde{u}(x,t) \quad \forall (x,t) \in \Gamma_2 \times (0,T) \ \text{(Transmitted data)}. \end{aligned} \tag{42}$$

It is enough to have one set of known boundary data to reconstruct the conductivity, but the reconstruction becomes better given both measurements. In the derivation of Tikhonov and Lagrangian functionals it is assumed that only backscatter data is known.

In the FEM computer simulations the data is generated by defining a completely known $a^*(x)$ that is used to compute $u^*(x,t)$. Noise is then added to the data by:

$$\begin{aligned} \tilde{u}(x,t) &= u^*(x,t) + \sigma \cdot P \cdot u^*(x,t) \quad \forall (x,t) \in \Gamma_1 \times (0,T) \ \text{(Backscatter data)}, \\ \tilde{u}(x,t) &= u^*(x,t) + \sigma \cdot P \cdot u^*(x,t) \quad \forall (x,t) \in \Gamma_2 \times (0,T) \ \text{(Transmitted data)}, \\ \tilde{u}(x,t) &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{else}, \end{aligned} \tag{43}$$

where P is some probability distribution and $\sigma \in [0,1]$ is the noise level. As an example, P can be the standard normal distribution or a random uniform distribution with values from $[-1,1]$. The random variable P is drawn for each (x,t) where data is collected.

7 Function spaces

Let $\Omega_T = \Omega \times (0,T)$ and define the real valued function spaces:

$$\begin{aligned} U^1 &= \Big\{ w \in H^1(\Omega_T) : w(x,0) = \frac{\partial w}{\partial t}(x,0) = 0 \Big\}, \\ V^1 &= \Big\{ w \in H^1(\Omega_T) : w(x,T) = \frac{\partial w}{\partial t}(x,T) = 0 \Big\}, \\ A &= \Big\{ w \in L_2(\Omega) : 1 \leq w \leq a_{max}, w = 1 \text{ on } \Gamma \Big\}. \end{aligned} \tag{44}$$

8 Variational formulation of forward problem

In this section the variational formulation is derived to the forward problem of finding $u \in U^1$, using the standard Galerkin method, assuming that $a(x) \in A$ is known. First step is to multiply the differential equation (38a) by the function $v(x,t) \in V^1$ that has the end-condition $v(x,T) = v_t(x,T) = 0$, then integrate over $\Omega \times (0,T)$:

$$\int_\Omega \int_0^T \frac{\partial^2 u}{\partial t^2} v \,\mathrm{d}x\mathrm{d}t - \int_\Omega \int_0^T \nabla \cdot (a \nabla u) v \,\mathrm{d}x\mathrm{d}t = 0.$$

Use partial integration on the time-derivative term and that $\nabla \cdot (va\nabla u) = \nabla v \cdot (a \nabla u) + v \nabla \cdot (a \nabla u)$, to get:

$$\begin{aligned} &\int_\Omega \left(\Big[\frac{\partial u}{\partial t} v \Big]_{t=0}^T - \int_0^T \frac{\partial u}{\partial t} \frac{\partial v}{\partial t} \,\mathrm{d}t \right) \mathrm{d}x + \int_\Omega \int_0^T a \,\nabla u \cdot \nabla v \,\mathrm{d}x\mathrm{d}t \\ &- \int_\Omega \int_0^T \nabla \cdot (av \nabla u) \,\mathrm{d}x\mathrm{d}t = 0. \end{aligned}$$

The first term is zero, since $v(x,T) = 0$ and $u_t(x,0) = 0$. Use Gauss divergence theorem and that $a = 1$ on Γ, to get:

$$\begin{aligned} &- \int_\Omega \int_0^T \frac{\partial u}{\partial t} \frac{\partial v}{\partial t} \,\mathrm{d}x\mathrm{d}t + \int_\Omega \int_0^T a \,\nabla u \cdot \nabla v \,\mathrm{d}x\mathrm{d}t \\ &- \int_\Gamma \int_0^T \frac{\partial u}{\partial n} v \,\mathrm{d}s\mathrm{d}t = 0, \end{aligned}$$

where $\frac{\partial u}{\partial n} = n \cdot \nabla u$. Insert the boundary conditions for u in equation (38d)-(38g) to get:

$$\begin{aligned} &- \int_\Omega \int_0^T \frac{\partial u}{\partial t} \frac{\partial v}{\partial t} \,\mathrm{d}x\mathrm{d}t + \int_\Omega \int_0^T a \,\nabla u \cdot \nabla v \,\mathrm{d}x\mathrm{d}t \\ &+ \int_{\Gamma_1} \int_{t_1}^T \frac{\partial u}{\partial t} v \,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2} \int_0^T \frac{\partial u}{\partial t} v \,\mathrm{d}s\mathrm{d}t = \int_{\Gamma_1} \int_0^{t_1} p(t) v \,\mathrm{d}s\mathrm{d}t. \end{aligned} \tag{45}$$

The variational formulation of the forward problem is then to find $u \in U^1$ such that equation (45) is fullfilled for all $v \in V^1$.

9 Tikhonov and Lagrangian functionals

Define the Tikhonov functional $F(a) = F(u(a), a)$ as:

$$F(u,a) = \frac{1}{2}\int_{\Gamma_1}\int_0^T (u-\tilde{u})^2 \,\mathrm{d}s\mathrm{d}t + \frac{1}{2}\gamma\int_\Omega (a-a_0)^2\,\mathrm{d}x, \tag{46}$$

where $\tilde{u}$ is the known measurements on the boundary, $\gamma \in (0,1)$ is the *Tikhonov regularization parameter* and a_0 is the initial guess of the conductivity.

Alternatively, if there is backscatter *and* transmitted data available, the integral over Γ_2 is be added for additional input, and the Tikhonov functional can be defined as:

$$\begin{aligned} F(u,a) =& \frac{1}{2}\int_{\Gamma_1}\int_0^T (u-\tilde{u})^2 \,\mathrm{d}s\mathrm{d}t + \frac{1}{2}\int_{\Gamma_2}\int_0^T (u-\tilde{u})^2 \,\mathrm{d}s\mathrm{d}t \\ &+\frac{1}{2}\gamma\int_\Omega (a-a_0)^2\,\mathrm{d}x. \end{aligned} \tag{47}$$

Define the Lagrangian $L(a) = L(u(a), v(a), a)$ as:

$$\begin{aligned} L(u,v,a) =& F(u,a) - \int_\Omega\int_0^T \frac{\partial u}{\partial t}\frac{\partial v}{\partial t}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla u\cdot\nabla v\,\mathrm{d}x\mathrm{d}t \\ &- \int_{\Gamma_1}\int_0^{t_1} p(t)v\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_1}\int_{t_1}^T \frac{\partial u}{\partial t}v\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2}\int_0^T \frac{\partial u}{\partial t}v\,\mathrm{d}s\mathrm{d}t, \end{aligned} \tag{48}$$

where the functions $(u,v,a) \in U^1 \times V^1 \times A$ from equation (44). The inverse problem is then formulated as an optimization problem by minimizing the Lagrangian with respect to (u,v,a).

10 Minimization problem

The minimization problem is to find $(u,v,a) \in U^1 \times V^1 \times A$ such that

$$L(u,v,a) \le L(\bar{u},\bar{v},\bar{a}) \quad \forall (\bar{u},\bar{v},\bar{a}) \in U^1 \times V^1 \times A, \tag{49}$$

where L is the Lagrangian functional defined in equation (48). This is done by searching for a stationary point (u,v,a) that has the Fréchet derivatives $\frac{\partial L}{\partial u}(u,v,a)(\bar{u}) = 0$, $\frac{\partial L}{\partial v}(u,v,a)(\bar{v}) = 0$ and $\frac{\partial L}{\partial a}(u,v,a)(\bar{a}) = 0$.

10.1 Differentiating $L(u,v,a)$ with respect to v

Let (u,v,a) be the stationary point that minimizes the Lagrangian functional and compute:

$$\begin{aligned} \frac{\mathrm{d}L}{\mathrm{d}\alpha}(u, v+\alpha\bar{v}, a) =& -\int_\Omega\int_0^T \frac{\partial u}{\partial t}\frac{\partial \bar{v}}{\partial t}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla u\cdot\nabla\bar{v}\,\mathrm{d}x\mathrm{d}t \\ &- \int_{\Gamma_1}\int_0^{t_1} p(t)\bar{v}\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_1}\int_{t_1}^T \frac{\partial u}{\partial t}\bar{v}\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2}\int_0^T \frac{\partial u}{\partial t}\bar{v}\,\mathrm{d}s\mathrm{d}t, \end{aligned}$$

where α is some scalar and $\bar{v}$ is any function in V^1. Let $\alpha \to 0$ to get $\frac{\partial L}{\partial v}(u,v,a)(\bar{v}) = 0$, which is:

$$\begin{aligned} &-\int_\Omega\int_0^T \frac{\partial u}{\partial t}\frac{\partial \bar{v}}{\partial t}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla u\cdot\nabla\bar{v}\,\mathrm{d}x\mathrm{d}t \\ &-\int_{\Gamma_1}\int_0^{t_1} p(t)\bar{v}\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_1}\int_{t_1}^T \frac{\partial u}{\partial t}\bar{v}\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2}\int_0^T \frac{\partial u}{\partial t}\bar{v}\,\mathrm{d}s\mathrm{d}t = 0, \end{aligned} \tag{50}$$

where $\bar{v}$ is any function in V^1. Equation (50) gives the *variational formulation* for the *forward problem*, which is defined as: find $u(x,t)$ that satisfies equation (50) $\forall \bar{v} \in V^1$. This is exactly the same as derived in equation (45). The strong solution to $u(x,t)$ fulfills equation (38).

10.2 Differentiating $L(u,v,a)$ with respect to u

Let (u,v,a) be the stationary point that minimizes the Lagrangian functional and compute:

$$\frac{\mathrm{d}L}{\mathrm{d}\alpha}(u+\alpha\bar{u},v,a) = \int_{\Gamma_1}\int_0^T (u+\alpha\bar{u}-\tilde{u})\bar{u}\,\mathrm{d}s\mathrm{d}t - \int_\Omega\int_0^T \frac{\partial\bar{u}}{\partial t}\frac{\partial v}{\partial t}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla\bar{u}\cdot\nabla v\,\mathrm{d}x\mathrm{d}t$$
$$+\int_{\Gamma_1}\int_{t_1}^T \frac{\partial\bar{u}}{\partial t}v\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2}\int_0^T \frac{\partial\bar{u}}{\partial t}v\,\mathrm{d}s\mathrm{d}t = 0,$$

where α is some scalar and $\bar{u}$ is any function in U^1. Let $\alpha \to 0$ to get $\frac{\partial L}{\partial u}(u,v,a)(\bar{u}) = 0$, which is:

$$\begin{aligned}&\int_{\Gamma_1}\int_0^T (u-\tilde{u})\bar{u}\,\mathrm{d}s\mathrm{d}t - \int_\Omega\int_0^T \frac{\partial v}{\partial t}\frac{\partial\bar{u}}{\partial t}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla v\cdot\nabla\bar{u}\,\mathrm{d}x\mathrm{d}t\\ &+\int_{\Gamma_1}\int_{t_1}^T v\frac{\partial\bar{u}}{\partial t}\,\mathrm{d}s\mathrm{d}t + \int_{\Gamma_2}\int_0^T v\frac{\partial\bar{u}}{\partial t}\,\mathrm{d}s\mathrm{d}t = 0,\end{aligned} \tag{51}$$

where $\bar{u}$ is any function in U^1 that fullfills the boundary conditions in equation (38d)-(38g). Equation (51) is the *variational formulation* for the *adjoint problem* of solving $v(x,t)$.

10.2.1 Strong solution to the adjoint problem

The strong solution to the adjoint problem can be found by rewriting the time-derivative terms in equation (51), using partial integration and initial conditions for v and $\bar{u}$, so that:

$$\begin{aligned}&\int_{\Gamma_1}\int_0^T (u-\tilde{u})\,\bar{u}\,\mathrm{d}s\mathrm{d}t + \int_\Omega\int_0^T \frac{\partial^2 v}{\partial t^2}\bar{u}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla v\cdot\nabla\bar{u}\,\mathrm{d}x\mathrm{d}t\\ &-\int_{\Gamma_1}\int_0^{t_1} 0\,\bar{u}\,\mathrm{d}s\mathrm{d}t - \int_{\Gamma_1}\int_{t_1}^T \frac{\partial v}{\partial t}\,\bar{u}\,\mathrm{d}s\mathrm{d}t - \int_{\Gamma_2}\int_0^T \frac{\partial v}{\partial t}\,\bar{u}\,\mathrm{d}s\mathrm{d}t - \int_{\Gamma_3}\int_0^T 0\,\bar{u}\,\mathrm{d}s\mathrm{d}t = 0.\end{aligned} \tag{52}$$

Identify the boundary conditions for $\frac{\partial v}{\partial n}$ and let $z_\delta(x)$ be a cut-off function with the property:

$$\int_\Omega f\,z_\delta\,\mathrm{d}x = \int_{\Gamma_1} f\,\mathrm{d}s. \tag{53}$$

Then equation (52) is simplified to:

$$\int_\Omega\int_0^T (u-\tilde{u})\,z_\delta\,\bar{u}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T \frac{\partial^2 v}{\partial t^2}\bar{u}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T a\,\nabla v\cdot\nabla\bar{u}\,\mathrm{d}x\mathrm{d}t - \int_\Gamma\int_0^T \frac{\partial v}{\partial n}\,\bar{u}\,\mathrm{d}s\mathrm{d}t = 0. \tag{54}$$

Use Gauss divergence theorem and that $-\nabla\cdot(a\bar{u}\nabla v) = -a\nabla\bar{u}\cdot\nabla v - \bar{u}\,\nabla\cdot(a\nabla v)$ and simplify to:

$$\int_\Omega\int_0^T (u-\tilde{u})\,z_\delta\,\bar{u}\,\mathrm{d}x\mathrm{d}t + \int_\Omega\int_0^T \frac{\partial^2 v}{\partial t^2}\bar{u}\,\mathrm{d}x\mathrm{d}t - \int_\Omega\int_0^T \nabla\cdot(a\,\nabla v)\,\bar{u}\,\mathrm{d}x\mathrm{d}t = 0, \tag{55}$$

which implies that the strong solution to $v(x,t)$ fulfills:

$$\begin{aligned}
\frac{\partial^2 v}{\partial t^2} - \nabla \cdot (a\,\nabla v) &= -(u-\tilde{u})\,z_\delta && \text{in } \Omega \times (0,T),\\
v(x,T) &= 0 && \text{in } \Omega,\\
\frac{\partial v}{\partial t}(x,T) &= 0 && \text{in } \Omega,\\
\frac{\partial v}{\partial n} &= 0 && \text{on } \Gamma_1 \times (0,t_1],\\
\frac{\partial v}{\partial n} &= \frac{\partial v}{\partial t} && \text{on } \Gamma_1 \times (t_1,T),\\
\frac{\partial v}{\partial n} &= \frac{\partial v}{\partial t} && \text{on } \Gamma_2 \times (0,T),\\
\frac{\partial v}{\partial n} &= 0 && \text{on } \Gamma_3 \times (0,T).
\end{aligned} \tag{56}$$

10.3 Differentiating $L(u,v,a)$ with respect to a

Let (u,v,a) be the stationary point that minimizes the Lagrangian functional and compute:

$$\frac{\mathrm{d}L}{\mathrm{d}\alpha}(u,v,a+\alpha\bar{a}) = \gamma \int_\Omega (a+\alpha\bar{a}-a_0)\bar{a}\,\mathrm{d}x + \int_\Omega \int_0^T \bar{a}\,\nabla u \cdot \nabla v\,\mathrm{d}x\mathrm{d}t,$$

where α is some scalar and $\bar{a}$ is any function in A. Let $\alpha \to 0$ to get $\frac{\partial L}{\partial a}(u,v,a)(\bar{a}) = 0$, which is:

$$\int_\Omega \int_0^T (\nabla u \cdot \nabla v)\bar{a}\,\mathrm{d}x\mathrm{d}t + \gamma \int_\Omega (a-a_0)\bar{a}\,\mathrm{d}x = 0, \tag{57}$$

where $\bar{a}$ is any function in A. Equation (57) is the *variational formulation* for solving $a(x)$.

10.3.1 Strong solution to the conductivity coefficient

Clearly the variational formulation in equation (57) corresponds to the strong solution of $a(x)$ that fulfills:

$$\begin{aligned}
\int_0^T \nabla u \cdot \nabla v\,\mathrm{d}t + \gamma(a-a_0) &= 0 && \forall x \in \Omega,\\
a(x) &\in [1, a_{max}] && \forall x \in \Omega,\\
a(x) &= 1 && \forall x \in \Gamma.
\end{aligned} \tag{58}$$

where $(u(a), v(a)) \in U^1 \times V^1$ are the stationary points that minimizes the Lagrangian functional. Define the functions $h(x)$ and $g(x)$ as:

$$\begin{aligned}
h(x) &= \int_0^T \nabla u \cdot \nabla v\,\mathrm{d}t,\\
g(x) &= \int_0^T \nabla u \cdot \nabla v\,\mathrm{d}t + \gamma(a-a_0).
\end{aligned} \tag{59}$$

11 Finite element function spaces

The weak solutions to the three variational formulations in equation (50), (51) and (57), are approximated by choosing functions that are continuous piecewise linear, and piecewise constant, over the discretized domain $\Omega \times (0,T)$.

Let K_h denote the discretized domain of $\Omega \in \mathbb{R}^d$, which consists of adjoint triangles when $d = 2$ and of tetrahedra when $d = 3$. Each subset (triangle or tetrahedra) K_i is known as an *element* to K_h, so that $K_h = \bigcup_{i=1}^{q} K_i$. The triangulation K_h consists of q elements and have m nodes.

Let J_τ denote the discretized interval of $J = [0,T]$, with constant step-size $\tau = t_j - t_{j-1}\ \forall j \in \{2,\dots,n\}$. J_τ have n nodes, and each sub-interval $J_j = [t_{j-1}, t_j]$ for $j \in \{2,\dots,n\}$ are known as *elements* to J_τ.

Let $P_1(K_h)$ denote the set of all functions that are continuous piecewise linear on K_h and let $P_1(J_\tau)$ denote the set of all functions that continuous piecewise linear on J_τ. Let $P_0(K_h)$ denote the set of all functions that are piecewise constant on K_h. Define the finite element spaces used for approximating the weak solutions as:

$$\begin{aligned} U_h^1 &= \Big\{ w \in U^1 : w \in P_1(K_h) \times P_1(J_\tau) \Big\}, \\ V_h^1 &= \Big\{ w \in V^1 : w \in P_1(K_h) \times P_1(J_\tau) \Big\}, \\ A_h &= \Big\{ w \in A : w \in P_0(K_h) \Big\}. \end{aligned} \tag{60}$$

In the FEM we choose approximating functions $u(x,t) \approx u_h(x,t) \in U_h^1$, $v(x,t) \approx v_h(x,t) \in V_h^1$ and $a(x) \approx a_h(x) \in A_h$. The approximated functions are written shortly as an superposition of basis functions:

$$\begin{aligned} u_h(x,t) &= \sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}\phi_i(x)\varphi_j(t) = \phi^T U \varphi, \\ v_h(x,t) &= \sum_{i=1}^{m}\sum_{j=1}^{n} v_{ij}\phi_i(x)\varphi_j(t) = \phi^T V \varphi, \\ a_h(x) &= \sum_{k=1}^{q} a_k \eta_k(x) = \eta^T a, \end{aligned} \tag{61}$$

where $\{\phi_i(x)\}_{i=1}^m$ is the set of linear basis functions on K_h with m nodes, $\{\varphi_j(t)\}_{j=1}^n$ is the set of linear basis functions on J_τ with n nodes and $\{\eta_k(x)\}_{k=1}^q$ is the set of constant basis functions on K_h with q elements. In the notation above, ϕ, φ and η are vectors defined as $\phi^T = [\phi_1(x),\dots,\phi_m(x)]$, $\varphi^T = [\varphi_1(t),\dots,\varphi_n(t)]$ and $\eta^T = [\eta_1(x),\dots,\eta_q(x)]$. The $m \times n$ matrices U, V and q vector a contains the nodal values to the basis functions. The constant basis functions $\eta_i(x)$ on K_h are defined as:

$$\eta_i(x) = \begin{cases} 1 & \text{if } x \in K_i, \\ 0 & \text{if } x \notin K_i. \end{cases} \tag{62}$$

The linear basis functions ϕ on $K_h \in \mathbb{R}^2$, and linear basis functions φ on $J_\tau \in \mathbb{R}$, and resulting FEM matrices used in numerical computations are explained in more detail in the appendix.

12 Finite element approximations to the variational formulations

In this section the equation systems for approximating solutions of the variational formulations are derived, by substituting the functions u_h, v_h and a_h, that are spanned by piecewise linear and piecewise constant basis functions as described earlier, into the three variational formulations.

By choosing the arbitrary functions $\bar{u}$, $\bar{v}$ as one of the pair $\phi_i(x)\varphi_j(t)$, and $\bar{a}$ as one of the $\eta_i(x)$, in each respective set of basis functions, it is shown that the variational formulations in equation (50), (51) and (57) gives $m \times n$ and q equations from where the nodal values to u_h, v_h and a_h can be solved. The resulting equation systems can be written shortly with matrix-matrix multiplication, using the matrices and vectors defined by:

$$
\begin{aligned}
m_{ik} &= \int_{K_h} \phi_k(x)\phi_i(x)\,\mathrm{d}x && \text{(global mass matrix on } K_h\text{)},\\
k_{ik} &= \int_{K_h} a(x)\,\nabla\phi_k(x)\cdot\nabla\phi_i(x)\,\mathrm{d}x && \text{(global stiffness matrix on } K_h\text{)},\\
m^{(1)}_{ik} &= \int_{\Gamma_1} \phi_k(x)\phi_i(x)\,\mathrm{d}s && \text{(mass matrix along } \Gamma_1\text{)},\\
m^{(2)}_{ik} &= \int_{\Gamma_2} \phi_k(x)\phi_i(x)\,\mathrm{d}s && \text{(mass matrix along } \Gamma_2\text{)},\\
b_i &= \int_{\Gamma_1} \phi_i(x)\,\mathrm{d}s && \text{(vector along } \Gamma_1\text{)},\\
r_{jl} &= \int_0^T \varphi_l(t)\varphi_j(t)\,\mathrm{d}t && \text{(global mass matrix on } J_\tau\text{)},\\
r^{(1)}_{jl} &= \int_0^{t_1} \varphi_l(t)\varphi_j(t)\,\mathrm{d}t && \text{(“partial” mass matrix on } J_\tau\text{)},\\
c_{jl} &= \int_0^T \varphi_l'(t)\varphi_j(t)\,\mathrm{d}t && \text{(global convection matrix on } J_\tau\text{)},\\
c^{(1)}_{jl} &= \int_{t_1}^T \varphi_l'(t)\varphi_j(t)\,\mathrm{d}t && \text{(“partial” convection matrix on } J_\tau\text{)},\\
s_{jl} &= \int_0^T \varphi_l'(t)\varphi_j'(t)\,\mathrm{d}t && \text{(global stiffness matrix on } J_\tau\text{)},
\end{aligned}
\tag{63}
$$

where $i,k \in \{1,\ldots,m\}$ and $j,l \in \{1,\ldots,n\}$.

The basis functions and matrices are computed in detail in the appendix, for $J_\tau \in \mathbb{R}$ and $K_h \in \mathbb{R}^2$. When Ω is a square domain in $\mathbb{R}^2$, the mass matrices along Γ_1 and Γ_2 are analogue to mass matrices in one dimension, with non-zero entries corresponding to nodes that are on Γ_1 or Γ_2. All the matrices are symmetric except convection matrices C and C_1.

12.1 Finite element approximation to the forward problem

Let $u \approx u_h$ be spanned by the two sets of basis functions, let $\bar{v}$ be each pair of those basis functions and interpolate $p(t)$ into $p \approx p_\tau$, so that:

$$\begin{aligned} u_h(x,t) &= \sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl}\phi_k(x)\varphi_l(t), \\ p_\tau(t) &= \sum_{l=1}^{n} p_l\varphi_l(t) && p_l = p(t_l), \\ \bar{v}(x,t) &= \phi_i(x)\varphi_j(t) && i \in \{1,\dots,m\}, j \in \{1,\dots,n\}. \end{aligned} \tag{64}$$

Insert the approximations into equation (50), the variational formulation for u, and simplify to get $m \times n$ equations. We get that for each $i \in \{1,\dots,m\}$ and $j \in \{1,\dots,n\}$:

$$\begin{aligned} &-\sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl} \int_{K_h} \phi_k(x)\phi_i(x)\,\mathrm{d}x \int_0^T \varphi_l'(t)\varphi_j'(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl} \int_{K_h} a(x)\,\nabla\phi_k(x)\cdot\nabla\phi_i(x)\,\mathrm{d}x \int_0^T \varphi_l(t)\varphi_j(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl} \int_{\Gamma_1} \phi_k(x)\phi_i(x)\,\mathrm{d}s \int_{t_1}^T \varphi_l'(t)\varphi_j(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl} \int_{\Gamma_2} \phi_k(x)\phi_i(x)\,\mathrm{d}s \int_0^T \varphi_l'(t)\varphi_j(t)\,\mathrm{d}t \\ &\quad= \sum_{l=1}^{n} p_l \int_{\Gamma_1} \phi_i(x)\,\mathrm{d}s \int_0^{t_1} \varphi_l(t)\varphi_j(t)\,\mathrm{d}t. \end{aligned} \tag{65}$$

Identify the matrices and vectors from equation (63) and write the equation system in (65) as:

$$\sum_{k=1}^{m}\sum_{l=1}^{n}\Big(-m_{ik}\,u_{kl}\,s_{jl} + k_{ik}\,u_{kl}\,r_{jl} + m_{ik}^{(1)}\,u_{kl}\,c_{jl}^{(1)} + m_{ik}^{(2)}\,u_{kl}\,c_{jl}\Big) = \sum_{l=1}^{n} b_i\,p_l\,r_{jl}^{(1)}, \tag{66}$$

for each $i \in \{1,\dots,m\}$ and $j \in \{1,\dots,n\}$, which can be written shortly using matrix-matrix multiplication as:

$$-MUS + KUR + M_1UC_1^T + M_2UC^T = b\,p^T R_1. \tag{67}$$

12.2 Finite element approximation to the adjoint problem

Let v_h, u_h, $\tilde{u}_h$ be spanned by the two sets of basis functions and chose $\bar{u}$ as each pair of $\phi_i(x)\varphi_j(t)$, that is:

$$\begin{aligned} v_h(x,t) &= \sum_{k=1}^{m}\sum_{l=1}^{n} v_{kl}\phi_k(x)\varphi_l(t), \\ u_h(x,t) &= \sum_{k=1}^{m}\sum_{l=1}^{n} u_{kl}\phi_k(x)\varphi_l(t), \\ \tilde{u}_h(x,t) &= \sum_{k=1}^{m}\sum_{l=1}^{n} \tilde{u}_{kl}\phi_k(x)\varphi_l(t), \\ \bar{u}(x,t) &= \phi_i(x)\varphi_j(t) \qquad\qquad i \in \{1,\dots,m\}, j \in \{1,\dots,n\}. \end{aligned} \tag{68}$$

The functions u_h and $\tilde{u}_h$ are assumed to be known. Insert the approximations into equation (51), the variational formulation for v, to get $m \times n$ equations. We get that for each $i \in \{1,\dots,m\}$ and $j \in \{1,\dots,n\}$:

$$\begin{aligned} &-\sum_{k=1}^{m}\sum_{l=1}^{n} v_{kl} \int_{K_h} \phi_k(x)\phi_i(x)\,\mathrm{d}x \int_0^T \varphi_l'(t)\varphi_j'(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} v_{kl} \int_{K_h} a(x)\,\nabla\phi_k(x)\cdot\nabla\phi_i(x)\,\mathrm{d}x \int_0^T \varphi_l(t)\varphi_j(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} v_{kl} \int_{\Gamma_1} \phi_k(x)\phi_i(x)\,\mathrm{d}s \int_{t_1}^T \varphi_l(t)\varphi_j'(t)\,\mathrm{d}t \\ &+\sum_{k=1}^{m}\sum_{l=1}^{n} v_{kl} \int_{\Gamma_2} \phi_k(x)\phi_i(x)\,\mathrm{d}s \int_0^T \varphi_l(t)\varphi_j'(t)\,\mathrm{d}t \\ &= -\sum_{k=1}^{m}\sum_{l=1}^{n} (u_{kl}-\tilde{u}_{kl}) \int_{\Gamma_1} \phi_k(x)\phi_i(x)\,\mathrm{d}s \int_0^T \varphi_l(t)\varphi_j(t)\,\mathrm{d}t. \end{aligned} \tag{69}$$

Identify the matrices and vectors from equation (63) and write the equation system in (69) as:

$$\sum_{k=1}^{m}\sum_{l=1}^{n}\Big(-m_{ik}\,v_{kl}\,s_{jl}+k_{ik}\,v_{kl}\,r_{jl}+m_{ik}^{(1)}\,v_{kl}\,c_{lj}^{(1)}+m_{ik}^{(2)}\,v_{kl}\,c_{lj}\Big) = -\sum_{k=1}^{m}\sum_{l=1}^{n} m_{ik}^{(1)}\,(u_{kl}-\tilde{u}_{kl})\,r_{jl}, \tag{70}$$

for each $i \in \{1,\dots,m\}$ and $j \in \{1,\dots,n\}$, which can be written shortly using matrix-matrix multiplication as:

$$-MVS + KVR + M_1VC_1 + M_2VC = -M_1(U-\tilde{U})R. \tag{71}$$

Note that the convection matrices C_1 and C are not transposed.

If there is also transmitted data available, so that the Tikhonov functional uses both backscatter and transmitted data, the equation system gets the additional term $-M_2(U-\tilde{U})R$ on the left hand side and can be defined as:

$$-MVS + KVR + M_1VC_1 + M_2VC = -(M_1+M_2)(U-\tilde{U})R. \tag{72}$$

The $m \times n$ matrix $U-\tilde{U}$ can be defined as having zero-rows corresponding to the nodes where no data is collected.

12.3 Finite element approximation to the conductivity coefficient

Let a_h and initial guess a_0 be spanned by the set of piecewise constant basis functions on K_h. The function $h_h(x)$ is piecewise constant in each element because u_h and v_h are continuous piecewise linear on K_h, meaning that $h(x) \approx h_h(x)$ can be defined by:

$$h_i = \int_{J_\tau} \nabla u_h \cdot \nabla v_h \, \mathrm{d}t \qquad\qquad x \in K_i \text{ for } i \in \{1, \ldots, q\}. \tag{73}$$

Chose $\bar{a}(x)$ as one of the piecewise constant basis functions η_i. The functions are then defined by:

$$\begin{aligned} a_h(x) &= \sum_{j=1}^{q} a_j \eta_j(x), \\ a_0(x) &= \sum_{j=1}^{q} a_j^{(0)} \eta_j(x), \\ h_h(x) &= \sum_{j=1}^{q} h_j \eta_j(x), \\ \bar{a} &= \eta_i(x) \qquad\qquad i \in \{1, \ldots, q\}. \end{aligned} \tag{74}$$

Insert the approximations into equation (57), the variational formulation for a, to get q equations that are simplified to:

$$\sum_{j=1}^{q} h_j \int_{K_h} \eta_j(x)\eta_i(x) \, \mathrm{d}x + \gamma \sum_{j=1}^{q} (a_j - a_j^{(0)}) \int_{K_h} \eta_j(x)\eta_i(x) \, \mathrm{d}x = 0, \quad . \tag{75}$$

for each $i \in \{1, \ldots, q\}$. Define the symmetric matrix E by:

$$e_{ij} = \int_{K_h} \eta_j(x)\eta_i(x) \, \mathrm{d}x \quad i, j \in \{1, \ldots, q\}. \tag{76}$$

Clearly we have that E is diagonal since $\eta_i \eta_j = \delta_{ij}$. When $K_h \in \mathbb{R}^2$, e_{ii} equals the area of element K_i and when $K_h \in \mathbb{R}^3$ it equals the volume of element K_i . The equation system can be written shortly as $Eh + \gamma E(a - a_0) = 0$, which is simplified to:

$$h + \gamma(a - a_0) = 0. \tag{77}$$

It might seems unnecessary to go through these steps, but if basis functions of higher order are chosen to span u_h, v_h, a_h the result will differ. The result will also differ if a_h is chosen from $P_1(K_h)$, the set of continuous piecewise linear functions on K_h.

13 Solving the finite element system of equations

The two FEM equations (67) and (71) for $m \times n$ matrices U and V can be solved column-by-column by the steps described below. The FEM equation (77) for q vector a can be computed once U and V are solved.

The general idea in solving U and V is to multiply in the j:th column of the tridiagonal $n \times n$ mass-, convection- and stiffness-matrices of J_τ into $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$, and then solve the linear system of equations that arise, using standard methods such as Gaussian elimination or Cholesky decomposition. Let $j \in \{1, 2, \ldots, k-1, k, k+1, \ldots, n-1, n\}$ where k is the node number corresponding to the time t_1 in the forward problem, which is the end-time for $p(t)$. We then have that:

$$\begin{aligned}(-MU)s_j + (KU)r_j + (M_1U)\,[C_1^T]_j + (M_2U)\,[C^T]_j &= (bp^T)r_j^{(1)},\\ (-MV)s_j + (KV)r_j + (M_1V)c_j^{(1)} + (M_2v)c_j &= (-M_1(U-\tilde{U}))r_j,\end{aligned} \tag{78}$$

for each j:th column vector, where $[C_1^T]_j$ and $[C^T]_j$ denotes the j:th column of the transposed convection matrices.

The first column vector $u_1 = 0$ in U is known because of initial condition $u_h(x, 0) = 0$, the other columns in U are computed by increasing j upwards from 1. All the column vectors in U can be computed for any given conductivity coefficient a, the exact or reconstructed. The only $m \times m$ matrix that is dependent on conductivity a is the stiffness matrix K.

The last column vector $v_n = 0$ in V is known because of end condition $v_h(x, T) = 0$, the other columns in V are computed by decreasing j from n. V is dependent on $U - \tilde{U}$ and can be computed once U is fully determined for the given conductivity a.

In the numerical examples the measured data $\tilde{U}$ on the boundary is generated by computing U^*, using the exact conductivity a^*, with noise added according to equation (43).

13.1 Computing the columns in U

Compute the sequence of column vectors $u_2, u_3, \ldots, u_n$ by solving u_{j+1} in the following steps:

step 1. $j = 1$

$$\begin{aligned}&-M(u_j - u_{j+1}) + \frac{\tau^2}{6}K(2u_j + u_{j+1}) + \frac{\tau}{2}M_2(-u_j + u_{j+1})\\ =&\frac{\tau^2}{6}b\,(2p_j + p_{j+1}).\end{aligned} \tag{79}$$

Use that $u_h(x, 0) = 0 \Rightarrow u_1 = 0$, and simplify to:

$$\underbrace{\left(M + \frac{\tau^2}{6}K + \frac{\tau}{2}M_2\right)}_{A_3} u_{j+1} = \underbrace{\frac{\tau^2}{6}b\,(2p_j + p_{j+1})}_{f_j}. \tag{80}$$

Identify matrix A_3 and vector f_j; u_{j+1} can be solved from L.S.E:

$$A_3 u_{j+1} = f_j. \tag{81}$$

step 2. $\forall j = 2, \ldots, k-1$

$$\begin{aligned}&-M(-u_{j-1} + 2u_j - u_{j+1}) + \frac{\tau^2}{6}K(u_{j-1} + 4u_j + u_{j+1})\\ &+\frac{\tau}{2}M_2(-u_{j-1} + u_{j+1}) = \frac{\tau^2}{6}b\,(p_{j-1} + 4p_j + p_{j+1}).\end{aligned} \tag{82}$$

Simplify to:

$$\begin{aligned}&\underbrace{\Big(M+\frac{\tau^2}{6}K-\frac{\tau}{2}M_2\Big)}_{A_1}u_{j-1}+\underbrace{\Big(-2M+\frac{4\tau^2}{6}K\Big)}_{A_2}u_j\\&+\underbrace{\Big(M+\frac{\tau^2}{6}K+\frac{\tau}{2}M_2\Big)}_{A_3}u_{j+1}=\underbrace{\frac{\tau^2}{6}b\left(p_{j-1}+4p_j+p_{j+1}\right)}_{f_j}.\end{aligned}\tag{83}$$

Identify matrices A_1, A_2, A_3 and vector f_j; u_{j+1} can be solved from L.S.E:

$$A_3u_{j+1}=f_j-A_1u_{j-1}-A_2u_j.\tag{84}$$

step 3. $j=k$

$$\begin{aligned}&-M(-u_{j-1}+2u_j-u_{j+1})+\frac{\tau^2}{6}K(u_{j-1}+4u_j+u_{j+1})\\&+\frac{\tau}{2}M_1(-u_j+u_{j+1})+\frac{\tau}{2}M_2(-u_{j-1}+u_{j+1})=\frac{\tau^2}{6}b\left(p_{j-1}+2p_j\right).\end{aligned}\tag{85}$$

Simplify to:

$$\begin{aligned}&\underbrace{\Big(M+\frac{\tau^2}{6}K-\frac{\tau}{2}M_2\Big)}_{A_1}u_{j-1}+\underbrace{\Big(-2M+\frac{4\tau^2}{6}K-\frac{\tau}{2}M_1\Big)}_{A_2}u_j\\&+\underbrace{\Big(M+\frac{\tau^2}{6}K+\frac{\tau}{2}(M_1+M_2)\Big)}_{A_3}u_{j+1}=\underbrace{\frac{\tau^2}{6}b\left(p_{j-1}+2p_j\right)}_{f_j}.\end{aligned}\tag{86}$$

Identify matrices A_1, A_2, A_3 and vector f_j; u_{j+1} can be solved from L.S.E:

$$A_3u_{j+1}=f_j-A_1u_{j-1}-A_2u_j.\tag{87}$$

step 4. $\forall j=k+1,\ldots,n-1$

$$\begin{aligned}&-M(-u_{j-1}+2u_j-u_{j+1})+\frac{\tau^2}{6}K(u_{j-1}+4u_j+u_{j+1})\\&+\frac{\tau}{2}M_1(-u_{j-1}+u_{j+1})+\frac{\tau}{2}M_2(-u_{j-1}+u_{j+1})=0.\end{aligned}\tag{88}$$

Simplify to:

$$\begin{aligned}&\underbrace{\Big(M+\frac{\tau^2}{6}K-\frac{\tau}{2}(M_1+M_2)\Big)}_{A_1}u_{j-1}+\underbrace{\Big(-2M+\frac{4\tau^2}{6}K\Big)}_{A_2}u_j\\&+\underbrace{\Big(M+\frac{\tau^2}{6}K+\frac{\tau}{2}(M_1+M_2)\Big)}_{A_3}u_{j+1}=0.\end{aligned}\tag{89}$$

Identify matrices A_1, A_2 and A_3; u_{j+1} can be solved from L.S.E:

$$A_3u_{j+1}=-A_1u_{j-1}-A_2u_j.\tag{90}$$

Note that M is sparse symmetric positive definite band matrix and K, M_1, M_2 are sparse symmetric positive *semi*-definite band matrices. Therefore A_3 is sparse S.P.D. band matrix in all four steps for computing all columns in U.

13.2 Computing the columns in V

Define $F = U - \tilde{U} = [f_1, \ldots, f_n]$. Compute the sequence of column vectors $v_{n-1}, v_{n-2}, \ldots, v_1$ by solving v_{j-1} in the following steps:

step 1. $j = n$

$$\begin{aligned} &-M(-v_{j-1} + v_j) + \frac{\tau^2}{6}K(v_{j-1} + 2v_j) \\ &+\frac{\tau}{2}(M_1 + M_2)(v_{j-1} + v_{j+1}) = -\frac{\tau^2}{6}M_1(f_{j-1} + 2f_j). \end{aligned} \tag{91}$$

Use that $v_h(x, T) = 0 \Rightarrow v_n = 0$, and simplify to:

$$\underbrace{\Big(M + \frac{\tau^2}{6}K + \frac{\tau}{2}(M_1 + M_2)\Big)}_{A_1} v_{j-1} = \underbrace{-\frac{\tau^2}{6}M_1(f_{j-1} + 2f_j)}_{c_j}. \tag{92}$$

Identify matrix A_1 and vector c_j; v_{j-1} can be solved from L.S.E:

$$A_1 v_{j-1} = c_j. \tag{93}$$

step 2. $\forall j = n-1, \ldots, k+1$

$$\begin{aligned} &-M(-v_{j-1} + 2v_j - v_{j+1}) + \frac{\tau^2}{6}K(v_{j-1} + 4v_j + v_{j+1}) \\ &+\frac{\tau}{2}(M_1 + M_2)(v_{j-1} - v_{j+1}) = -\frac{\tau^2}{6}M_1(f_{j-1} + 4f_j + f_{j+1}). \end{aligned} \tag{94}$$

Simplify to:

$$\begin{aligned} &\underbrace{\Big(M + \frac{\tau^2}{6}K + \frac{\tau}{2}(M_1 + M_2)\Big)}_{A_1} v_{j-1} + \underbrace{\Big(-2M + \frac{4\tau^2}{6}K\Big)}_{A_2} v_j \\ &+\underbrace{\Big(M + \frac{\tau^2}{6}K - \frac{\tau}{2}(M_1 + M_2)\Big)}_{A_3} v_{j+1} = \underbrace{-\frac{\tau^2}{6}M_1(f_{j-1} + 4f_j + f_{j+1})}_{c_j}. \end{aligned} \tag{95}$$

Identify matrices A_1, A_2, A_3 and vector c_j; v_{j-1} can be solved from L.S.E:

$$A_1 v_{j-1} = c_j - A_2 v_j - A_3 u_{j+1}. \tag{96}$$

step 3. $j = k$

$$\begin{aligned} &-M(-v_{j-1} + 2v_j - v_{j+1}) + \frac{\tau^2}{6}K(v_{j-1} + 4v_j + v_{j+1}) \\ &+\frac{\tau}{2}M_1(-v_j - v_{j+1}) + \frac{\tau}{2}M_2(v_{j-1} - v_{j+1}) = -\frac{\tau^2}{6}M_1(f_{j-1} + 4f_j + f_{j+1}). \end{aligned} \tag{97}$$

Simplify to:

$$\begin{aligned} &\underbrace{\Big(M + \frac{\tau^2}{6}K + \frac{\tau}{2}M_2\Big)}_{A_1} v_{j-1} + \underbrace{\Big(-2M + \frac{4\tau^2}{6}K - \frac{\tau}{2}M_1\Big)}_{A_2} v_j \\ &+\underbrace{\Big(M + \frac{\tau^2}{6}K - \frac{\tau}{2}(M_1 + M_2)\Big)}_{A_3} v_{j+1} = \underbrace{-\frac{\tau^2}{6}M_1(f_{j-1} + 4f_j + f_{j+1})}_{c_j}. \end{aligned} \tag{98}$$

Identify matrices A_1, A_2, A_3 and vector c_j; v_{j-1} can be solved from L.S.E:

$$A_1 v_{j-1} = c_j - A_2 v_j - A_3 u_{j+1}. \tag{99}$$

step 4. $\forall j = k-1, \ldots, 2$

$$\begin{aligned} &-M(-v_{j-1} + 2v_j - v_{j+1}) + \frac{\tau^2}{6} K(v_{j-1} + 4v_j + v_{j+1}) \\ &+\frac{\tau}{2} M_2(v_{j-1} - v_{j+1}) = -\frac{\tau^2}{6} M_1(f_{j-1} + 4f_j + f_{j+1}). \end{aligned} \tag{100}$$

Simplify to:

$$\begin{aligned} &\underbrace{\left(M + \frac{\tau^2}{6}K + \frac{\tau}{2}M_2\right)}_{A_1} v_{j-1} + \underbrace{\left(-2M + \frac{4\tau^2}{6}K\right)}_{A_2} v_j \\ &+ \underbrace{\left(M + \frac{\tau^2}{6}K - \frac{\tau}{2}M_2\right)}_{A_3} v_{j+1} = \underbrace{-\frac{\tau^2}{6} M_1(f_{j-1} + 4f_j + f_{j+1})}_{c_j}. \end{aligned} \tag{101}$$

Identify matrices A_1, A_2, A_3 and vector c_j; v_{j-1} can be solved from L.S.E:

$$A_1 v_{j-1} = c_j - A_2 v_j - A_3 u_{j+1}. \tag{102}$$

Note that M is sparse symmetric positive definite band matrix and K, M_1, M_2 are sparse symmetric positive *semi*-definite band matrices. Therefore A_1 is sparse S.P.D. band matrix in all four steps for computing all columns in V.

If both backscatter and transmitted data are used, then $M_1 \to (M_1 + M_2)$ in the terms for computing c_j.

14 Minimization methods applied on FEM solutions

This section derives some of the functions and parameters that needs to be computed so that fixed point iterations (FPI), conjugate gradient method (CGM) and iteratively regularized conjugate gradient method (IR-CGM) can be implemented on the minimization problem. The algorithms are then defined for the minimization problem in section 15. The FPI and CGM are described for general real valued functions $f(x) : \mathbb{R}^n \to \mathbb{R}$ in section 3. To implement these algorithms on the Lagrangian functional we need to identify a fixed point, and be able to determine step-sizes in the gradient methods.

The functional that is minimized to solve the inverse problem is the Lagrangian $L(u, v, a)$, defined in equation (48), with respect to a. The functions $(u_h, v_h, a_h) \in U_h^1 \times V_h^1 \times A_h$ are chosen from the approximate function spaces defined in equation (60), and computed using FEM as described in the previous section. Let $a_0, a^* \in A_h$ denote the initial guess and exact solution respectively. The sequential solutions to $u_h^{(k)}$, $v_h^{(k)}$ and $a_h^{(k)}$ are determined by the nodal values in the $m \times n$ matrices U_k, V_k and by the element values in the q vector a_k, such that:

$$\begin{aligned} u_h^{(k)}(x,t) &= \sum_{i=1}^{m}\sum_{j=1}^{n} u_{ij}^{(k)}\phi_i(x)\varphi_j(t) = \phi(x)U_k\varphi, \\ v_h^{(k)}(x,t) &= \sum_{i=1}^{m}\sum_{j=1}^{n} v_{ij}^{(k)}\phi_i(x)\varphi_j(t) = \phi^T V_k\varphi, \\ a_h^{(k)}(x) &= \sum_{i=1}^{q} a_i^{(k)}\eta_k(x) = \eta^T a_k, \end{aligned} \tag{103}$$

Define the sequential q vectors h_k and g_k by:

$$\begin{aligned} h_h^{(k)}(x) &= \int_0^T \nabla u_h^{(k)} \cdot \nabla v_h^{(k)}\,\mathrm{d}t = \eta^T h_k, \\ g_h^{(k)}(x) &= \int_0^T \nabla u_h^{(k)} \cdot \nabla v_h^{(k)}\,\mathrm{d}t + \gamma(a_h^{(k)} - a_0) = \eta^T g_k. \end{aligned} \tag{104}$$

Define the scalar product (f, g) of two real valued functions, and the L-2 norm $\|f\|_{L_2}$, by:

$$\begin{aligned} (f,g) &= \int_{K_h} f\,g\,\mathrm{d}x, \\ \|f\|_{L_2}^2 &= (f,f). \end{aligned} \tag{105}$$

The scalar product of two piecewise constant functions such as $a_h(x) = \eta(x)^T a$, $h_h(x) = \eta(x)^T h$ can easily computed using matrix multiplication. As before, $\eta(x)^T$ is a vector containing the piecewise constant functions for each element, and a, h are constant vectors containing the constant value of the coefficients in each element. The scalar products are then computed as:

$$(a_h(x), h_h(x)) = (a^T\eta(x), \eta(x)^T h) = a^T(\eta(x), \eta(x)^T)h = a^T A h, \tag{106}$$

where A is a diagonal symmetric positive definite matrix, where elements a_{ij} are defined by:

$$a_{ij} = \int_{K_h} \eta_j\eta_i\,\mathrm{d}x = \begin{cases} \mathrm{area}(K_i) & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{107}$$

for $K_h \in \mathbb{R}^2$. If $K_h \in \mathbb{R}^3$, then $a_{ii} = \mathrm{volume}(K_i)$. As before, K_i denotes the i:th element of the discretized domain K_h.

14.1 Fixed point to Lagrangian

Algorithm of fixed point iteration can be applied to the minimizing problem using equation (77), which states that $g(x) = 0$ on the stationary point that minimizes the Lagrangian functional. By rewriting the equation we get that:

$$a^* = a_0 - \frac{1}{\gamma} h(a^*) \quad \gamma \in (0, 1], \tag{108}$$

which clearly is a fixed point to $a_0 - h(a)/\gamma = f(a)$. The FPI sequence $a_{k+1} = f(a_k)$ converges to a if f is Lipschitz continuous, which depends on K_h, J_τ, $p(t)$, a_0 and γ. The regularization parameter γ acts as an step-size and is chosen as $\gamma = 1$, otherwise the sequence will diverge. Assuming that each a_{k+1} gets closer to the exact a^*, let $a_0 \to a_k$ be the last guess. The the FPI is to compute the sequence:

$$a_{k+1} = a_k - h_k, \tag{109}$$

until convergence criteria is satisfied.

14.2 Gradient methods applied on Lagrangian

In both the gradient method (GM) and conjugate gradient method (CGM), the sequence of a_{k+1} is computed by $a_{k+1} = a_k + \alpha_k d_k$, where the step-size $\alpha_k \in (0, 1]$ is chosen so that $\alpha_k = \operatorname{argmin}\{L(u_k, v_k, a_k + \alpha d_k) : \alpha \geq 0\}$, and d_k is the chosen direction.

In the gradient method d_k is chosen as the steepest descent $d_k = -\frac{\partial L}{\partial a}(a_k) = -g_k$, and in the conjugate gradient method a fraction $\beta \in (0, 1]$ of the last direction is added so that $d_k = -g_k + \beta_k d_{k-1}$.

Let $G(\alpha) = L(u, v, a + \alpha d)$, then the step-size that minimizes $G(\alpha)$ can be estimated "directly" for any given (u, v, a, d), if we assume that u and v are not dependent on a. Using the definition of $L(u, v, a)$ in equation (48), the functional $G(\alpha)$ is defined as:

$$G(\alpha) = \int_\Omega \int_0^T (a + \alpha d) \nabla u \cdot \nabla v \, \mathrm{d}x\mathrm{d}t + \frac{1}{2}\gamma \int_\Omega (a + \alpha d - a_0)^2 \, \mathrm{d}x + T(u, v), \tag{110}$$

where $T(u, v)$ is a collection of the terms not dependent on αd. Differentiate by α to get:

$$\frac{\mathrm{d}G}{\mathrm{d}\alpha} = \int_\Omega \int_0^T d \nabla u \cdot \nabla v \, \mathrm{d}x\mathrm{d}t + \gamma \int_\Omega (a + \alpha d - a_0) \, d \, \mathrm{d}x = 0 \tag{111}$$

Use the definition for $h(x)$ and simplify to:

$$\int_\Omega h \, d \, \mathrm{d}x + \gamma \int_\Omega (a - a_0) \, d \, \mathrm{d}x + \gamma\alpha \int_\Omega d \, d \, \mathrm{d}x = 0 \tag{112}$$

Use the notation $(\cdot, \cdot)$ for scalar products, we get an estimate for the α that minimizes $G(\alpha)$ as:

$$\alpha = -\frac{(h, d) + \gamma(a - a_0, d)}{\gamma(d, d)} = -\frac{(g, d)}{\gamma(d, d)} \quad \gamma \in (0, 1], \tag{113}$$

for any given (u, v, a, d).

When deriving α for a real valued quadratic function $f(x)$ in the appendix, it is known that $f'(x_k)$ and $f'(x_{k-1})$ are orthogonal so that $(f'(x_k), f'(x_{k-1})) = 0$ for all k. But no such assumption can be made for gradients g_k and g_{k-1} from the Lagrangian functional, since u and v changes for each a.

For the CGM method the fraction β_k is chosen as:

$$\beta_k = \frac{\|g_k\|_{L_2}^2}{\|g_{k-1}\|_{L_2}^2} = \frac{(g_k, g_k)}{(g_{k-1}, g_{k-1})}. \tag{114}$$

Since the gradients g_k and g_{k-1} can not be assumed to be orthogonal, or that $\|g_k\|_{L_2} \leq \|g_{k-1}\|_{L_2}$, it might happen that $\beta > 1$ for some k. I observed this when running the CGM algorithm and subsequently the

sequence diverged to $\|a_k\|_{L_2} \to \infty$. To get around this problem, it is sufficient to limit $\beta \in (0, 1]$. Once β is always less or equal to one, the step-size α_k will always be non-negative. Therefore define β_k and α_k as:

$$\begin{aligned} \beta_k &= \min\Big\{\frac{(g_k, g_k)}{(g_{k-1}, g_{k-1})}, 1\Big\}, \\ \alpha_k &= \min\Big\{-\frac{(g_k, d_k)}{\gamma(d_k, d_k)}, 1\Big\}. \end{aligned} \tag{115}$$

The step-size needs to be limited from above too, since it is divided by γ.

If we can assume that each computed a_{k+1} gets closer to a^* for each iteration, we can let a_0 be the last guess a_{k-1}, so that gradient g_k is updated by:

$$\begin{aligned} g_k &= h_k + \gamma(a_k - a_{k-1}) \qquad k = 1, 2, \ldots, \\ g_0 &= h_k \qquad\qquad\qquad\qquad \text{(first iteration)}. \end{aligned} \tag{116}$$

This works quite well for nice problems with low noise.

14.3 Iteratively regularized gradient methods applied on Lagrangian

On the CG and CGM, the Lagrangian can be iteratively regularized by choosing a sequence of regularization parameters $\{\gamma_k\}$ that slowly converges to zero, using the rule:

$$\gamma_k = \frac{\gamma_0}{(k+1)^p} \quad \gamma_0, p, \in (0, 1], \tag{117}$$

and computing gradient $g_k = h_k + \gamma_k(a_k - a_0)$. This ensures that the Tikhonov term in the functional tends to zero for large k:s, instead of some positive constant. The step-size α_k is computed in the way described in equation (115) for each γ_k.

For theory behind choosing the sequence of regularization parameters, see section 2, that describes general theory about ill-posed inverse problems, and [6, 11].

15 Iterative algorithms for solving inverse problem

In this section we describe in detail the iterative algorithms which are used to reconstruct the conductivity. The algorithms use the assumption that $a(x) \geq 1$, the known lower bound from equation (39), when computing a_{k+1}, so that a_{k+1} is always in the feasible set. It is not necessary to round off entries in a_{k+1} for convergence, the algorithms work just as well without it, but the reconstruction will be better for low numbers of iterations. The initial guess is usually $a_0(x) = 1$, which should be a qualified guess close to the exact a^*.

The CGM have two ways of computing the gradient g_k. It can be computed using last vector a_{k-1} instead of a_0, which implies that the $\gamma\|a_k - a_{k-1}\|_{L_2}$ term in the Tikhonov functional tends to zero for problems with good convergence, because $\|a_k - a_{k-1}\| \to 0$ as $k \to \infty$ for any constant γ. CGM with gradient $g_k = h_k + \gamma(a_k - a_0)$ is denoted CGM (1), and with $g_k = h_k + \gamma(a_k - a_{k-1})$ denoted CGM (2).

The IRCGM algorithm is just as CGM, except that the regularization parameter γ_k is iteratively updated so that $\gamma_k \to 0$ as $k \to \infty$, forcing the $\gamma_k\|a_k - a_0\|_{L_2}$ term in the functional to zero as $k \to \infty$.

15.1 Fixed point iterations (FPI)

step 0. Given input data of measurements on the boundary (42), choose mesh K_h, partitioned time interval J_τ and initial guess $a_0(x)$ that fulfills the assumptions in (39). Let $k = 0$ and $a_k = a_0$ be the vector representing the discretized conductivity. Compute the sequence of a_k by the following steps.

step 1. Compute the FEM solution U_k using a_k.

step 2. Compute the FEM solution to adjoint problem V_k using a_k and U_k.

step 3. Compute h_k.

step 4. Compute a_{k+1} by:

$$a_{k+1} = a_k - h_k$$

and use the assumption that $a(x) \geq 1$ to round off any entries in the vector a_{k+1} smaller than one.

step 5. Check convergence criteria: if $\|h_h^k\|_{L_2} < \theta$ is satisfied for some chosen tolerance θ, then let $a_h(x) = \eta(x)^T a_{k+1}$, else increment k by one and go to step 1.

15.2 Conjugate gradient method (CGM)

step 0. Given input data of measurements on the boundary (42), choose mesh K_h, partitioned time interval J_τ and initial guess $a_0(x)$ that fulfills the assumptions in (39). Let $k = 0$ and $a_k = a_0$ be the vector representing the discretized conductivity. Choose a constant $\gamma \in (0, 1]$. Compute the sequence of a_k by the following steps.

step 1. Compute the FEM solution U_k using a_k.

step 2. Compute the FEM solution to adjoint problem V_k using a_k and U_k.

step 3. Compute h_k.

step 4. Compute gradient g_k as:

$$g_k = \begin{cases} h_k + \gamma(a_k - a_0) & \text{CGM (1)}, \\ h_k + \gamma(a_k - a_{k-1}) & \text{CGM (2)}, \end{cases}$$

where CGM (1) is standard and CGM (2) is iteratively updated Tikhonov functional, $g_0 = h_0$ (first iteration).

step 5. Compute β_k as:

$$\beta_k = \min\Big\{\frac{(g_k, g_k)}{(g_{k-1}, g_{k-1})}, 1\Big\},$$

where $\beta_0 = 0$ (first iteration).

step 6. Compute the descent direction d_k as:

$$d_k = -g_k + \beta_k d_{k-1},$$

where $d_0 = -g_0$ (first iteration).

step 7. Compute the step-length α_k as:

$$\alpha_k = \min\Big\{-\frac{(g_k, d_k)}{\gamma(d_k, d_k)}, 1\Big\}.$$

step 8. Compute a_{k+1} by:

$$a_{k+1} = a_k + \alpha_k d_k$$

and use the assumption that $a_{k+1} \geq 1$ to round up any entries in the vector smaller than one.

step 9. Check convergence criteria: if $\|g_k\|_{L_2} < \theta$ is satisfied for some chosen tolerance θ, then let $a_h(x) = \eta(x)^T a_{k+1}$, else increment k by one and go to step 1.

15.3 Iteratively regularized conjugate gradient method (IRCGM)

step 0. Given input data of measurements on the boundary (42), choose mesh K_h, partitioned time interval J_τ and initial guess $a_0(x)$ that fulfills the assumptions in (39). Let $k = 0$ and $a_k = a_0$ be the vector representing the discretized conductivity. Choose a constant $\gamma_0 \in (0, 1]$ and $p \in (0, 1]$. Compute the sequence of a_k by the following steps.

step 1. Compute the FEM solution U_k using a_k.

step 2. Compute the FEM solution to adjoint problem V_k using a_k and U_k.

step 3. Compute h_k.

step 4. Compute γ_k and gradient g_k as:

$$\gamma_k = \frac{\gamma_0}{(k+1)^p},$$

$$g_k = h_k + \gamma_k(a_k - a_0),$$

where $g_0 = h_0$ (first iteration).

step 5. Compute β_k as:

$$\beta_k = \min\Big\{\frac{(g_k, g_k)}{(g_{k-1}, g_{k-1})}, 1\Big\},$$

where $\beta_0 = 0$ (first iteration).

step 6. Compute the descent direction d_k as:

$$d_k = -g_k + \beta_k d_{k-1},$$

where $d_0 = -g_0$ (first iteration).

step 7. Compute the step-length α_k as:

$$\alpha_k = \min\Big\{-\frac{(g_k, d_k)}{\gamma_k(d_k, d_k)}, 1\Big\}.$$

step 8. Compute a_{k+1} by:

$$a_{k+1} = a_k + \alpha_k d_k$$

and use the assumption that $a_{k+1} \geq 1$ to round up any entries in the vector smaller than one.

step 9. Check convergence criteria: if $\|g_k\|_{L_2} < \theta$ is satisfied for some chosen tolerance θ, then let $a_h(x) = \eta(x)^T a_{k+1}$, else increment k by one and go to step 1.

16 Numerical studies

The four iterative algorithms are compared by applying them to two case studies. The computer simulations are performed in Matlab on a single processor, using a computer with Intel i5-2500 CPU (3.30 GHz), 64-bit system, RAM 8.00 GB.

The first case uses a coarse mesh and simple piecewise constant conductivity, it takes about 1 minute to perform 1000 iterations. The second case uses a fine mesh and more complicated conductivity, it takes about 30 minutes to perform 1000 iterations. Of interest in the case studies is to compare how good reconstruction it is possible to get when:

- using both backscatter and transmitted data without any noice.
- using only backscatter data and varying the noise levels.

The input data is generated using the same mesh $K_h \times J_\tau$ in space and time, which is a *variational crime*. This means that there is no mismatch between the model used to generate $\tilde{u}$ on the boundary and the model used to reconstruct the conductivity. If $\tilde{u}$ are from a real life measurement of an object, then there is always an mismatch between the two models, which depends on noise added from measuring equipment, mesh K_h, time step τ and small terms that are neglected or considered constant in the model problem described in equation (38).

Since the same mesh is used for generating the measured data with exact conductivity, and reconstructing the conductivity in the inverse problem, there is no difference between the two models, and it is possible to get extremely good results when no noise is added to $\tilde{u}$.

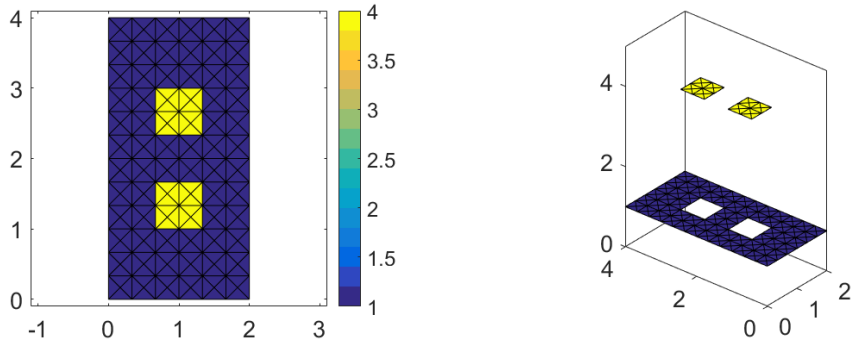
I tried generating the measured data for case 1 using a finer mesh, and a coarser mesh in the minimization problem, but to get good results both meshes have to be extremely fine, which significantly increase the computational times needed.

When noise is added, there is a mismatch between the model used to generate the boundary data and the model used to reconstruct the conductivity. That is, the variational crime is avoided by introducing statistically generated noise to $\tilde{u}$ as described in (43). The chosen probability distribution for the numerical studies is the uniform distribution with values from $[-1, 1]$, amplified by noise levels $\sigma \in \{0.00, 0.03, 0.10, 0.20\}$.

The parameter p used to compute the sequence of regularization parameters in equation (117), is set to one in all the iteratively regularized schemes, to ensure fastest possible convergence. In both cases the exact conductivity $a^* \in A_h$ is completely known and the error norm $\varepsilon_k = \|a_k - a^*\|_{L_2}$ can be computed for each k. You would not know a^* in an real life application, but it is used here to evaluate the efficiency of the iterative algorithms.

17 Case 1

Use a coarse mesh on the square domain $\Omega = [0, 2] \times [0, 4]$ with two enclosures where the exact $a^* = 4$ inside and $a^* = 1$ else. The mesh and exact $a^*(x)$ are shown in figure 4.



(a) Piecesewise constant $a^*(x)$.

(b) 3D view.

Figure 4: The triangulated domain K_h, with a contour plot of $a^*(x)$ on it. Each "square" in the mesh is determined by step lengths $h_x = h_y$, which is divided into four triangle elements.

The parameters used to generate the mesh are:

$$\begin{aligned}
L_x &= 2 && [m], \\
L_y &= 4 && [m], \\
n_x &= 7 && \text{(number of points in x-direction)}, \\
n_y &= 13 && \text{(number of points in y-direction)}, \\
n_n &= 163 && \text{(number of nodes)}, \\
n_{el} &= 288 && \text{(number of elements)}, \\
h_x = h_y &= \frac{L_i}{n_i - 1} = \frac{1}{3} && [m].
\end{aligned} \tag{118}$$

The time dependent parameters used to generate the FEM solutions are:

$$\begin{aligned}
T &= \frac{3}{2} L_x = 3 && [s], \\
f &= \frac{2}{3 h_x} = 2 && \left[\frac{1}{s}\right], \\
t_1 &= \frac{1}{2f} && [s], \\
\tau &= \frac{t_1}{10} && [s], \\
n_t &= \frac{T}{\tau} + 1 = 121 && \text{(number of time nodes)}, \\
p(t) &= \left(6 + \frac{2}{3}\right) \sin(2\pi f t).
\end{aligned} \tag{119}$$

The frequency f is chosen so that the $L_p/h_x = 1.5$, where the wavelength L_p of $p(t)$ is computed by $L_p = c/f$ [m]. The end-time T is chosen to give enough time for a wave, with speed $c = 1$ [m/s], to propagate through the length L_x of the domain and bounce back from the enclosures. This is easy to verify visually once $u_h^*(x, t)$ has been computed. For large enough time T, we have that $u_h^*(x, T)$ is approximately constant.

To get a better convergence, the impulse $p(t)$ is amplified $p(t) = c\sin(2\pi ft)$, so that $u_h^*(x,T) \approx 1$. If $u_h^*(x,T)$ is very small, the gradients are miniscule in the iterative schemes, and if $u_h^*(x,T)$ is very large, the gradients are to large.

The constant is found by firstly computing $u_h^*(x,t)$ with $p(t) = \sin(2\pi ft)$, and estimating $u_h^*(x,t) \approx b$. Then define a new impulse as $p(t) = (1/b)\sin(2\pi ft)$ and compute the FEM solution using the amplified impulse. The shape of the wave remains the same, but now $u_h^*(x,T) \approx 1$.

In the first simulation, the conductivity $a(x)$ is reconstructed using backscatter and transmitted data $\tilde{u}(x,t)$ from boundary Γ_1 and Γ_2, without any noise added. To do so, I run 10 000 iterations of each algorithm, noting the results when $\|g_k\|_{L_2} \leq \theta$ for $\theta = 10^{-2}, 10^{-3}, 10^{-4}, \ldots$ and so on until $k = 10000$.

In the second simulation, the conductivity is reconstructed using only backscatter data with noise level $\sigma \in \{0.00, 0.03, 0.10, 0.20\}$. To do so, I run 10 000 iterations of each algorithm and present the a_k closest to a^*, where k might be smaller than 10 000 if the sequence diverges.

It requires about 10 minutes to compute one 10 000 iterations in Matlab with this coarse mesh. The parameter p used to compute γ_k in IRCGM, is set to one in all the simulations to get the fastest possible convergence rates.

17.1 Results using backscatter and transmitted data without any noise

The results of the simulations can be seen in figures 6 to 10.Error norms versus number of iterations are shown in table 1 and figure 5.

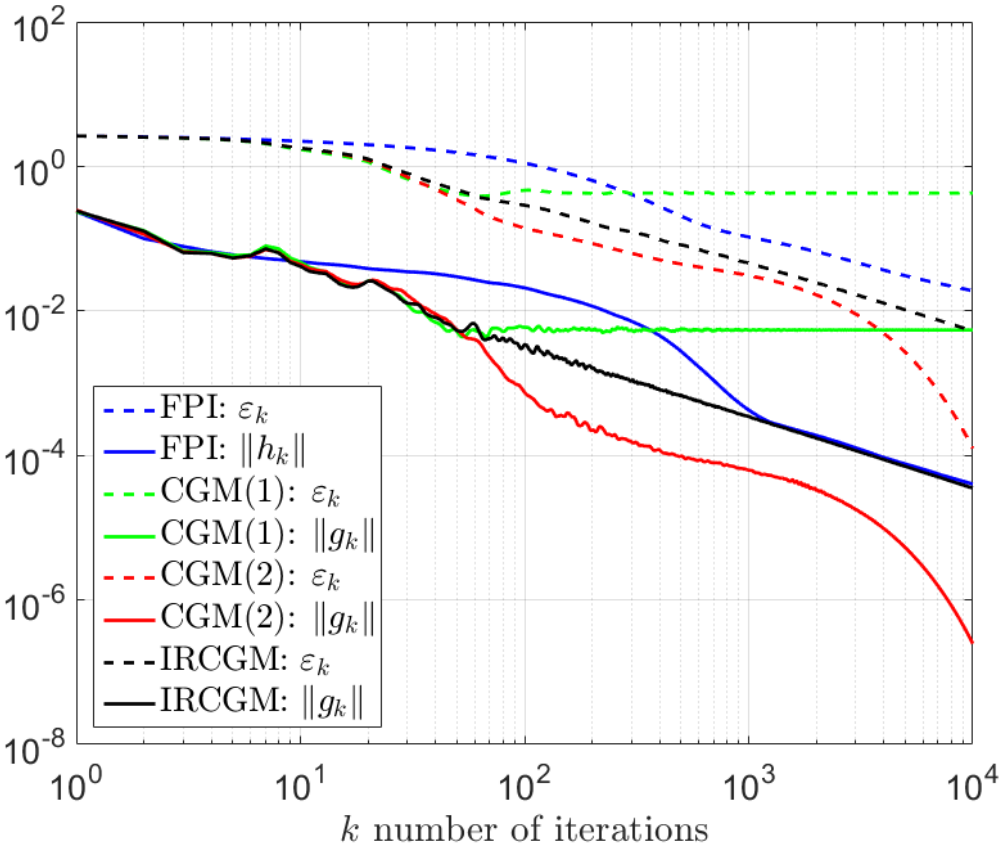


Figure 5: Logarithmic plot of the error norm ε_k and gradient norms $\|h_k\|_{L_2}$, $\|g_k\|_{L_2}$ versus the number of iterations k for the four algorithms. Both backscatter and transmitted data are used without any noise.

The inverse problem has a good convergence in case 1, when both backscatter and transmitted data are used. All algorithms converge to a^* or stabilizes near a^*. It appears that it is possible to reconstruct a to any desired degree of accuracy using FPI, CGM (2) or IRCGM, with CGM (2) being the fastest.

CGM (1) is the only iterative method that does not converge to a^*, but to a point near a^*. This is because of the constant regularization term $\gamma\|a_k - a_0\|^2_{L_2}/2$ which limits the reconstruction to a neighbourhood near a_0. The gradient and error norm stabilizes around 100 iterations for CGM (1), and the algorithm is not able to compute any better reconstructions, meaning that a_{100} and a_{10000} are largely the same. The regularization parameter γ determines how close the reconstructed conductivity field can get to the exact, and the number of iterations needed before the norms stabilizes. As $\gamma \to 0$, the norms tend to zero. For CGM (1), $\gamma = 0.001$ is chosen smaller than for the other algorithms, so that the norm of g_k is able to become smaller than 10^{-2}. If γ is set even lower, the curves for CGM (1) in figure 5 will follow those of IRCGM until the norms stabilizes. The reconstructions, in figure 7, are however adequate to make predictions of the shape and numerical values of the exact a^*. The enclosures where $a^* > 1$ are clearly outlined, with an average around 3.5.

The simplest algorithm FPI gives good results, clearly $a_k \to a^*$ as $k \to \infty$. FPI is slow but steady, compared to the conjugate gradient methods. This is because FPI has no line-search, where the optimal step-size α_k is computed, nor any fraction of the previous direction added to the gradient. After around 300 iterations, the reconstruction is better then that of CGM (1), but worse then that of CGM (2) and IRCGM.

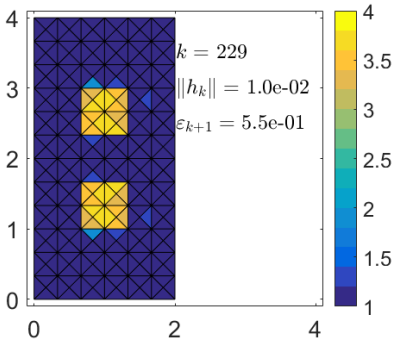
When k is large, the value and convergence rate of the gradient norm are comparable that of IRCGM, and the error norm has about the same convergence rate as IRCGM. The reconstructions, in figure 6, are very good and almost exact for large k:s. The enclosure where $a^* > 1$ is clearly visible in all the subfigures. Around 700 iterations is needed to make a qualified guess of the numerical value inside the enclosure.

Method	θ	k	$\|h_k\|_{L_2}$	ε_{k+1}	figure
FPI	10^{-2}	229	$1.0 \cdot 10^{-2}$	$5.5 \cdot 10^{-1}$	6a
	10^{-3}	720	$1.0 \cdot 10^{-3}$	$1.3 \cdot 10^{-1}$	6c
	10^{-4}	3 767	$1.0 \cdot 10^{-4}$	$3.9 \cdot 10^{-2}$	6e
	10^{-5}	10 000	$4.0 \cdot 10^{-5}$	$1.9 \cdot 10^{-2}$	6g
Method	θ	k	$\|g_k\|_{L_2}$	ε_{k+1}	figure
CGM (1), $\gamma = 0.001$	10^{-2}	33	$9.6 \cdot 10^{-3}$	$5.9 \cdot 10^{-1}$	7a
	10^{-3}	2 435	$5.4 \cdot 10^{-3}$	$4.3 \cdot 10^{-1}$	7c
	10^{-3}	10 000	$5.4 \cdot 10^{-3}$	$4.3 \cdot 10^{-1}$	7e
CGM (2), $\gamma = 0.05$	10^{-2}	38	$9.9 \cdot 10^{-3}$	$5.0 \cdot 10^{-1}$	8a
	10^{-3}	90	$1.0 \cdot 10^{-3}$	$1.5 \cdot 10^{-1}$	8c
	10^{-4}	473	$1.0 \cdot 10^{-4}$	$4.6 \cdot 10^{-2}$	8e
	10^{-5}	3 957	$1.0 \cdot 10^{-5}$	$5.0 \cdot 10^{-3}$	9a
	10^{-6}	7 723	$1.0 \cdot 10^{-6}$	$5.0 \cdot 10^{-4}$	9c
	10^{-7}	10 000	$2.5 \cdot 10^{-7}$	$1.3 \cdot 10^{-4}$	9e
IRCGM, $\gamma_0 = 0.05$	10^{-2}	36	$9.4 \cdot 10^{-3}$	$6.3 \cdot 10^{-1}$	10a
	10^{-3}	320	$9.9 \cdot 10^{-4}$	$1.2 \cdot 10^{-1}$	10c
	10^{-4}	3 493	$1.0 \cdot 10^{-4}$	$1.5 \cdot 10^{-2}$	10e
	10^{-5}	10 000	$3.5 \cdot 10^{-5}$	$5.3 \cdot 10^{-3}$	10g

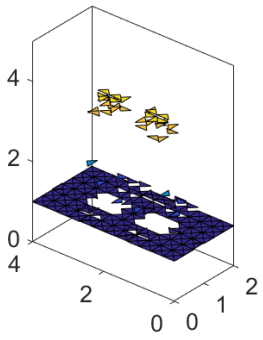
Table 1: The stop data for the simulations, α_k and β_k are not shown. CGM (1) uses $g_k = h_k + \gamma(a_k - a_0)$ and CGM (2) uses $g_k = h_k + \gamma(a_k - a_{k-1})$.

IRCGM converges linearly to a^* when k is large. It gives extremely good reconstructions, as can be seen in figure 10, and you need around 300 iterations to make a qualified guess of the numerical values inside the enclosures. When comparing to CGM (2), the shape of the reconstruction from IRCGM looks better for the same number of iterations, but the error norm is slightly higher.

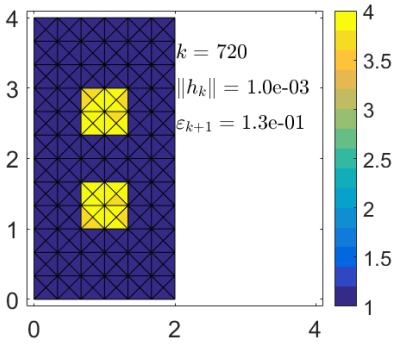
CGM (2) gives the best reconstructions in the least number of iterations in this case. Around 100-500 iterations is necessary to be able to make good predictions of the conductivity inside the enclosures.



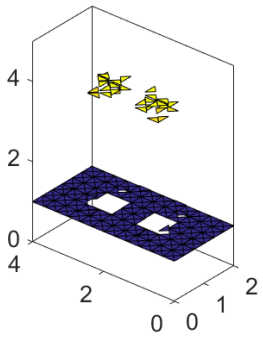
(a) Stopping criteria $\theta = 10^{-2}$.



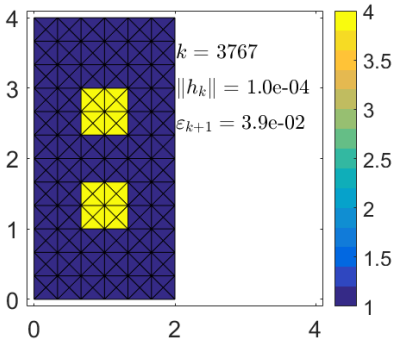
(b) 3D view.



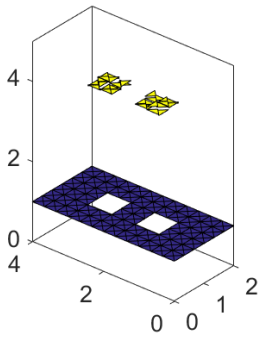
(c) Stopping criteria $\theta = 10^{-3}$.



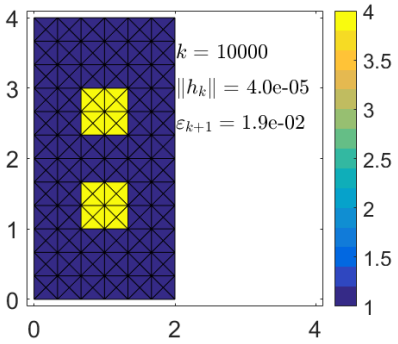
(d) 3D view.



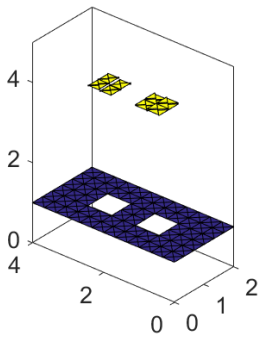
(e) Stopping criteria $\theta = 10^{-4}$.



(f) 3D view.

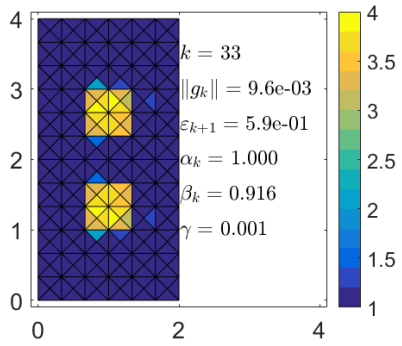


(g) Stopping criteria $\theta = 10^{-5}$.

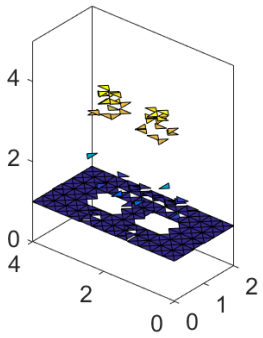


(h) 3D view.

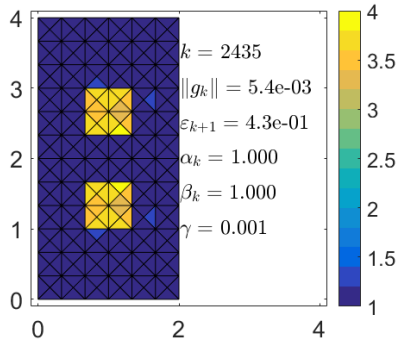
Figure 6: Reconstructed conductivity $a(x)$ for case 1, using **FPI**, both backscatter and transmitted data without any noise.



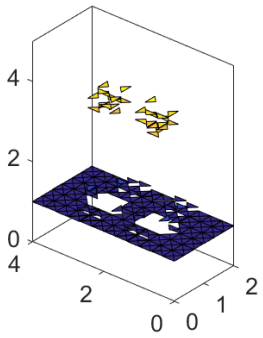
(a) Stopping criteria $\theta = 10^{-2}$.



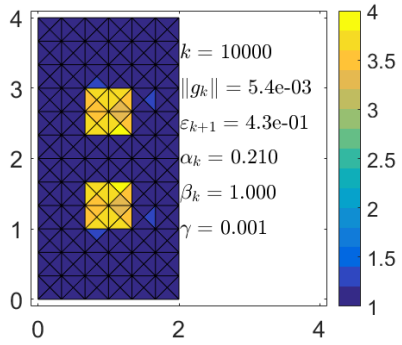
(b) 3D view.



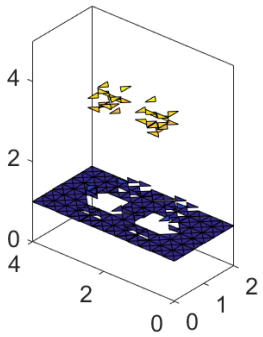
(c) Stopping criteria $\theta = 10^{-3}$.



(d) 3D view.

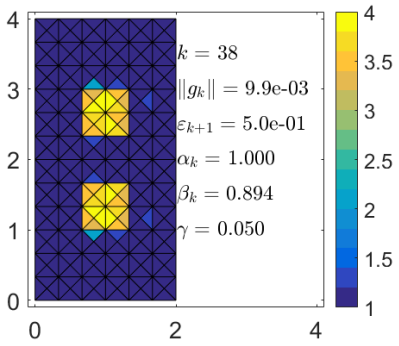


(e) Stopping criteria $\theta = 10^{-3}$.

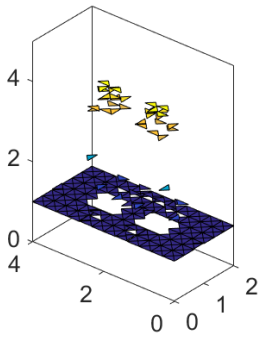


(f) 3D view.

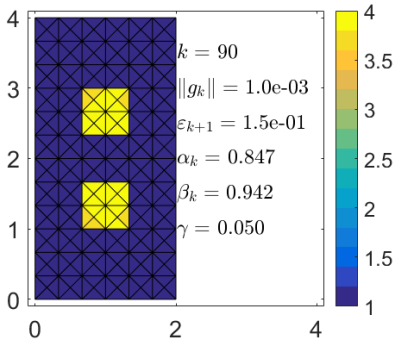
Figure 7: Reconstructed conductivity $a(x)$ for case 1, using **CGM (1)** with $g_k = h_k + \gamma(a_k - a_0)$, both backscatter and transmitted data without any noise.



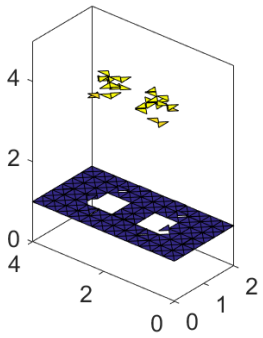
(a) Stopping criteria $\theta = 10^{-2}$.



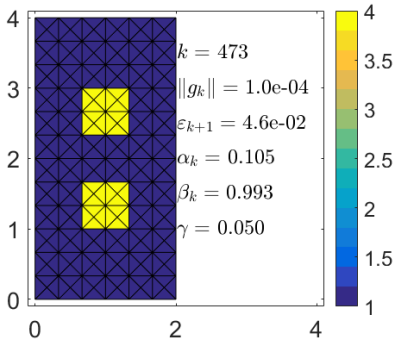
(b) 3D view.



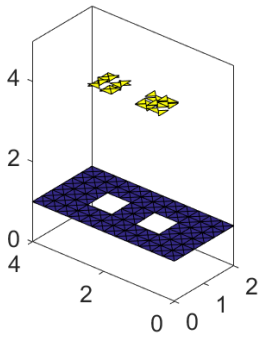
(c) Stopping criteria $\theta = 10^{-3}$.



(d) 3D view.

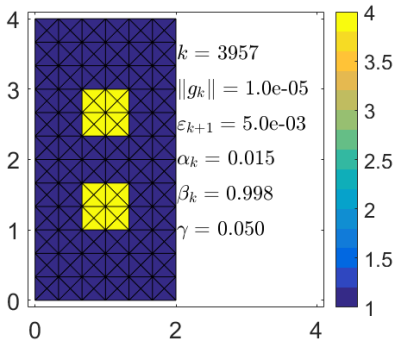


(e) Stopping criteria $\theta = 10^{-4}$.

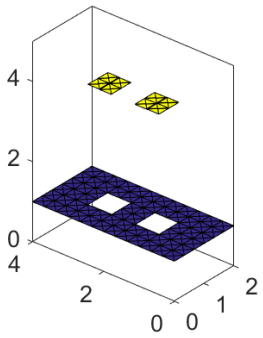


(f) 3D view.

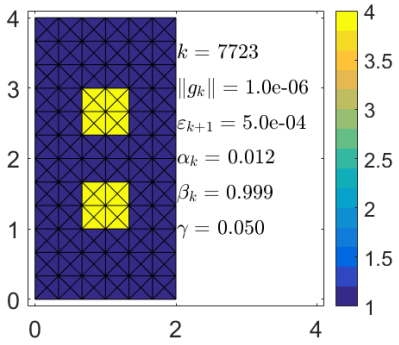
Figure 8: Reconstructed conductivity $a(x)$ for case 1, using **CGM (2)** with $g_k = h_k + \gamma(a_k - a_{k-1})$, both backscatter and transmitted data without any noise.



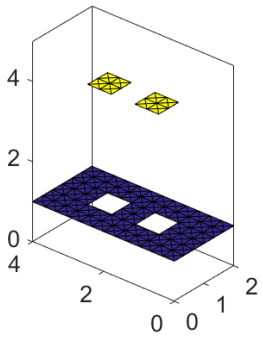
(a) Stopping criteria $\theta = 10^{-5}$.



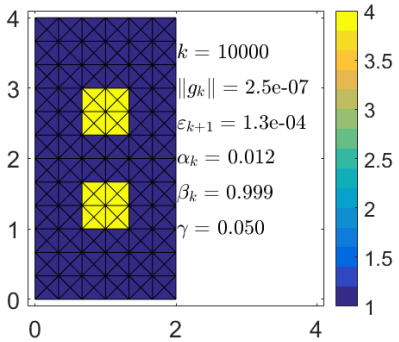
(b) 3D view.



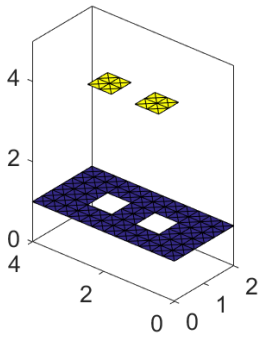
(c) Stopping criteria $\theta = 10^{-6}$.



(d) 3D view.

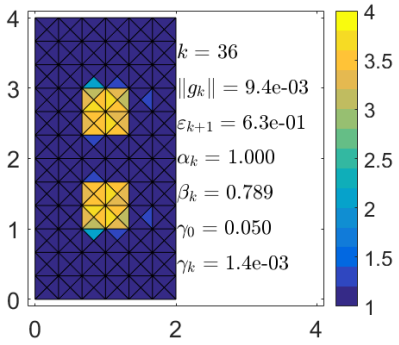


(e) Stopping criteria $\theta = 10^{-7}$.

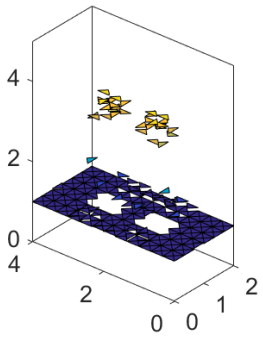


(f) 3D view.

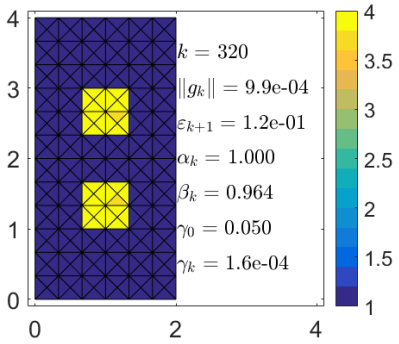
Figure 9: Reconstructed conductivity $a(x)$ for case 1, using **CGM (2)** with $g_k = h_k + \gamma(a_k - a_{k-1})$, both backscatter and transmitted data without any noise.



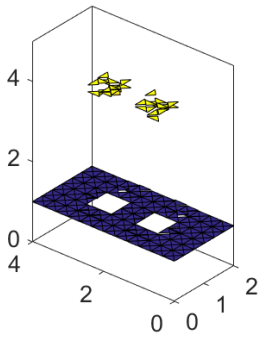
(a) Stopping criteria $\theta = 10^{-2}$.



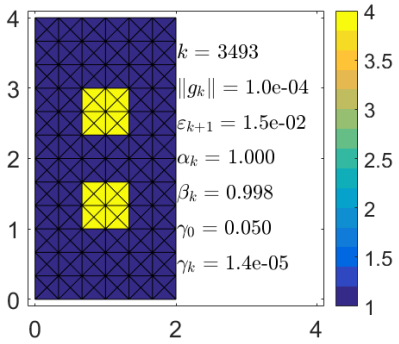
(b) 3D view.



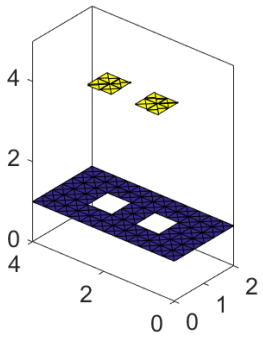
(c) Stopping criteria $\theta = 10^{-3}$.



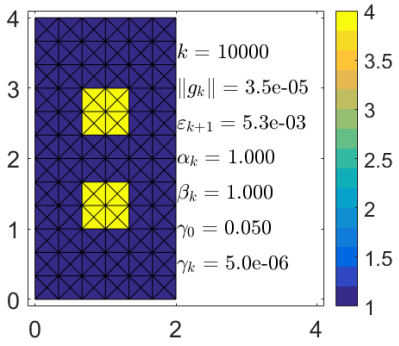
(d) 3D view.



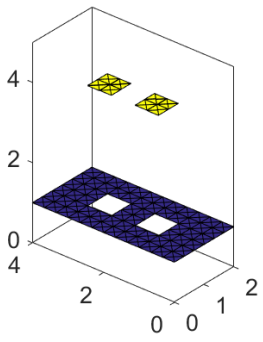
(e) Stopping criteria $\theta = 10^{-4}$.



(f) 3D view.



(g) Stopping criteria $\theta = 10^{-5}$.

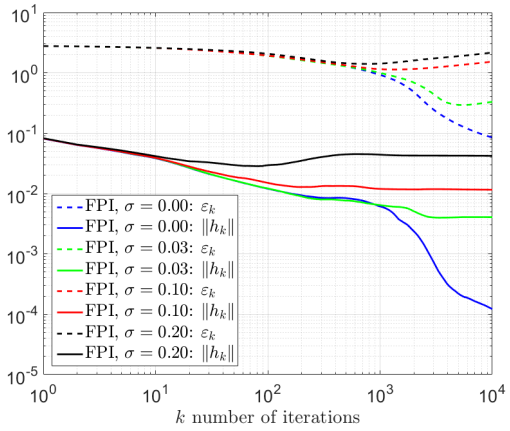


(h) 3D view.

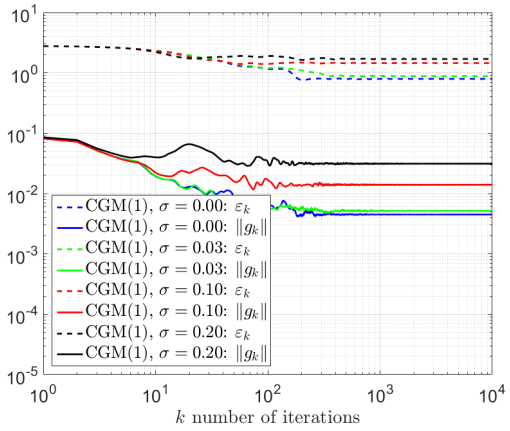
Figure 10: Reconstructed conductivity $a(x)$ for case 1, using **IRCGM**, both backscatter and transmitted data without any noise.

17.2 Results using only backscatter data, varying noise level

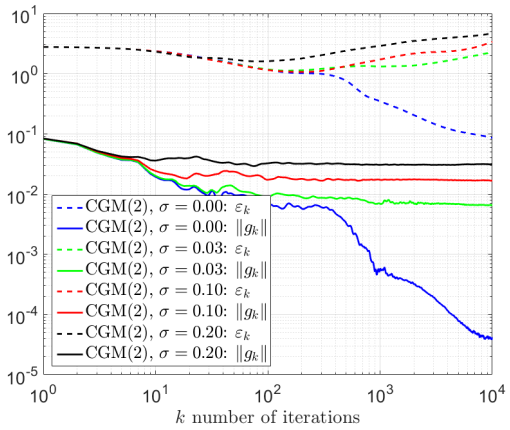
The results of the simulations can be seen in figures 12 to 15. Error norms versus number of iterations are shown in table 2 and figure 11.



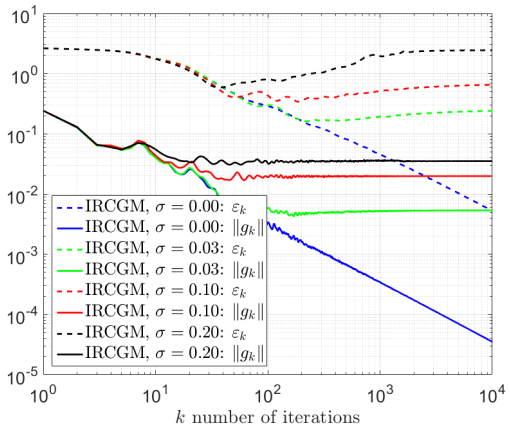
(a) FPI.



(b) CGM (1).



(c) CGM (2).



(d) IRCGM.

Figure 11: The error norm ε_k and gradient norm $\|h_k\|_{L_2}$, $\|g_k\|_{L_2}$ versus number of iterations k for the four algorithms. Only backscatter data is used, noise level σ is varied.

The inverse problem has good convergence in case 1, when only backscatter data is used and the noise level is zero. For non-zero noise levels FPI, CGM (2) and IRCGM diverge after some number of iterations, indicating that the inverse problem is ill-posed as the solutions become numerically unstable when noise is introduced. Clearly the reconstructions get worse as more noise is added, and the algorithms diverge or stabilizes for lower number of iterations as the noise level is increased.

For FPI, the best possible reconstruction, using maximum 10 000 iterations, is shown in figure 12. With no noise, the algorithm is able to reconstruct a very satisfactory. The result is comparable to that of using both backscatter and transmitted data. When $\sigma \neq 0$, the solution diverges after some number of iterations. However, the gradient norm stabilizes before the error norm diverges. The reconstruction for $\sigma = 0.03$ is quite good compared to those of $\sigma = 0.10, 0.20$. It is possible to predict the shape of the enclosures for all the noise levels simulated. One way to do so, in a post process, is to let all the elements with $a < 2$ be set to 1, and take the mean value of the enclosures that appear.

CGM (1) never diverges. The error norm stabilizes when the gradient norm stabilizes, which is one of the good properties of CGM (1). Figure 13 shows the stabilized reconstruction a_{10000} for all noise levels.

Clearly the reconstruction becomes worse near the boundary Γ_2, in all noise levels, when compared to the previous simulation for CGM (1). The shape of the enclosures can be estimated in all noise levels. Using $\sigma = 0.00, 0.03$ gives comparable results.

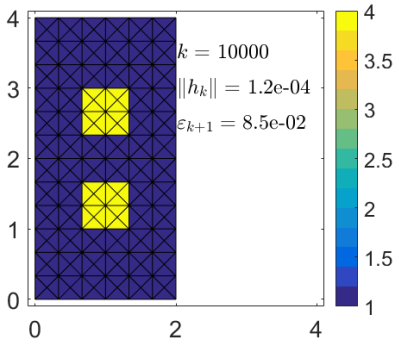
Method	σ	k	$\|h_k\|_{L_2}$	ε_{k+1}	figure
FPI	0.00	10 000	$1.2 \cdot 10^{-4}$	$8.5 \cdot 10^{-2}$	12a
	0.03	5 425	$4.1 \cdot 10^{-3}$	$2.9 \cdot 10^{-1}$	12c
	0.10	1 327	$1.2 \cdot 10^{-2}$	$1.1 \cdot 10^{0}$	12e
	0.20	748	$4.5 \cdot 10^{-2}$	$1.4 \cdot 10^{0}$	12g
Method	σ	k	$\|g_k\|_{L_2}$	ε_{k+1}	figure
CGM (1), $\gamma = 0.001$	0.00	10 000	$4.5 \cdot 10^{-3}$	$7.9 \cdot 10^{-1}$	13a
	0.03	10 000	$5.2 \cdot 10^{-3}$	$8.8 \cdot 10^{-1}$	13c
	0.10	10 000	$1.4 \cdot 10^{-2}$	$1.4 \cdot 10^{0}$	13e
	0.20	10 000	$3.1 \cdot 10^{-2}$	$1.7 \cdot 10^{0}$	13g
CGM (2), $\gamma = 0.05$	0.00	10 000	$4.1 \cdot 10^{-5}$	$8.6 \cdot 10^{-2}$	14a
	0.03	146	$9.6 \cdot 10^{-3}$	$1.1 \cdot 10^{0}$	14c
	0.10	202	$1.8 \cdot 10^{-2}$	$1.1 \cdot 10^{0}$	14e
	0.20	82	$3.0 \cdot 10^{-2}$	$1.6 \cdot 10^{0}$	14g
IRCGM, $\gamma_0 = 0.05$	0.00	10 000	$3.5 \cdot 10^{-5}$	$5.3 \cdot 10^{-3}$	15a
	0.03	222	$4.9 \cdot 10^{-3}$	$1.6 \cdot 10^{-1}$	15c
	0.10	186	$2.0 \cdot 10^{-2}$	$3.4 \cdot 10^{-1}$	15e
	0.20	37	$3.5 \cdot 10^{-2}$	$5.9 \cdot 10^{-1}$	15g

Table 2: Stop data for lowest possible error norm ε_k for the different noise levels σ and algorithms. CGM (1) uses $g_k = h_k + \gamma(a_k - a_0)$ and CGM (2) uses $g_k = h_k + \gamma(a_k - a_{k-1})$.

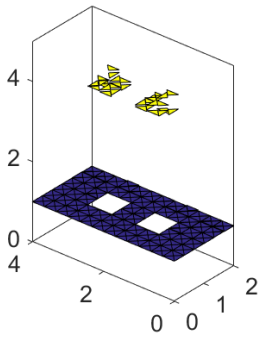
CGM (2) gets very good results when $\sigma = 0$, but quickly diverges for non-zero noise levels. It appears to be the least suitable algorithm in the case, compared to the other algorithms, because it gets the highest error norm for low noise levels.

The reconstructions for IRCGM in figure 15 are extremely good and clearly gives the best solutions in this case. The solution is almost perfect for $\sigma = 0.00, 0.03$ and adequate for $\sigma = 0.10, 0.20$. The elements near boundary Γ_2 are close to 1 and the enclosures are easily made out in all noise levels, which is something that the other algorithms lack. However, the error norm diverges instead of stabilizing as the CGM (1) does.

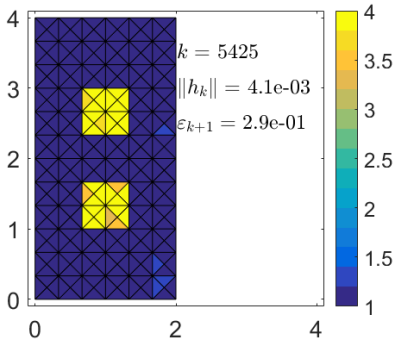
For both CGM (2) and IRCGM, it is critical that the convergence criteria in the algorithms can determine when the gradient norm have stabilized, so that the computing sequence is broken before the error norm increases to much. Also, the divergence point can be raised by increasing γ and γ_0, meaning that the algorithms need more iterations to produce the same results, but are more stable. For IRCGM the parameter $p \in (0, 1]$ is always set to one in the simulations, choosing a smaller p so that $\gamma_k \to 0$ more slowly will also increase the divergence point.



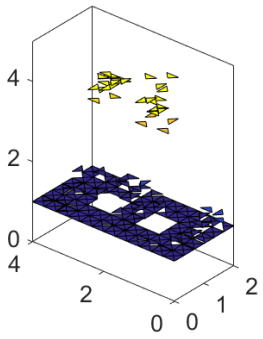
(a) $\sigma = 0.00$, $k = 10000$.



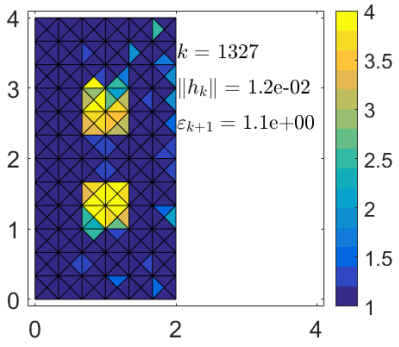
(b) 3D view.



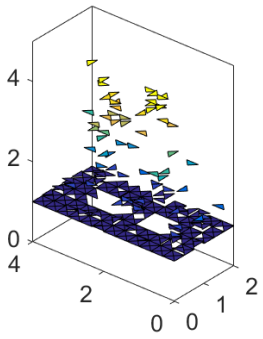
(c) $\sigma = 0.03$, $k = 5425$.



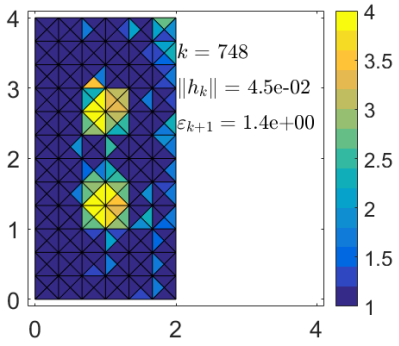
(d) 3D view.



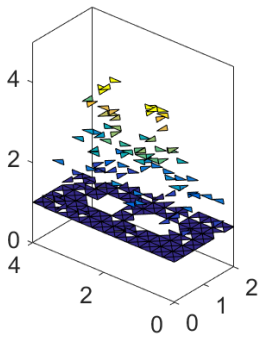
(e) $\sigma = 0.10$, $k = 1327$.



(f) 3D view.

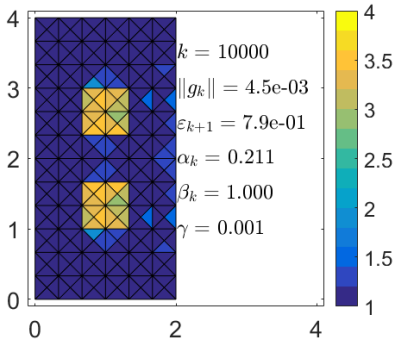


(g) $\sigma = 0.20$, $k = 748$.

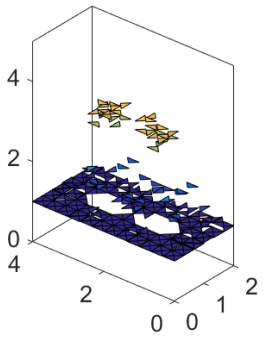


(h) 3D view.

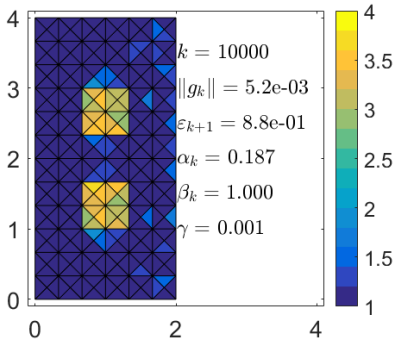
Figure 12: Best reconstructed conductivity $a(x)$ for case 1, using **FPI**, only backscatter data and varying noise level σ.



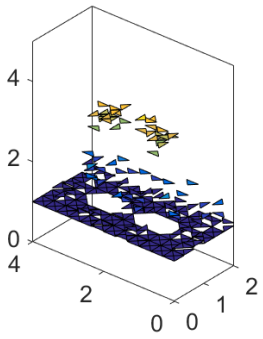
(a) $\sigma = 0.00$, $k = 10000$.



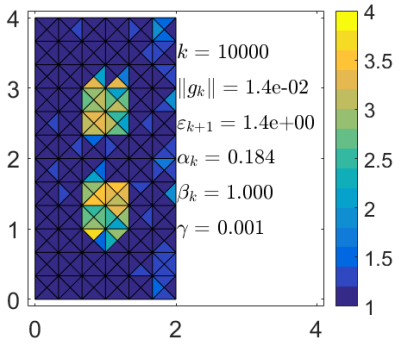
(b) 3D view.



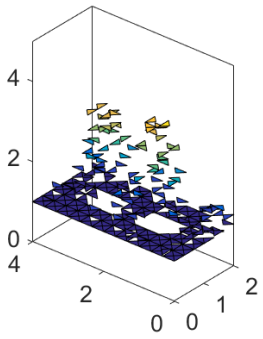
(c) $\sigma = 0.03$, $k = 10000$.



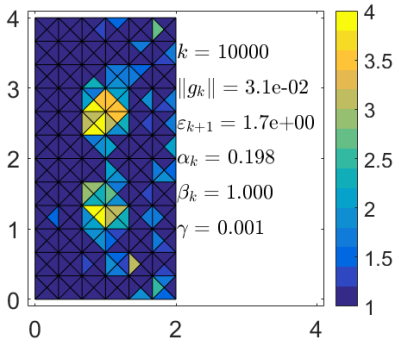
(d) 3D view.



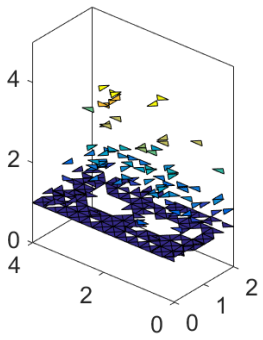
(e) $\sigma = 0.10$, $k = 10000$.



(f) 3D view.

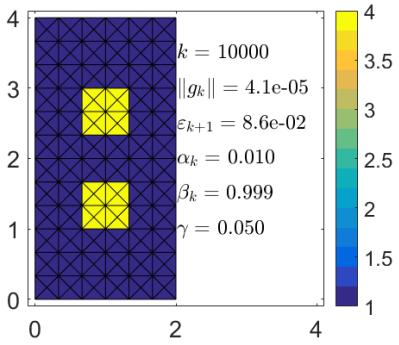


(g) $\sigma = 0.20$, $k = 10000$.

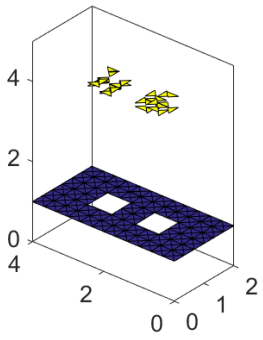


(h) 3D view.

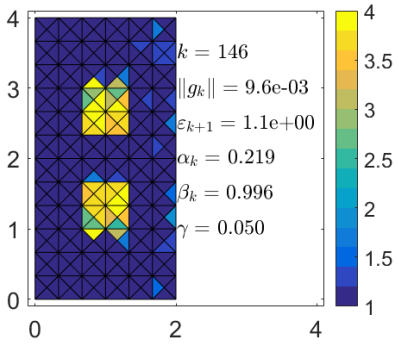
Figure 13: Best reconstructed conductivity $a(x)$ for case 1, using **CGM (1)**, only backscatter data and varying noise level σ.



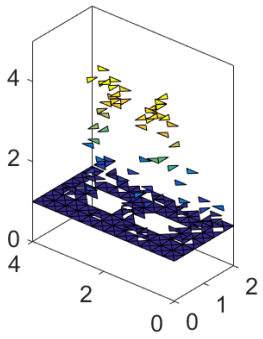
(a) $\sigma = 0.00$, $k = 10000$.



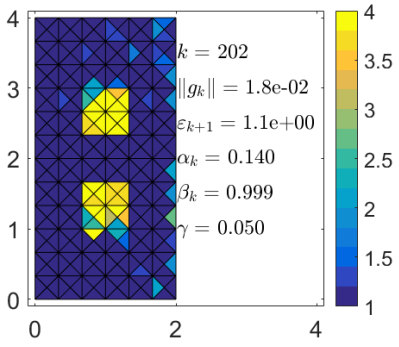
(b) 3D view.



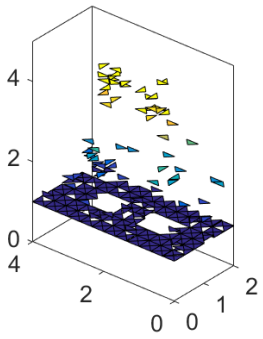
(c) $\sigma = 0.03$, $k = 146$.



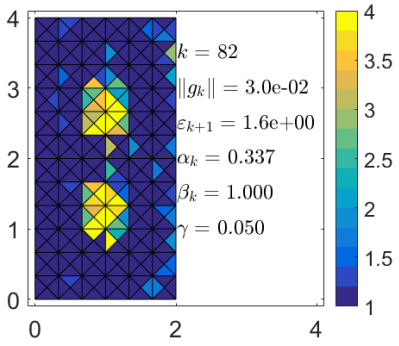
(d) 3D view.



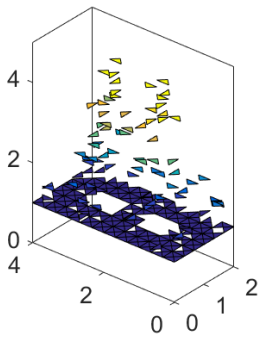
(e) $\sigma = 0.10$, $k = 202$.



(f) 3D view.

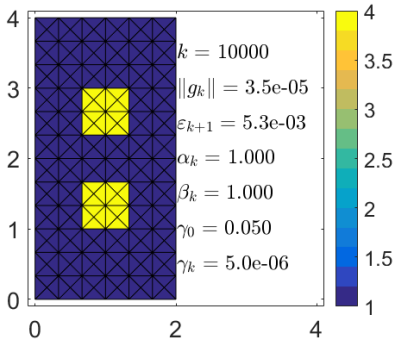


(g) $\sigma = 0.20$, $k = 82$.

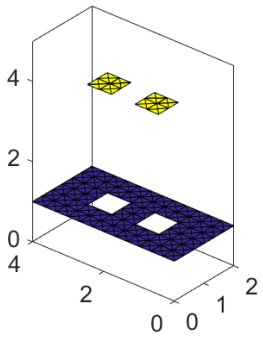


(h) 3D view.

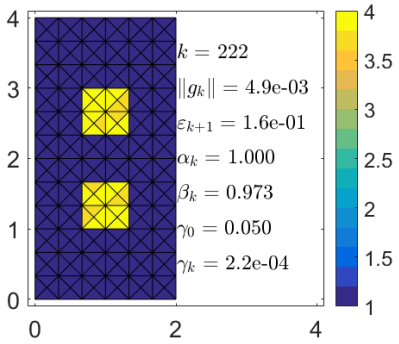
Figure 14: Best reconstructed conductivity $a(x)$ for case 1, using **CGM (2)**, only backscatter data and varying noise level σ.



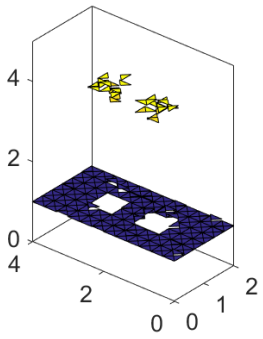
(a) $\sigma = 0.00$, $k = 10000$.



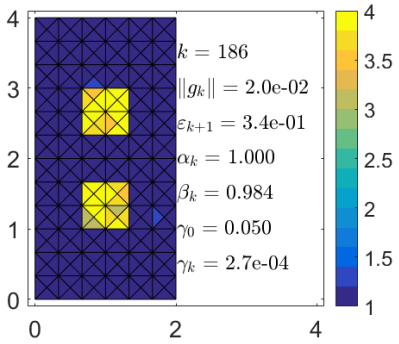
(b) 3D view.



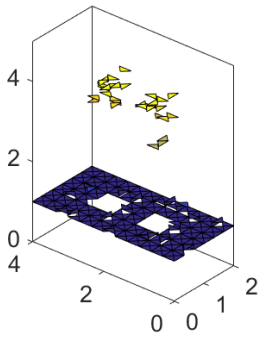
(c) $\sigma = 0.03$, $k = 222$.



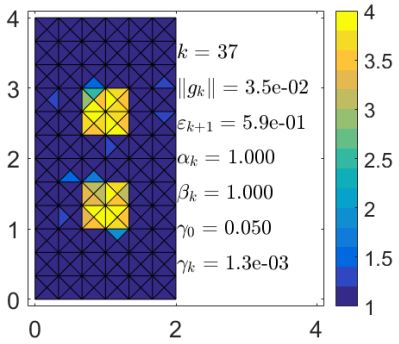
(d) 3D view.



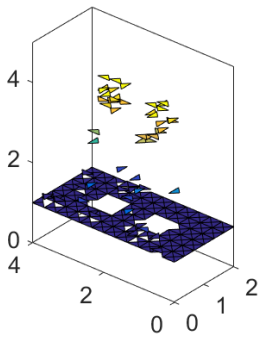
(e) $\sigma = 0.10$, $k = 186$.



(f) 3D view.



(g) $\sigma = 0.20$, $k = 37$.



(h) 3D view.

Figure 15: Best reconstructed conductivity $a(x)$ for case 1, using **IRCGM**, only backscatter data and varying noise level σ.

18 Case 2

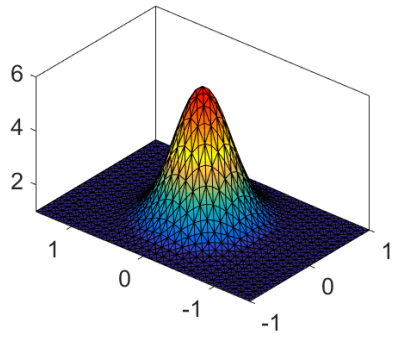


Figure 16: Exact $a^*(x, y)$ plotted in 3D over Ω.

Compare the results of the algorithms using a fine mesh with a exact conductivity function:

$$a^*(x, y) = 1 + 5 \cdot e^{-5(x^2+y^2)}. \tag{120}$$

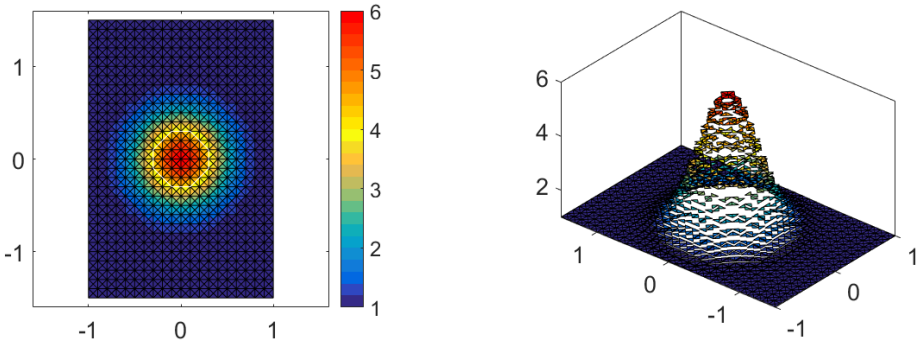
Let $\Omega = [-1, 1] \times [-1.5, 1.5]$ and use step lengths $h_x = h_y = 0.1$ in the triangulation of Ω, which means that the mesh has $n_x = 21$, $n_y = 31$, $n_n = 1251$ number of spatial nodes and $n_{el} = 2400$ number of spatial elements. The exact a^* is projected into the space of piecewise constant of functions by:

$$a_i = \frac{1}{\text{area}(K_i)} \iint_{K_i} a^*(x, y)\,\mathrm{d}x\mathrm{d}y \quad \forall i \in \{1, 2, \ldots, n_{el}\}. \tag{121}$$

When the elements are so small that a^* is approximately linear in each element, we get the simpler approximation:

$$a_i = \frac{1}{3}\Big(a^*(x_1, y_1) + a^*(x_2, y_2) + a^*(x_3, y_3)\Big) \quad \forall i \in \{1, 2, \ldots, n_{el}\}, \tag{122}$$

where (x_j, y_j), $j \in \{1, 2, 3\}$ denotes the corner nodes that defines each triangle element.



(a) Piecesewise constant $a^*(x)$.

(b) 3D view.

Figure 17: The triangulated domain K_h, with a contour plot of $a^*(x, y) \in A_h$ on it. Each "square" in the mesh is divided into four triangle elements. The white line in (a) is the contour where $a^* = 4$.

The time dependent parameters used to compute FEM solutions are:

$$\begin{aligned} T &= 2L_x = 4 && [s], \\ f &= \frac{2}{3h_x} = 6 + \frac{2}{3} && \left[\frac{1}{s}\right], \\ t_1 &= \frac{1}{2f} = \frac{3}{40} = 0.075 && [s], \\ \tau &= \frac{t_1}{10} && [s], \\ n_t &= \frac{T}{\tau} + 1 \approx 534 && \text{(number of time nodes)}, \\ p(t) &= 20\sin(2\pi f t). \end{aligned} \tag{123}$$

As for case 1, frequency f is chosen so that the wavelength of $p(t)$ divided by step length h_x equals 1.5. We have that angular frequency $\omega = 2\pi f \approx 41.9$ [rad/s]. The end time T is chosen so that $u_h^*(x,T) \approx b$ where b is some real constant, and $p(t)$ is amplified so that the constant $b \approx 1$.

Each simulation requires about 0.5 h of computational time to perform 1000 iterations. The gradient norm and error norm are saved and used to find the number of iterations k_{min} needed to minimize the error norm, which may be smaller than 1000 if the sequence diverges, then the simulation is run one more time with $k_{min} < 1000$ as maximum number of iterations to reconstruct the best possible conductivity field.

In the first simulation, the conductivity $a(x)$ is reconstructed using backscatter and transmitted data $\tilde{u}(x,t)$ from boundary Γ_1 and Γ_2, without any noise added.

In the second simulation, the conductivity is reconstructed using only backscatter data with noise level $\sigma \in \{0.00, 0.03, 0.10, 0.20\}$.

The parameter p used to compute γ_k in IRCGM, is set to one in all the simulations to get the fastest possible convergence rates.

18.1 Results using backscatter and transmitted data without any noise

The results of the simulations can be seen in figure 19. Error norms versus number of iterations are shown in table 3 and figure 5.

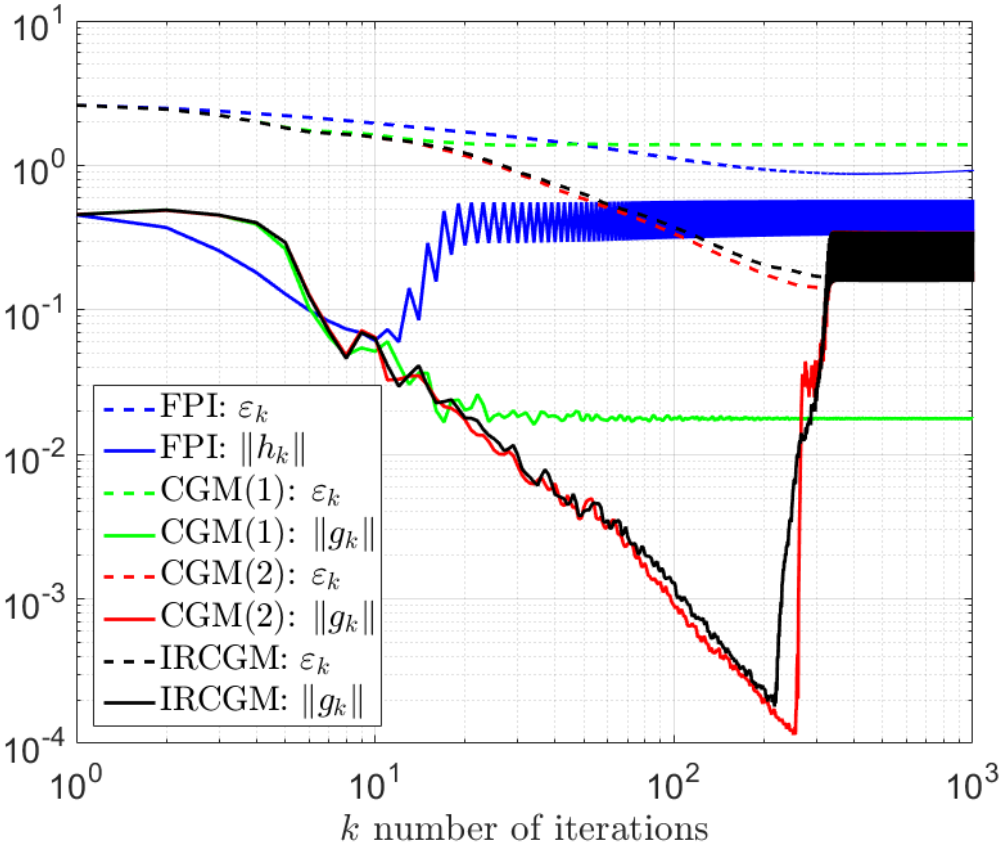


Figure 18: Logarithmic plot of the error norm ε_k and gradient norms $\|h_k\|_{L_2}$, $\|g_k\|_{L_2}$ versus the number of iterations k for the four algorithms. Both backscatter and transmitted data is used without any noise.

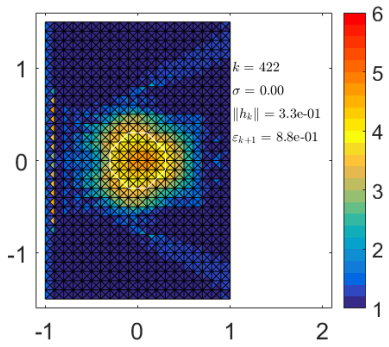
Method	k	$\|h_k\|_{L_2}$	ε_{k+1}	figure
FPI	422	$3.3 \cdot 10^{-1}$	$8.8 \cdot 10^{-1}$	19a
Method	k	$\|g_k\|_{L_2}$	ε_{k+1}	figure
CGM (1), $\gamma = 0.01$	1 000	$1.8 \cdot 10^{-2}$	$1.4 \cdot 10^{0}$	19c
CGM (2), $\gamma = 0.01$	320	$5.3 \cdot 10^{-2}$	$1.4 \cdot 10^{-1}$	19e
IRCGM, $\gamma_0 = 0.01$	315	$5.0 \cdot 10^{-2}$	$1.7 \cdot 10^{-1}$	19g

Table 3: Stop data for lowest possible error norm ε_k of the algorithms, using both backscatter and transmitted data without any noise. CGM (1) uses $g_k = h_k + \gamma(a_k - a_0)$ and CGM (2) uses $g_k = h_k + \gamma(a_k - a_{k-1})$.

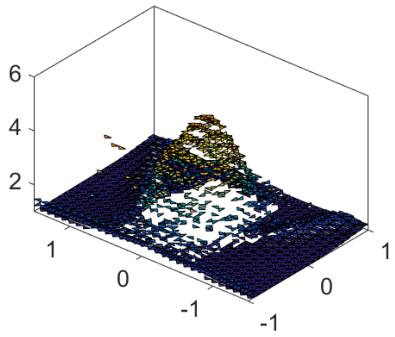
The inverse problem has quite bad convergence in the more complex case 2. The error norm diverges or stabilizes in all the algorithms, even though both backscatter and transmitted data without any noise is used. The regularization parameter is chosen so that $\gamma = \gamma_0 = 0.01$ in the conjugate gradient methods, so that they are more comparable. This means that the reconstructions from CGM (1) are quite bad, which needs a smaller γ to enable smaller norms, and that the CGM (2) and IRCGM both converges and diverges fast.

The best possible reconstructions are shown in figure 19, where the white line in the contour plots shows the boundary where $a^* = 4$.

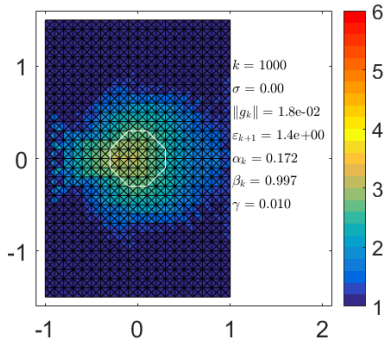
The FPI algorithm is able to produce a reconstruction where the circular shape of a^* is somewhat visible, where the maximum value of a is a little above 4. The CGM (1) reconstruction is quite bad and can not be used for any good predictions. Of course, it can be expected that the reconstruction gets better with lower γ. The CGM (2) and IRCGM gets almost the same results in the almost the same number of iterations. The shape looks very much like a^* and the numerical values are very close to the exact values.



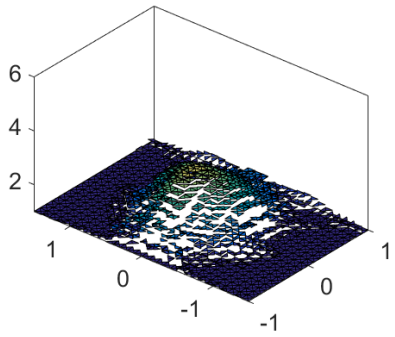
(a) FPI.



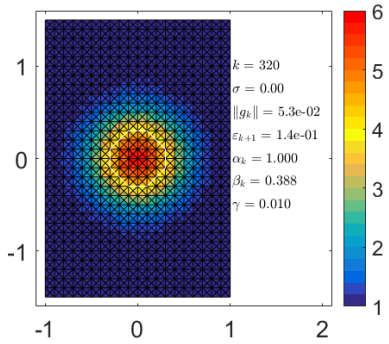
(b) 3D view.



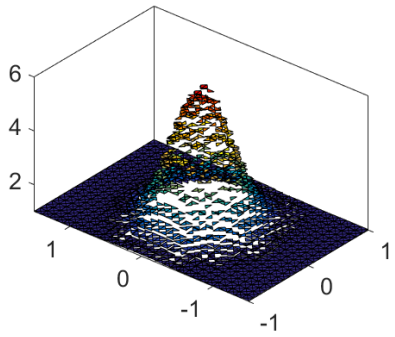
(c) CGM (1).



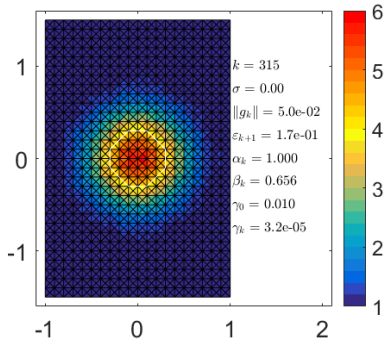
(d) 3D view.



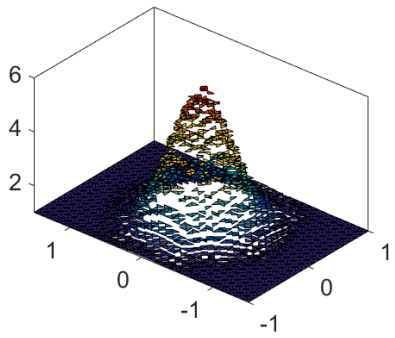
(e) CGM (2).



(f) 3D view.



(g) IRCGM.

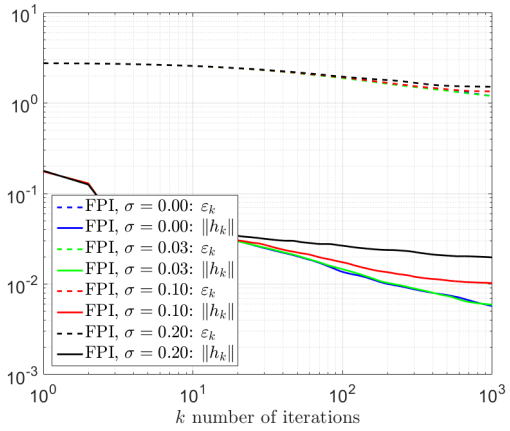


(h) 3D view.

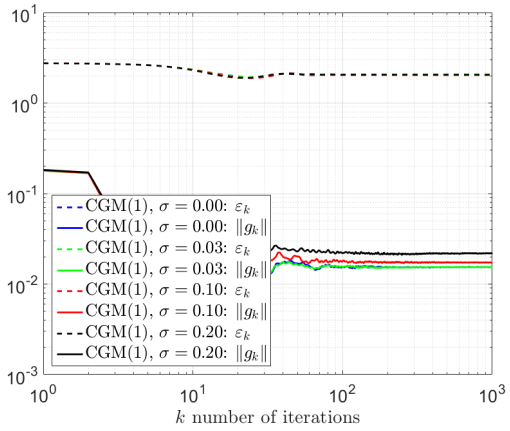
Figure 19: Best reconstructed conductivity $a(x)$ for case 2, using both backscatter and transmitted data without any noise.

18.2 Results using only backscatter data, varying noise level

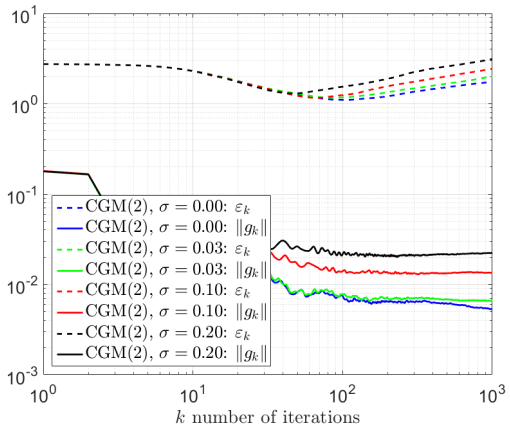
The results of the simulations can be seen in figures 21 to 24. Error norms versus number of iterations are shown in table 4 and figure 20.



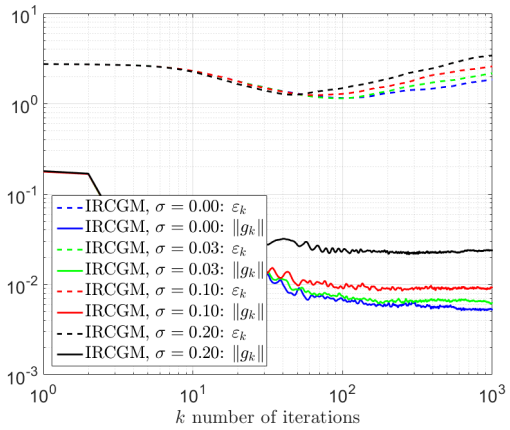
(a) FPI.



(b) CGM (1).



(c) CGM (2).



(d) IRCGM.

Figure 20: The error norm ε_k and gradient norm $\|h_k\|_{L_2}$, $\|g_k\|_{L_2}$ versus number of iterations k for the four algorithms. Only backscatter data is used, noise level σ is varied.

The error norm diverges or stabilizes after some number of iterations for all the algorithms, when only backscatter data is used with varying noise level.

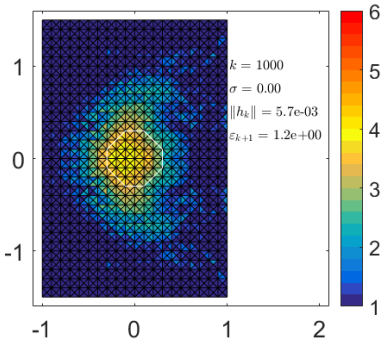
The error norms for FPI diverges for $k > 1000$. As before, better reconstructions are attainable with low noise levels, but the difference is quite small. The shape of the reconstruction is about the same for $\sigma = 0.00, 0.03$, but the center of the shape is not in the origin.

The CGM (1) gets very bad reconstructions. The shape is spread out and off center, which clearly is the result from the missing information of the transmitted data. However, the reconstructions are largely the same for all noise levels. The error norm for CGM (1) stabilizes around 2.05 which is rounded to one decimal point in the figures.

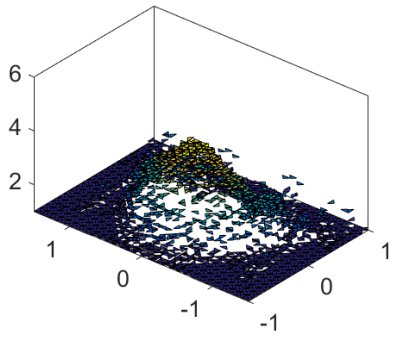
Method	σ	k	$\|h_k\|_{L_2}$	ε_{k+1}	figure
FPI	0.00	1 000	$5.7 \cdot 10^{-3}$	1.2	21a
	0.03	1 000	$5.9 \cdot 10^{-3}$	1.2	21c
	0.10	1 000	$1.0 \cdot 10^{-2}$	1.3	21e
	0.20	1 000	$2.0 \cdot 10^{-2}$	1.5	21g
Method	σ	k	$\|g_k\|_{L_2}$	ε_{k+1}	figure
CGM (1), $\gamma = 0.01$	0.00	1 000	$1.5 \cdot 10^{-2}$	2.1	22a
	0.03	1 000	$1.5 \cdot 10^{-2}$	2.1	22c
	0.10	1 000	$1.7 \cdot 10^{-2}$	2.0	22e
	0.20	1 000	$2.2 \cdot 10^{-2}$	2.1	22g
CGM (2), $\gamma = 0.01$	0.00	106	$6.8 \cdot 10^{-3}$	1.1	23a
	0.03	84	$8.5 \cdot 10^{-3}$	1.2	23c
	0.10	67	$1.6 \cdot 10^{-2}$	1.2	23e
	0.20	47	$2.5 \cdot 10^{-2}$	1.3	23g
IRCGM, $\gamma_0 = 0.01$	0.00	106	$6.5 \cdot 10^{-3}$	1.2	24a
	0.03	97	$7.7 \cdot 10^{-3}$	1.1	24c
	0.10	65	$1.0 \cdot 10^{-2}$	1.2	24e
	0.20	46	$3.0 \cdot 10^{-2}$	1.3	24g

Table 4: Stop data for lowest possible error norm ε_k for the different noise levels σ and algorithms. CGM (1) uses $g_k = h_k + \gamma(a_k - a_0)$ and CGM (2) uses $g_k = h_k + \gamma(a_k - a_{k-1})$.

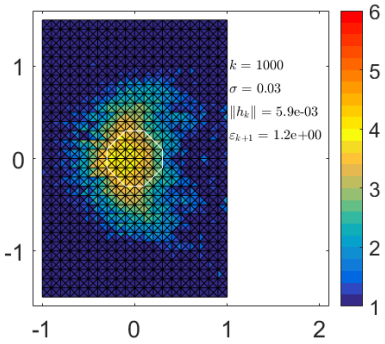
For CGM (2) and IRCGM, the error norm diverges after some number of iterations depending on σ. If γ is chosen higher, the reconstruction might become better, but it will take more iterations. Both algorithms have extremely similar results for the same number of iterations and noise level. The shape is similar to that of FPI, spread out and slightly off center, but the maximum values are higher, and the number of iterations needed to compute the best possible reconstruction are lower.



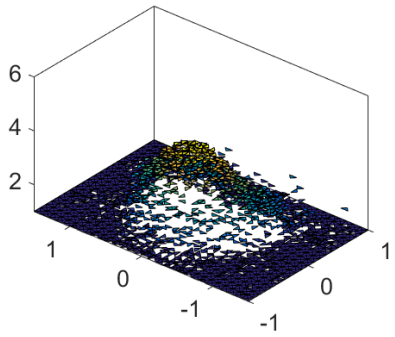
(a) $\sigma = 0.00$, $k = 1000$.



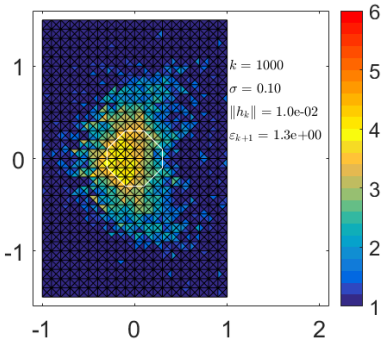
(b) 3D view.



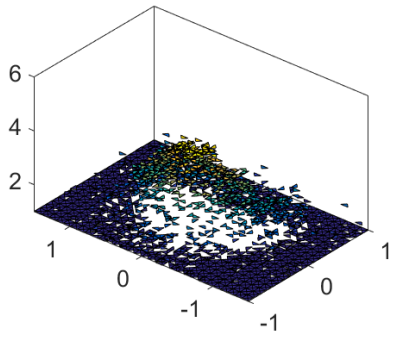
(c) $\sigma = 0.03$, $k = 1000$.



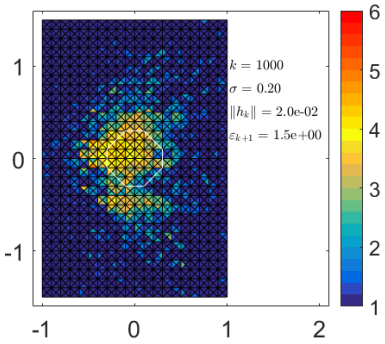
(d) 3D view.



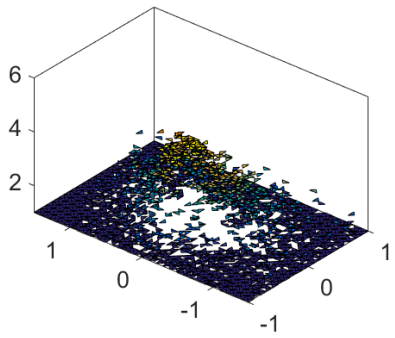
(e) $\sigma = 0.10$, $k = 1000$.



(f) 3D view.

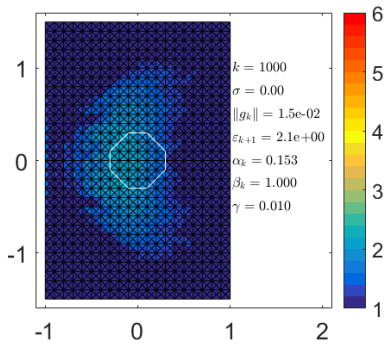


(g) $\sigma = 0.20$, $k = 1000$.

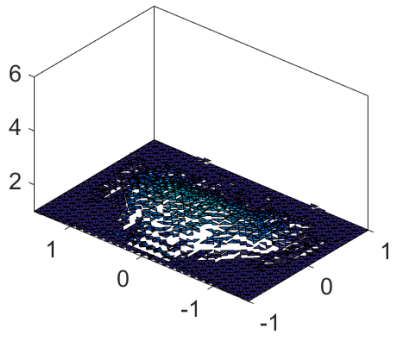


(h) 3D view.

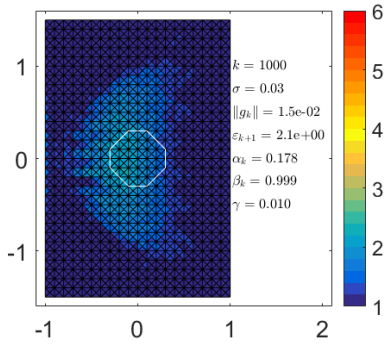
Figure 21: Best reconstructed conductivity $a(x)$ for case 2, using **FPI**, only backscatter data and varying noise level σ.



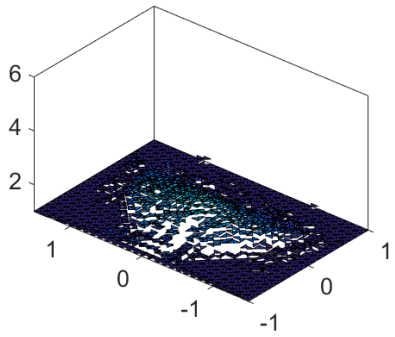
(a) $\sigma = 0.00$, $k = 1000$.



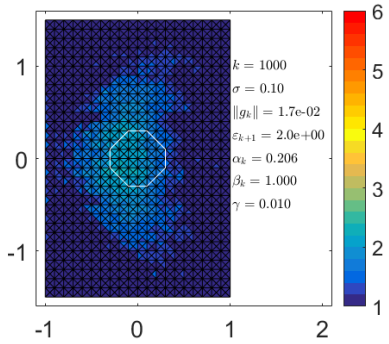
(b) 3D view.



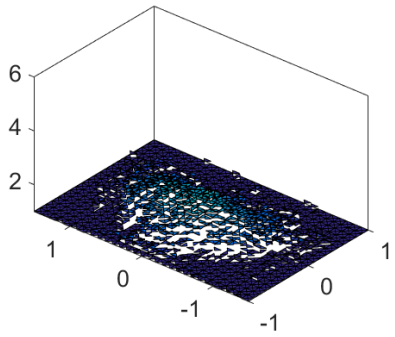
(c) $\sigma = 0.03$, $k = 1000$.



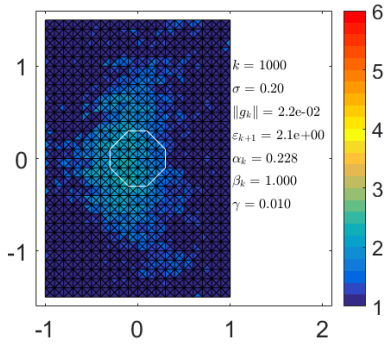
(d) 3D view.



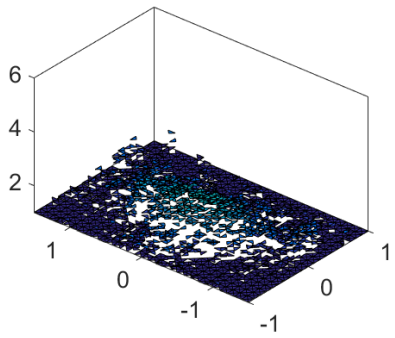
(e) $\sigma = 0.10$, $k = 1000$.



(f) 3D view.

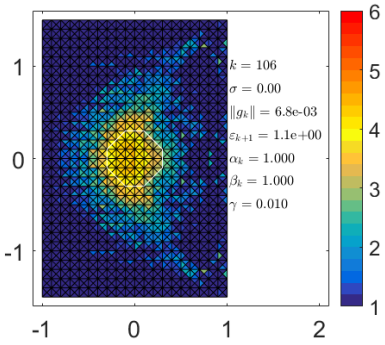


(g) $\sigma = 0.20$, $k = 1000$.

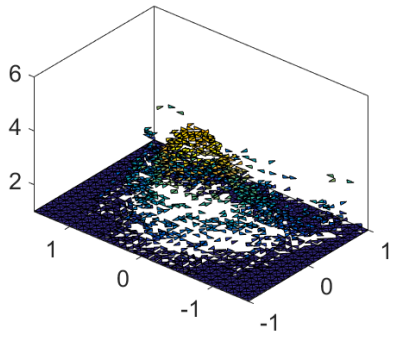


(h) 3D view.

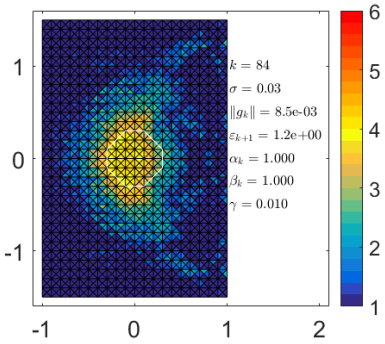
Figure 22: Best reconstructed conductivity $a(x)$ for case 2, using **CGM (1)**, only backscatter data and varying noise level σ.



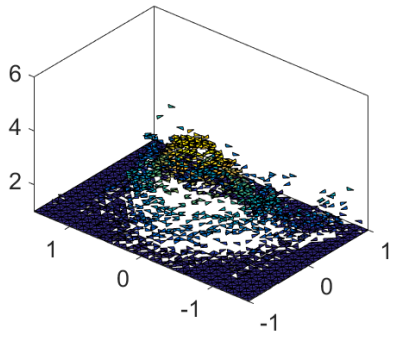
(a) $\sigma = 0.00$, $k = 106$.



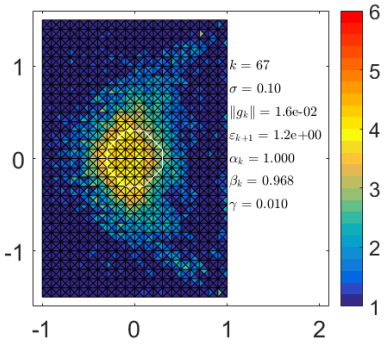
(b) 3D view.



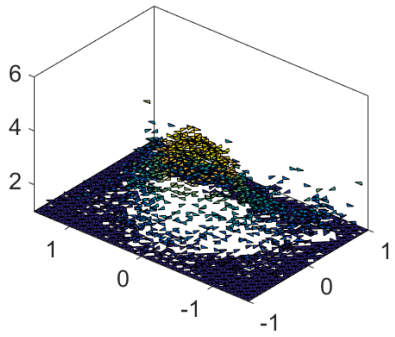
(c) $\sigma = 0.03$, $k = 84$.



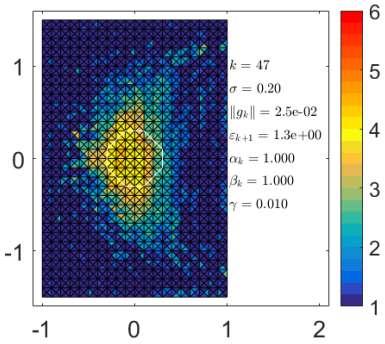
(d) 3D view.



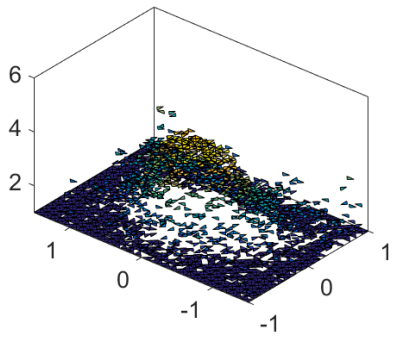
(e) $\sigma = 0.10$, $k = 67$.



(f) 3D view.

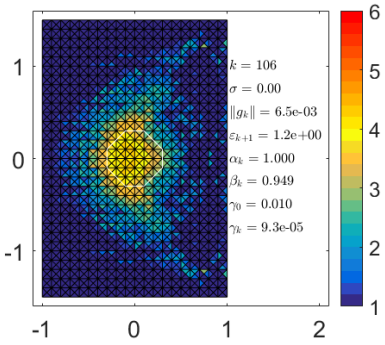


(g) $\sigma = 0.20$, $k = 47$.

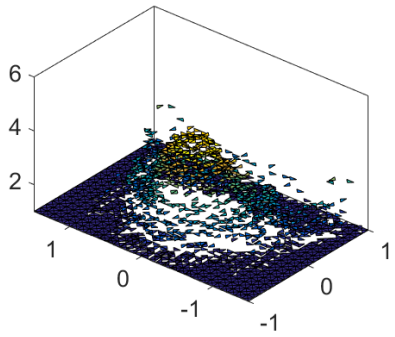


(h) 3D view.

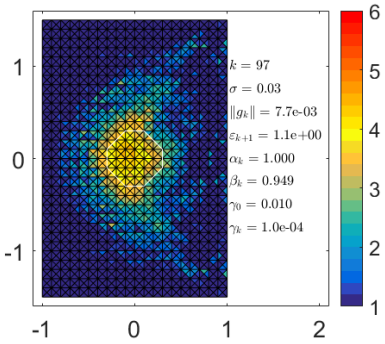
Figure 23: Best reconstructed conductivity $a(x)$ for case 2, using **CGM (2)**, only backscatter data and varying noise level σ.



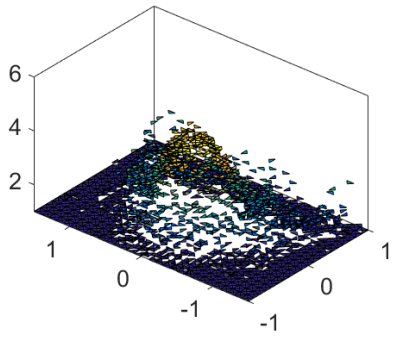
(a) $\sigma = 0.00$, $k = 106$.



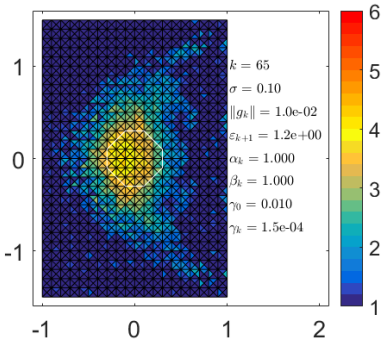
(b) 3D view.



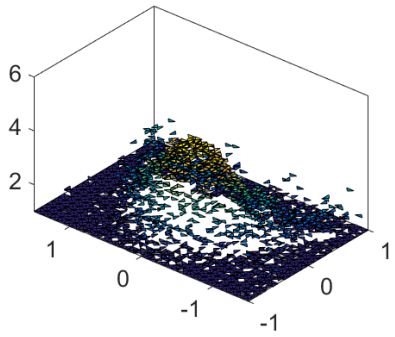
(c) $\sigma = 0.03$, $k = 97$.



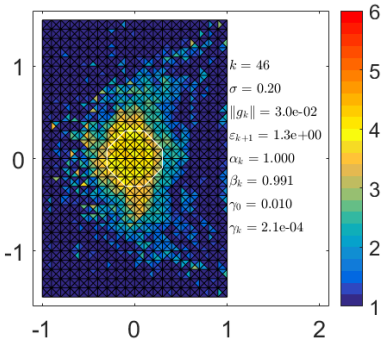
(d) 3D view.



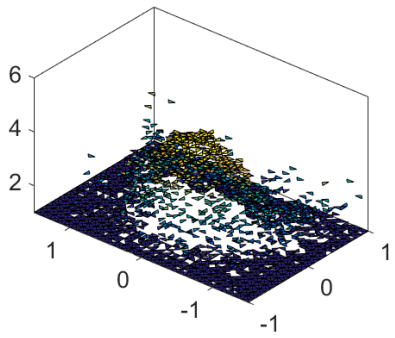
(e) $\sigma = 0.10$, $k = 65$.



(f) 3D view.



(g) $\sigma = 0.20$, $k = 46$.



(h) 3D view.

Figure 24: Best reconstructed conductivity $a(x)$ for case 2, using **IRCGM**, only backscatter data and varying noise level σ.

19 Discussion and conclusion

The iteratively regularized conjugate gradient method (IRCGM) shows promising results for the two case studies. It appears to be the most versatile algorithm that can cope with noisy input data and still be able to reconstruct the conductivity in a satisfactory way.

Conjugate gradient methods updated with $g_k = h_k + \gamma(a_k - a_{k-1})$ (CGM (2)) is able to reconstruct the conductivity extremely good in a faster way than IRCGM, for nice inverse problems such as case 1 that have a good convergence. But the method quickly becomes numerically unstable when noisy input data is used.

Conjugate gradient methods updated with $g_k = h_k + \gamma(a_k - a_0)$ (CGM (1)) is very stable, but the quality of the reconstruction depends greatly on the value of the regularization parameter. To generate acceptable reconstructions it is often necessary to run the simulation multiple times trying different γ:s, which the IRCGM does in a better way.

Fixed point iterations (FPI) performs good on inverse problems with good convergence, but is slower than the conjugate gradient methods. The algorithm becomes numerically unstable for ill-posed inverse problems, but is often able to reconstruct a better conductivity function than CGM (1). The method is the simplest, but the gradient methods are much faster and more suitable when computational time is an issue.

One way of solving the problem where the error norm diverges instead of stabilizing in IRCGM, might be to restrict the sequence of γ_k in equation (117) to only decrease if the gradient g_k decreases. That way, the lowest possible γ_k is found for the conjugate gradient method. In a Matlab program the restriction can be written as:

```
if beta_k < theta2
        gamma_k = gamma_0 / (1+counter)^p;
        counter = counter + 1;
end
```

This means that if $\|g_k\|_{L_2} \geq \|g_{k-1}\|_{L_2}$, then let $\gamma_k = \gamma_{k-1}$, and we can expect the error norm to stabilize just as the CGM (1) algorithm does. The $counter$ is an integer initially set to one. If the algorithm is able to find a a_k closer to a^*, then it is likely that $\|g_k\|_{L_2} < \|g_{k-1}\|_{L_2}$ and γ_k will decrease. The algorithm can then find the a that minimizes the Tikhonov functional for that γ_k.

The gradient norm is not constantly decreasing for all k:s in any of the algorithms, as can be seen in the figures. The trend is of course decreasing, but on the small scale the gradient norm often oscillate. A condition that only allows γ_k to decrease if the gradient norm is smaller than the previous, ensures that the algorithm is not stopped prematurely.

To handle numerical uncertainty when $\|g_k\|_{L_2} \approx \|g_{k-1}\|_{L_2}$, a good statement is $\|g_k\|_{L_2}/\|g_{k-1}\|_{L_2} < \theta$, where $\theta \leq 1$ a small non-negative constant chosen close to one. In computer programs, it is convenient to use

$$\beta_k = \frac{\|g_k\|_{L_2}^2}{\|g_{k-1}\|_{L_2}^2} < \theta^2 \tag{124}$$

to check the statement. The $theta2$ scalar in the Matlab code above, is chosen as $theta2 = \theta^2$. Using a restriction as this means that γ_{k+1} needs to be computed after β_k, for each iteration k in the algorithm, with initial chosen γ_0 and $\beta_0 = 0$.

The CGM (1) can be improved by updating a_0 two or three times, making a combination of CGM (1) and CGM (2). The idea is to chose a_0 and compute a_{k+1} until $\|g_k\|_{L_2}$ is smaller than some tolerance or is stabilized. If the gradient norm is stabilized, then let $a_0 := a_{k+1}$ and run the algorithm one more time.

This might work, because a_0 is supposed to be chosen in the vicinity to the exact a^* and the regularization parameter γ limits the possible set of conductivity functions that minimizes the Tikhonov functional to a neighbourhood around a_0. If we assume that the best reconstruction from the first run of the algorithm is closer to a^*, then it is possible to get better reconstructions using this conductivity as a_0 in a second run of the algorithm.

Of course, all reconstructions improve when transmitted data is available to the inverse problem, and real life applications should benefit greatly by using both sets of data.

19.1 Restricting the iteratively updated regularization parameter

To get an idea of how the error norm from IRCGM behaves when γ_k is only allowed to decrease if equation (124) is fulfilled, I run a couple if simulations on case 1, using only backscatter data and noise level $\sigma = 0.10$. The IRCGM algorithm is compared when updating γ_k with and without the restriction. The result can be seen in figure 25, where the same measured $\tilde{u}$ on the boundary is used in the four simulations. However, it is not exactly the same data-set as used in the results for case 1, due to the randomness of the noise. In the

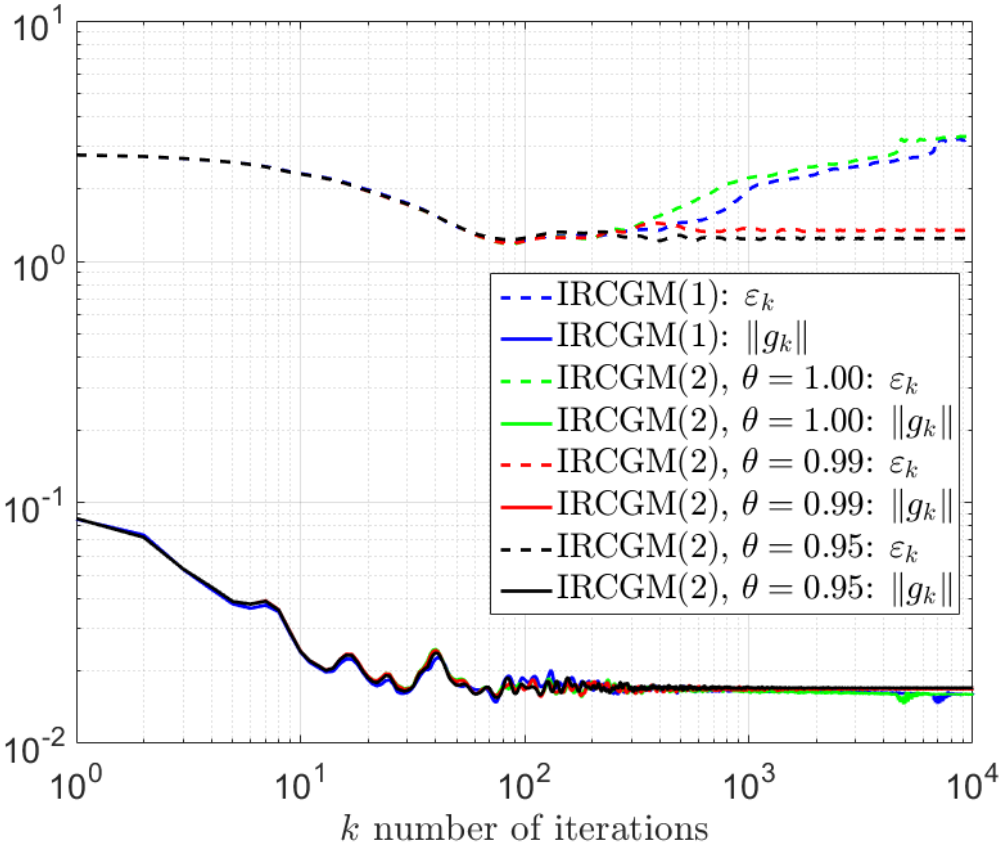


Figure 25: Logarithmic plot of the error norm ε_k and gradient norm $\|g_k\|_{L_2}$ versus the number of iterations k for the IRCGM algorithm with and without a restriction on γ_k, using only backscatter data with noise level $\sigma = 0.10$.

plot **IRCGM (1)** denotes the algorithm as described without any restriction on γ_k, and **IRCGM (2)** includes the if statement, where γ_{k+1} is computed after β_k, if it is less than θ^2 and $\theta \in \{1.00, 0.99, 0.95\}$.

Clearly the error norm stabilizes close to the minimum as $\|g_k\|_{L_2}$ stabilizes, when $\theta < 1$, instead of diverging. All the simulations have approximately the same minimum point for the error norm, which is $\varepsilon_k \approx 1.21 - 1.24$ at $k = 85 - 91$, but the resulting reconstruction at $k = 10000$ differs largely when $\theta < 1$. If the criteria that $\gamma_{k+1} \leq \gamma_k$ is only allowed to decrease if $\|g_k\|_{L_2}/\|g_{k-1}\|_{L_2} < \theta$ is imposed, then the iteratively regularized CGM algorithm can easily use the convergence criteria to stop computing a_{k+1} if the gradient norms are considered stabilized. This can be done practically by comparing the sequence of $\|g_k\|_{L_2}$, or sequence of β_k or counting the number of successive times that $\gamma_{k+1} = \gamma_k$. In IRCGM (1), it is harder to determine when the gradient norm can be considered to be stabilized, and the error norm diverges for each iteration after the minimum point.

The best possible reconstruction obtained in the result for case 1 is better, using IRCGM under the same conditions, but that was computed using a different data-set as input, since the generated random variables used to compute $\tilde{u}$ are not the same.

A 1D linear basis functions and resulting FEM matrices

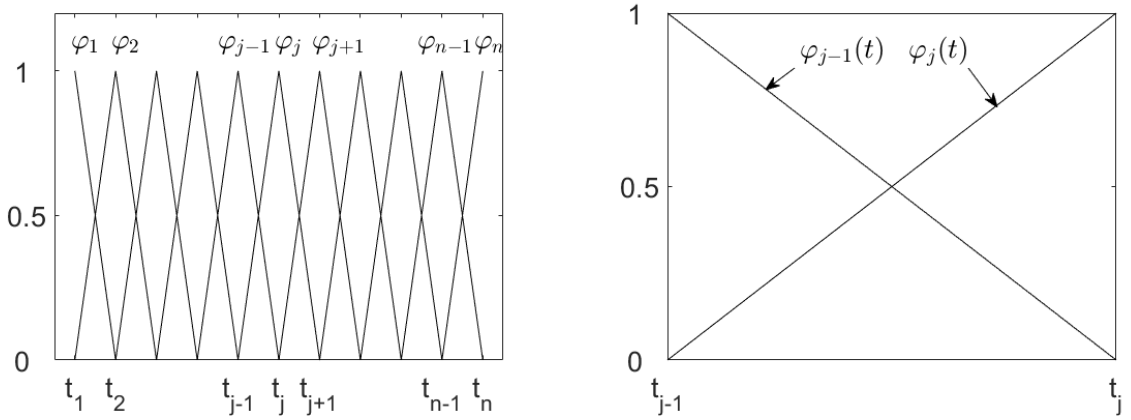
Discretize the interval $J = [t_1, t_n]$ into $n-1$ sub-intervals between the points $t_1 < t_2 < \cdots < t_n$, with step-length $\tau_j = t_j - t_{j-1}$. If we have equal step-length then $\tau_j = \tau$ for all j. The sub-intervals are defined by $J_j = [t_{j-1}, t_j]$. The n hat-functions $\{\varphi_j(t)\}_{j=1}^n$ spanning the interval are then defined by:

$$\text{(first end-point) } \varphi_j(t) = \begin{cases} \frac{t_{j+1}-t}{t_{j+1}-t_j} & \text{if } t \in [t_j, t_{j+1}], \\ 0 & \text{else.} \end{cases} \qquad j = 1 \tag{125a}$$

$$\text{(inner-points) } \varphi_j(t) = \begin{cases} \frac{t-t_{j-1}}{t_j-t_{j-1}} & \text{if } t \in [t_{j-1}, t_j], \\ \frac{t_{j+1}-t}{t_{j+1}-t_j} & \text{if } t \in (t_j, t_{j+1}], \\ 0 & \text{else.} \end{cases} \qquad j = 2, \ldots, n-1 \tag{125b}$$

$$\text{(last end-point) } \varphi_j(t) = \begin{cases} \frac{t-t_{j-1}}{t_j-t_{j-1}} & \text{if } t \in [t_{j-1}, t_j], \\ 0 & \text{else.} \end{cases} \qquad j = n \tag{125c}$$

The hat-functions on the end-points are also known as half hat-functions.



(a) Plotted over J.

(b) Plotted over J_j.

Figure 26: The hat-functions plotted over the time interval J and the subinterval J_j. The subinterval has two non-zero hat-functions and is considered to be a standard element.

A.1 Mass matrix

Define the $n \times n$ symmetric mass matrix R by:

$$r_{ij} = \int_{t_1}^{t_n} \varphi_j(t)\varphi_i(t)\,\mathrm{d}t \qquad i, j \in \{1, \ldots, n\}. \tag{126}$$

The entries in R can be computed to:

$$R = \frac{1}{6}\begin{bmatrix} 2\tau_2 & \tau_2 & & & & & 0 \\ \tau_2 & 2(\tau_2+\tau_3) & \tau_3 & & & & \\ & \tau_3 & 2(\tau_3+\tau_4) & \tau_4 & & & \\ & & \tau_4 & 2(\tau_4+\tau_5) & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & 2(\tau_{n-1}+\tau_n) & \tau_n \\ 0 & & & & & \tau_n & 2\tau_n \end{bmatrix}. \tag{127}$$

When we have equal step-length $\tau_j = \tau$, the matrix is simplified to:

$$R = \frac{\tau}{6}\begin{bmatrix} 2 & 1 & & & & 0 \\ 1 & 4 & 1 & & & \\ & 1 & 4 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & 4 & 1 \\ 0 & & & & 1 & 2 \end{bmatrix}. \tag{128}$$

A.1.1 R_1

In the FEM we have a mass matrix R_1 which is defined by integrating over $(0, t_1)$ instead of over the whole interval $(0, T)$. The time t_1 is not to be confused by the points on the discretized time interval. Assume that t_1 corresponds to the k:th time node, counting $t = 0$ as the first time node and $t = T$ as the last n:th node, then the $n \times n$ matrix R_1 can be defined by block matrices as:

$$R_1 = \begin{bmatrix} \tilde{R} & 0 \\ 0 & 0 \end{bmatrix}, \tag{129}$$

where $\tilde{R}$ is an $k \times k$ matrix defined as equation (128)

A.2 Convection matrix

Define the $n \times n$ convection matrix C by:

$$c_{ij} = \int_{t_1}^{t_n} \varphi_j'(t)\varphi_i(t)\,\mathrm{d}t \qquad\qquad i, j \in \{1, \ldots, n\}, \tag{130}$$

where $c_{ji} = -c_{ij}$ on all entries of C, except at $i = j = 1, n$. The entries in C can be computed to:

$$C = \frac{1}{2}\begin{bmatrix} -1 & 1 & & & & 0 \\ -1 & 0 & 1 & & & \\ & -1 & 0 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & 0 & 1 \\ 0 & & & & -1 & 1 \end{bmatrix}. \tag{131}$$

C is not dependent on τ_j and remains the same when equal step-length $\tau_j = \tau$ is used.

A.2.1 C_1

In the FEM we have a convection matrix C_1 which is defined by integrating over (t_1, T) instead of over the whole interval $(0, T)$. The time t_1 is not to be confused by the points on the discretized time interval. Assume that t_1 corresponds to the k:th time node, counting $t = 0$ as the first time node and $t = T$ as the last n:th node, then the $n \times n$ matrix C_1 can be defined by block matrices as:

$$C_1 = \begin{bmatrix} 0 & 0 \\ 0 & \tilde{C} \end{bmatrix}, \tag{132}$$

where $\tilde{C}$ is an $(n - k + 1) \times (n - k + 1)$ matrix defined as equation (131).

A.3 Stiffness matrix

Define the $n \times n$ symmetric stiffness matrix S by:

$$s_{ij} = \int_{t_1}^{t_n} \varphi_j'(t)\varphi_i'(t)\,\mathrm{d}t \qquad\qquad i, j \in \{1, \ldots, n\}. \tag{133}$$

The entries in S can be computed to:

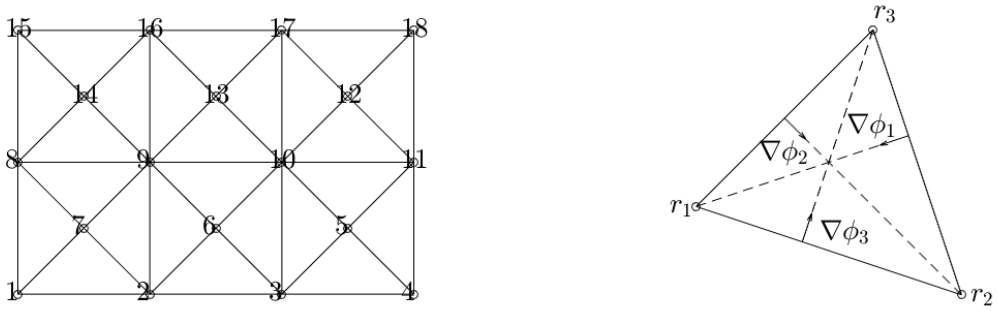
$$S = \begin{bmatrix} \frac{1}{\tau_2} & -\frac{1}{\tau_2} & & & & & & 0 \\ -\frac{1}{\tau_2} & (\frac{1}{\tau_2} + \frac{1}{\tau_3}) & -\frac{1}{\tau_3} & & & & & \\ & -\frac{1}{\tau_3} & (\frac{1}{\tau_3} + \frac{1}{\tau_4}) & -\frac{1}{\tau_4} & & & & \\ & & -\frac{1}{\tau_4} & (\frac{1}{\tau_4} + \frac{1}{\tau_5}) & \ddots & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & (\frac{1}{\tau_{n-1}} + \frac{1}{\tau_n}) & -\frac{1}{\tau_n} & \\ 0 & & & & & -\frac{1}{\tau_n} & \frac{1}{\tau_n} & \end{bmatrix}. \tag{134}$$

If equal step length is used, S is simplified to:

$$C = \frac{1}{\tau}\begin{bmatrix} 1 & -1 & & & & 0 \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & 2 & -1 \\ 0 & & & & -1 & 1 \end{bmatrix}. \tag{135}$$

B 2D linear basis functions and resulting FEM matrices

Discretize the domain Ω into K_h by drawing arbitrary, adjoint triangles on it, so that there are n nodes and q elements in K_h. The triangles can be acute-, right- or obtuse-angled. Each triangle is determined by its three corner nodes. It is assumed throughout the computations, that the corner nodes for each triangle is given in a counter-clockwise fashion. This ensures that area computations are always positive for each element. The two spatial axes are denoted by x and y.



(a) Triangulation K_h of Ω.

(b) Standard element A.

Figure 27: Above is an example of an triangulation of an square domain, and one arbitrary standard element. The three corner nodes in the standard element are denoted 1, 2, 3 instead of the more arbitrary i, j, k.

B.1 Computing basis functions on one arbitrary triangle element

Let $r_i = (x_i, y_i)$, $i \in \{1, 2, 3\}$ denote the corner nodes that span the arbitrary standard triangle element denoted A. The continuous piecewise linear function $u(x, y)$ is then described as:

$$u(x,y) = c_1 + c_2 x + c_3 y = \begin{bmatrix} 1 & x & y \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}, \tag{136}$$

where the unknown constants c_1, c_2, c_3 are specific for each element. Assume that $u(x, y)$ is known at the corner nodes so that $u_i = u(r_i)$. This gives three equations for three unknown constants:

$$\begin{cases} u_1 = c_1 + c_2 x_1 + c_3 y_1 \\ u_2 = c_1 + c_2 x_2 + c_3 y_2 \\ u_3 = c_1 + c_2 x_3 + c_3 y_3 \end{cases} \Leftrightarrow \underbrace{\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}}_{=v} = \underbrace{\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}}_{=R} \underbrace{\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix}}_{=c}, \tag{137}$$

where R^{-1} can be computed via method of cofactors and $c = R^{-1}v$, inserting c in equation (136) gives $u(x,t) = [1, x, y]R^{-1}v = \phi(x,y)^T v$, where the row vector $\phi(x,y)^T = [1, x, y]\, R^{-1}$. When computed, the

three basis functions that span the element are:

$$\begin{cases} \phi_1(x,y) = \dfrac{1}{d}\Big(x_2y_3 - x_3y_2 + x(y_2-y_3) + y(x_3-x_2)\Big), \\ \phi_2(x,y) = \dfrac{1}{d}\Big(x_3y_1 - x_1y_3 + x(y_3-y_1) + y(x_1-x_3)\Big), \\ \phi_3(x,y) = \dfrac{1}{d}\Big(x_1y_2 - x_2y_1 + x(y_1-y_2) + y(x_2-x_1)\Big), \\ \qquad d = x_2y_3 - x_3y_2 + x_3y_1 - x_1y_3 + x_1y_2 - x_2y_1, \end{cases} \tag{138}$$

where $d = \det(R)$ is twice the $\mathrm{area}(A)$ of the triangle element, d can be computed easily in Matlab using a skew-symmetric matrix D so that $d = y^T Dx$, that is:

$$d = \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix} \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 2 \cdot \mathrm{area}(A). \tag{139}$$

B.2 Element mass matrix

The 3×3 symmetric element mass matrix M computed on one element. M is defined by:

$$m_{ij} = \iint_A \phi_j \phi_i \,\mathrm{d}x\mathrm{d}y \qquad\qquad i,j \in \{1,2,3\}, \tag{140}$$

where A denotes the element spanned by the three corner nodes.

To compute this integral you can rotate the (x,y)-coordinate system by substituting $u = r^T n_u$ and $v = r^T n_v$, so that $\phi_i(u,v) = \phi_i(u)$. This is the case when n_u is parallel to the gradient of ϕ_i. Then determine $\phi_j(u,v)$ and compute the integral.

The resulting element mass matrix is:

$$M = \frac{d}{24} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}. \tag{141}$$

B.3 Element stiffness matrix

The 3×3 symmetric element stiffness matrix K computed on one element. K is defined by:

$$k_{ij} = \iint_A a\,(\nabla\phi_j)^T \nabla\phi_i \,\mathrm{d}x\mathrm{d}y \qquad\qquad i,j \in \{1,2,3\}, \tag{142}$$

where A denotes the element spanned by the three corner nodes, and $a(x,y)$ is the piecewise constant conductivity. Using the definitions from equation (138), we get that:

$$\begin{cases} \nabla\phi_1 = \dfrac{1}{d} \begin{bmatrix} -(y_3-y_2) \\ x_3-x_2 \end{bmatrix}, \\ \nabla\phi_2 = \dfrac{1}{d} \begin{bmatrix} y_3-y_1 \\ -(x_3-x_1) \end{bmatrix}, \\ \nabla\phi_3 = \dfrac{1}{d} \begin{bmatrix} -(y_2-y_1) \\ x_2-x_1 \end{bmatrix}. \end{cases} \tag{143}$$

Since the scalar product of the gradients does not depend on (x, y), and $a(x, y)$ is constant in each element, we get that $k_{ij} = (\nabla\phi_j)^T \nabla\phi_i \operatorname{area}(A)\, a$, where element $\operatorname{area}(A) = d/2$.

One easy way to compute K in Matlab, is to define the Jacobian $J = \frac{\mathrm{d}(\phi_1,\phi_2,\phi_3)}{\mathrm{d}(x,y)}$ by $J^T = \nabla\phi^T$ or $J = [\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}]$, then $K = (ad/2)JJ^T$.

B.4 Global matrices

The global $n \times n$ matrices are computed by adding up all the q element matrices. As an example, compute the global mass matrix by:

$$\begin{aligned} K_h &= \bigcup_{l=1}^{q} K_l, \\ m_{ij} &= \iint_{K_h} \phi_j\phi_i \,\mathrm{d}x\mathrm{d}y = \sum_{l=1}^{q} \iint_{K_l} \phi_j\phi_i \,\mathrm{d}x\mathrm{d}y \quad i,j \in \{1,2,\ldots,n\}, \end{aligned} \tag{144}$$

where $\iint_{K_l} \phi_j\phi_i \,\mathrm{d}x\mathrm{d}y$ is non-zero when i, j is one of the three nodes that span element K_l.

As an example: if b_{el} is the set of three nodes that span element K_{el} and M_{el} is the 3×3 element mass matrix, then the global mass matrix M is easily constructed in Matlab by:

```
M = zeros(n,n);
for i = 1:q
        % Determine b_el and compute M_el.
        M(b_el,b_el) = M(b_el,b_el) + M_el;
end
```

B.5 Mass matrix M_1 and M_2

The mass matrices M_1 and M_2 along Γ_1 and Γ_2, used in the FEM equations, are defined by:

$$\begin{aligned} m_{ij}^{(1)} &= \int_{\Gamma_1} \phi_j(x)\phi_i(x) \,\mathrm{d}s \quad i,j \in \{1,2,\ldots,n\}, \\ m_{ij}^{(2)} &= \int_{\Gamma_2} \phi_j(x)\phi_i(x) \,\mathrm{d}s \quad i,j \in \{1,2,\ldots,n\}, \end{aligned} \tag{145}$$

where $m_{ij}^{(1)}$ is non-zero for nodes i, j on Γ_1 and $m_{ij}^{(2)}$ is non-zero for nodes i, j on Γ_2. The integral is analogue to computations made for the 1D linear basis functions on J_τ. Assume that b_i is the set of nodes that are on Γ_i for $i \in \{1, 2\}$, arranged in an counter-clockwise way. Then M_i can easily be computed in Matlab by:

```
M_i = zeros(n,n);
% Determine b_i
% Compute block matrix M_tilde along boundary.
M_1(b_i,b_i) = M_1(b_i,b_i) + M_tilde;
```

As an example: the triangulation in figure 27a have $b_1 = \{15, 8, 1\}$ and $b_2 = \{4, 11, 18\}$. Let e_i for $i \in \{1, \ldots, n = 18\}$ be unit vectors, then M_1 and M_2 are computed by:

$$\tilde{M} = \frac{h_y}{6} \begin{bmatrix} 2 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 2 \end{bmatrix},$$

$$\begin{bmatrix} e_{15}^T \\ e_8^T \\ e_1^T \end{bmatrix} M_1 \begin{bmatrix} e_{15} & e_8 & e_1 \end{bmatrix} = \tilde{M}, \tag{146}$$

$$\begin{bmatrix} e_4^T \\ e_{11}^T \\ e_{18}^T \end{bmatrix} M_2 \begin{bmatrix} e_4 & e_{11} & e_{18} \end{bmatrix} = \tilde{M},$$

where h_y is the equal step length in the y-direction.

References

[1] L. Beilina and K. Niinimäki, *Numerical studies of the Lagrangian approach for reconstruction of the conductivity in a waveguide*, arXiv:1510.00499, 2015.

[2] M. Asadzadeh, *An Introduction to the Finite Element Method (FEM) for Differential Equations*, lecture notes available at http://www.math.chalmers.se/~mohammad/teaching/PDEbok/draft_FEM_version6.pdf, 2016.

[3] L. Beilina, *Domain decomposition finite element/finite difference method for the conductivity reconstruction in a hyperbolic equation*, *Communications in Nonlinear Science and Numerical Simulation*, Elsevier, doi:10.1016/j.cnsns.2016.01.016, preprint available at arXiv:1509.01399, 2015.

[4] J. R. Senning, *Computing and estimating the rate of convergence*, lecture notes available at http://www.math-cs.gordon.edu/courses/ma342/handouts/rate.pdf, 2007.

[5] S. Hosseinzadegan, *Iteratively Regularized Adaptive Finite Element Method for Reconstruction of Coefficients in Maxwell's System*, master's thesis available at http://publications.lib.chalmers.se/records/fulltext/218637/218637.pdf, 2015.

[6] H. W. Engl, M. Hanke and A. Neubauer, *Regularization of Inverse Problems*, Kluwer Academic Publishers, 2000.

[7] S. Larsson and V. Thomée, *Partial Differential Equations with Numerical Methods*, Springer, 2009, 2003.

[8] J. W. Demmel, *Applied Numerical Linear Algebra*, Siam, 1997.

[9] N. Ottosen and H. Petersson, *Introduction to the Finite Element Method*, Prentice Hall, 1992.

[10] L. Beilina and S. Hosseinzadegan, *An adaptive finite element method in reconstruction of coefficients in Maxwell's equations from limited observations*, arXiv:1510.07525, 2015.

[11] A. B. Bakushinsky, M. Y. Kokurin and A. Smirnova, *Iterative Methods for Ill-Posed Problems*, de Gruyter, 2011.