# Technical Report

TR-2006-016

**Adaptive finite volume method for distributed non-smooth parameter identification**

by

Uri Ascher, Eldad Haber, Stefan Heldmann

Mathematics and Computer Science

**EMORY UNIVERSITY**

# Adaptive finite volume method for distributed non-smooth parameter identification

Eldad Haber  Stefan Heldmann *  Uri Ascher †

December 6, 2006

**Abstract**

In this paper we develop a finite volume adaptive grid refinement method for the solution of distributed parameter estimation problems with almost discontinuous coefficients. We discuss discretization on locally refined grids, as well as optimization and refinement criteria. An OcTree data structure is utilized. We show that local refinement can significantly reduce the computational effort of solving the problem, and that the resulting reconstructions can significantly improve resolution, even for noisy data and diffusive forward problems.

## 1 Introduction

In this paper we study an efficient method for the solution of distributed parameter estimation problems of the form

$$\min_{m,u} \quad \frac{1}{2}\sum_{j=1}^{n} \|Q_j u_j - \mathbf{d}_j\|^2 + \beta \int_\Omega \rho(|\nabla m|)\, d\mathbf{x} \tag{1.1a}$$

$$\text{s.t} \quad \mathcal{A}(m)u_j - b_j = 0 \quad j = 1, \ldots, n. \tag{1.1b}$$

Here $m$ is the sought model, $u_j$ are the fields that are obtained for a given model as the solutions of (1.1b), $\mathcal{A}(m)$ is a (typically elliptic) PDE operator, $\mathbf{d}_j$ are some discrete measured data which correspond to the $j^{\text{th}}$ field, and $Q_j$ is a projection operator that projects $u_j$ to its measurement locations. The rightmost term in (1.1a) is a regularization functional, where $\beta$ is a regularization parameter. We are particularly interested in regularization operators with functions $\rho(\cdot)$ that do not over-penalize rapid profile changes in the model; see e.g. [12, 4].

Numerical optimization problems of this kind are often solved in geophysics [21], medical physics [11], computer vision [9] and other fields which involve data fitting. To be more

*Department of Mathematics and Computer Science, Emory University, Atlanta, GA.
†Department of Computer Science, University of British Columbia, Vancouver.

concrete we motivate this paper using the following 3D model problem which arises in Direct Current resistivity, where one attempts to invert for the log-conductivity $m = \ln \sigma$ (see for example [34, 29, 33] and references therein) the elliptic problems associated with

$$\mathcal{A}(m)u_j = \nabla \cdot (\mathrm{e}^m \nabla u_j) = b_j \quad j = 1 \dots n. \tag{1.2}$$

For simplicity, we assume that Neumann boundary conditions are prescribed for these PDEs, but other boundary conditions could be easily incorporated. The model problem has multiple right hand sides, which is typical to many other inverse problems (e.g. [23]).

Even for the relatively simple forward problem defined by (1.2) the computational cost can be substantial. This is because each PDE can be time consuming to solve in 3D, especially when the coefficient function $m$ varies rapidly. In the case that $m$ is outright discontinuous, it is not always clear that this highly ill-posed inverse problem can be meaningfully solved [4, 1]. In such a case it is particularly advantageous to use additional a priori information if possible. Additional restrictions to the solution space can be in the form that $m$ is allowed to take at each spatial location only one of two values, which turns the problem into one of shape optimization [17, 36, 10]. More generally, however, no such information may be available, and we do not assume it in the present article. If $m$ varies rapidly but smoothly then we can seek to resolve regions of such rapid variation using a very fine grid. Practically solving the inverse problem then becomes rather challenging, and advanced computational techniques are needed.

One option to achieve major computational savings is by using a multilevel approach [2, 3, 17]. Such an approach is particularly beneficial in this context because it enables us to obtain a good approximation to the regularization parameter, $\beta$, as well as the solution $m$ on coarse grids, reducing the number of more expensive fine grid iterations. Nevertheless, the cost of the algorithm is dominated by the fine grid problem [2]. For problems on regular orthogonal grids this can be expensive because no adaptivity is used and the overall number of unknowns is large. It is therefore advantageous to use adaptive grids which can dramatically reduce the number of the unknowns in the discrete problem.

There are several options for the design of adaptive grids. One could use unstructured grids and finite elements which are naturally designed for adaptivity. In our experience such an approach, although conceptually attractive, is potentially slow due to significant overhead and the uneasy transfer of data blocks from memory into cache. The following related difficulties arise. Since the parameter $m$ is changing through the optimization problem, re-gridding is often needed. However, re-gridding for large scale 3D problems is a computationally intensive task by itself (see for example [31]). Moreover, the stiffness matrix (i.e. the assembled discrete version $A$ of $\mathcal{A}(m)$) must be re-evaluated when $m$ changes. This is also a time consuming process, especially when comparing with the standard structured grid approximations. Finally, derivatives are required, and this also necessitates element-by-element computations.

A different approach is to use orthogonal finite difference or finite volume methods, allowing only a very restricted form of local refinement. Thus, in 1D a grid subinterval may only be refined into two equal halves, and this principle of *local* refinement is extended to

grids in higher dimensions, resulting in a QuadTree in 2D or an OcTree data structure in 3D (see e.g. [22, 27], or just google 'octree' for much, much more). Such an approach has the advantage that gridding is almost trivial, and it yields re-usable sparse matrices. Also, it has been demonstrated that such an approach can deal with adaptivity in the forward problem [15, 14, 25].

While adaptive grid refinement is certainly not a new field, little such work has been done in the general context of inverse problems, and even less attention has been given to recovering rapidly varying coefficient functions. A framework for adaptive finite elements for inverse problems has been presented in [6]. Similar work in the context of finite elements is reported in [7, 8]. See also [26] for a different approach. The above work assumes quadratic regularization operators and obtains smooth model solutions. No previous work on adaptive finite volume methods for inverse problems is known to us.

The goal of this article is to develop highly efficient finite volume solvers for inverse problems of the form (1.1), exploring and generalizing adaptive refinement on orthogonal nested grids. Such adaptation is particularly interesting if the regularization operator allows for very steep solutions. In this case, adaptivity allows us to "zoom into" potentially increasing resolution without much increase in computational cost.

The transition from uniform grids to nonuniform ones, and from smooth model solutions to models that admit abrupt changes, yields several issues that this paper addresses. In Section 2 we discuss the discretization on OcTrees of the forward and inverse problems, as well as the choice of the function $\rho$ in the regularization functional. In Section 3 we describe an optimization technique to solve the inverse problem given a particular grid. Following this, in Section 4 we present an adaptive multilevel refinement strategy, and in Section 5 we use numerical experiments to demonstrate the advantage of our approach. Finally, in Section 6 we summarize our findings.

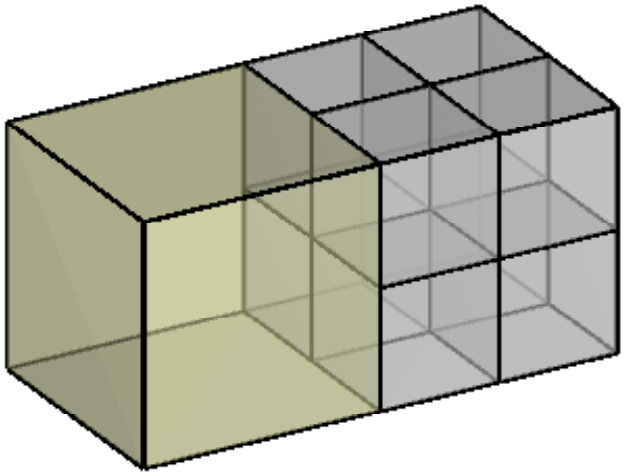
# 2 Problem discretization

In this section we discuss the discretization of the problem (1.1). As in [2] we envision a uniform underlying coarse orthogonal grid with cell width $h_c$ and a uniform underlying fine orthogonal grid of size $2^{N_1} \times 2^{N_2} \times 2^{N_3}$ with cell width $h_f = 2^{-nlevel} h_c$. The fine grid is determined by our desire for high resolution, and the coarse grid is inexpensive to work with while still producing a solution with coarse resolution that is accurate enough for our application. The difference from [2] is that the local grid width varies between $h_c$ and $h_f$, allowing for a larger number of levels *nlevel* while still maintaining efficiency.

Next we review the OcTree data structure and explain how to effectively discretize simple differential operators that are needed in the course of the solution of the inverse problem.

## 2.1 OcTree data structure

The Quad/OcTree representation is an efficient way to compress gridded data, storing a finer solution detail in parts where large changes occur. Our grid is composed of $N$ square/cube



(a)

$$S(:,:,1) = \begin{pmatrix} 2 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

$$S(:,:,2) = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

(b)

Figure 1: (a) OcTree and (b) its representation as a $2 \times 4 \times 2$ (sparse) array.

cells of different widths. Each cell can have a width $h = 2^{-l}h_c$, for some $0 \leq l \leq nlevel$. To make the data structure easier to handle and the discretization more accurate we allow only a factor of 2 change between adjacent cells. This obviously leads to a tree structure depicting the refined grid, where each node has up to 4 children in a QuadTree and up to 8 in an OcTree. The data is then stored as a sparse array. The size of each cell is stored in the upper left corner of the array. This allows us to quickly find neighbors, which is a major operation in the discretization process. This data structure is closely related to the one suggested in [22]. An example of a small 3D grid is plotted in Figure 1.

Given a particular OcTree grid one has to decide where to discretize the different variables. Here, similar to [2] we choose to use staggered discretization and discretize $u$ at the nodes and $m$ at cell centers. As shown in [2] this discretization of the optimization problem is h-elliptic. For the importance of h-elipticity, see [35].

## 2.2 Discretization of the forward problem

To discretize the forward problem on the OcTree grid we consider its Ritz form

$$\min_u \int_\Omega \left[\frac{\sigma}{2}|\nabla u|^2 \; - \; ub\right] d\mathbf{x}, \tag{2.3}$$

rather than the PDE directly. To demonstrate, let us use a 2D example; the extension to 3D is straightforward.

Consider the QuadTree cell plotted in Figure 2. The quantity $\frac{\partial u}{\partial x_i}$ is naturally discretized to second order accuracy along the centers of the $x_i$-edges of each cell, $i = 1, 2$. For the

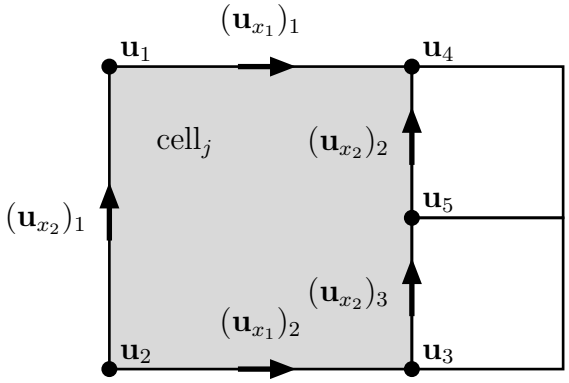



Figure 2: Discretization of $\nabla \mathbf{u}$

QuadTree in Figure 2 we can write

$$\begin{aligned}
(u_{x_1})_2 &= (2h)^{-1}(u_3 - u_2) + \mathcal{O}(h^2) \\
(u_{x_1})_1 &= (2h)^{-1}(u_4 - u_1) + \mathcal{O}(h^2) \\
(u_{x_2})_1 &= (2h)^{-1}(u_1 - u_2) + \mathcal{O}(h^2) \\
(u_{x_2})_2 &= h^{-1}(u_4 - u_5) + \mathcal{O}(h^2) \\
(u_{x_2})_3 &= h^{-1}(u_5 - u_3) + \mathcal{O}(h^2).
\end{aligned}$$

Using this simple, second order difference we can discretize the gradient of $u$ on the Quad/OcTree. To discretize the optimization formulation (2.3) we write

$$\int_\Omega \frac{\sigma}{2} |\nabla u|^2 d\mathbf{x} = \sum_{\text{cells}} \int_{\text{cell}} \frac{\sigma}{2} |\nabla u|^2 d\mathbf{x}.$$

We assume that $\sigma$ is constant over each of the cells, which is consistent with mixed finite element formulations. Next, we approximate the integral over each cell using the edge approximations for the derivatives of $u$. For regular cells (with neighbors of the same size) only 4 edges need to be considered.

In the case of an irregular cell such as the one plotted in Figure 2, we average the square of the gradient to cell center, that is

$$\begin{aligned}
\sigma_{\text{cell}} \int_{\text{cell}} (u_{x_1})^2 \, d\mathbf{x} &= \frac{\sigma_{\text{cell}} V_{\text{cell}}}{4h^2} \left( (u_4 - u_1)^2 + (u_3 - u_2)^2 \right) + \mathcal{O}(h^2), \\
\sigma_{\text{cell}} \int_{\text{cell}} \left( (u_{x_2})^2 \right)^2 \, d\mathbf{x} &= \frac{\sigma_{\text{cell}} V_{\text{cell}}}{4h^2} \left( \frac{1}{2}(u_4 - u_5)^2 + \frac{1}{2}(u_5 - u_3)^2 + (u_3 - u_2)^2 \right) + \mathcal{O}(h^2),
\end{aligned}$$

where $V_{\text{cell}}$ is the cell's volume. The corresponding term $\int_{\text{cell}} ub \, d\mathbf{x}$ is discretized by simple averaging of node values over the cell. Summing over all of the cells we obtain an $\mathcal{O}(h^2)$ approximation to the optimization problem (2.3).

In our view, it is useful to express the above formulas in a "matrix friendly" form. To this end we define the matrix GRAD as the difference matrix which computes the gradient

of $u$ on the edges of the cells. We also let $\mathbf{A}_e^c = \text{diag}\left\{\mathbf{A}_{e1}^c, \mathbf{A}_{e2}^c\right\}$ be an averaging matrix from the edges to the cell center of each cell. Also, let $\mathbf{A}_n^c$ be an averaging matrix from the nodes of the grid to the cell center grid. Then the integral in (2.3) can be approximated by

$$\int_\Omega \frac{1}{2}\left[\sigma|\nabla u|^2 - ub\right]\ d\mathbf{x} \ = \ \frac{1}{2}\mathbf{v}^\top \text{diag}\left(\sigma\right)\mathbf{A}_e^c\ \left((\text{GRAD}\mathbf{u}) \odot (\text{GRAD}\mathbf{u})\right) - \mathbf{v}^\top \mathbf{A}_n^c(\mathbf{b} \odot \mathbf{u}) + \mathcal{O}(h^2),$$

where $\mathbf{u}$ and $\mathbf{b}$ are nodal OcTree grid functions, $\sigma$ is a cell-center grid function, and $\mathbf{v}$ is a vector that contains the volume of each cell. After some algebra this can be rearranged into a more familiar form

$$\min_{\mathbf{u}} \frac{1}{2}\mathbf{u}^\top \text{GRAD}^\top\ \left(\text{diag}\left((\mathbf{A}_e^c)^\top\sigma\right)\text{diag}\left(\mathbf{v}\right)\right)\ \text{GRAD}\mathbf{u}\ -\ \mathbf{u}^\top\ \text{diag}\ \left((\mathbf{A}_n^c)^\top\mathbf{v}\right)\ \mathbf{b}. \tag{2.4}$$

Finally, to obtain the discrete system for the forward problem we differentiate with respect to $\mathbf{u}$ to obtain

$$\text{GRAD}^\top\ \left(\text{diag}\left((\mathbf{A}_e^c)^\top\sigma\right)\text{diag}\left(\mathbf{v}\right)\right)\ \text{GRAD}\mathbf{u}\ =\ \text{diag}\ \left((\mathbf{A}_n^c)^\top\mathbf{v}\right)\ \mathbf{b}. \tag{2.5}$$

Note that given a particular grid, the matrices $\text{GRAD}, \mathbf{A}_n^c$ and $\mathbf{A}_e^c$ are computed only once. We can then reuse them for different grid functions $\sigma$ in order to quickly assemble the forward modeling matrix.

## 2.3 Discretization of the regularization operator

For the regularization operator we require the discretization of the **grad** of cell centered variables. Although our code is for problems in 3D, it is again worthwhile to follow the discretization in 2D for simplicity.

There are various ways to define the discrete cell-center gradient operator. First, it is important to note that the gradient of cell centered variables is naturally defined on cell faces. One simple way to define the gradient is to use a short difference. This approximation has been investigated by [15] and recently used in [28].

Consider the arrangement in Figure 3 and let $\mathbf{x}_0$ be the grid location where $(m_{x_1})_2$ is discretized. With $h$ the width of the smaller cell the approximation is

$$\frac{\partial m}{\partial x_1}(\mathbf{x}_0) \approx \frac{m_2 - m_1}{h}. \tag{2.6a}$$

For subdomains with uniform cells this is a second order approximation, but for areas where adjacent cells are nonuniform this approximation is only $\mathcal{O}(1)$ pointwise, because it corresponds to approximating $m$ by a constant throughout the large square in Figure 3, and this $\mathcal{O}(h)$ approximation gets divided by $h$ when forming derivative approximations. Assuming that $m$ is piecewise constant is consistent with our forward discretization (Section 2.2). Moreover, differentiating a piecewise constant function in the context of ODE mesh refinement has been known and used for years [5]. But here the resulting approximation

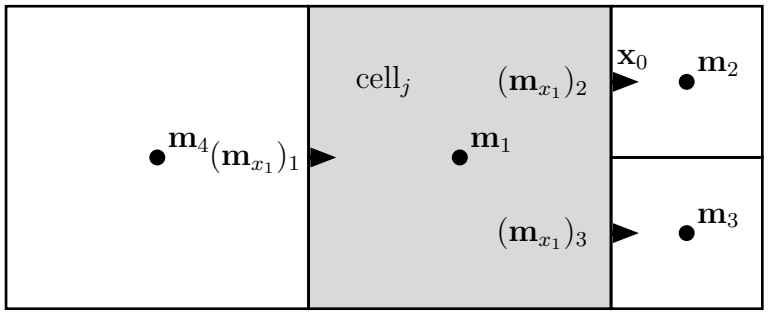



Figure 3: Discretization of $\nabla m$

appears in the objective function and the grid is adaptively refined based on computational quantities, so we opt to be more careful.

A better pointwise approximation to the derivative at the cell's edge is

$$\frac{\partial m}{\partial x_1}(\mathbf{x}_0) \approx \frac{m_2 + m_3 - 2m_1}{3h}. \tag{2.6b}$$

It is easy to see that this one is $\mathcal{O}(h)$ accurate. The extension to 3D is straightforward. The approximation to the gradient of $m$ in any other direction are done in a similar way.

Given the approximation of the gradient of $m$ we approximate the integral of $\rho(|\nabla m|)$ at the cell center

$$\int_{\text{cell}} \rho(|\nabla m|) = \int_{\text{cell}} \rho(\sqrt{m_{x_1}^2 + m_{x_2}^2})\, d\mathbf{x} =$$
$$V_{\text{cell}}\rho\left\{\sqrt{\frac{1}{2}\left(\frac{m_2 + m_3 - 2m_1}{3h}\right)^2 + \frac{1}{2}\left(\frac{m_4 - m_1}{2h}\right)^2 + \text{approx } (m_{x_2})^2}\right\} + \mathcal{O}(h).$$

Once again, we rewrite the expression using matrices and vectors. For this, let $\text{GRAD}_c$ be the cell centered gradient matrix and let $\mathbf{A}_f^c$ be an averaging matrix from the grid faces to the cell centers. Then the discrete regularization operator can be expressed as

$$R(\mathbf{m}) = \mathbf{v}^\top \rho\left(\sqrt{\mathbf{A}_f^c(\text{GRAD}_c\mathbf{m})^2}\right). \tag{2.7}$$

## 2.4 Discretization of the optimization problem

To create the discrete version of the optimization problem (1.1), we must discretize $n+1$ grid functions, namely, the fields $u_j,\ j = 1, \ldots, n$, and the model $m$. First, note that we need not have a single grid for all $u_j$ and $m$, so assume that $u_j$ is discretized on a grid $S_{u_j}$ while the model $m$ is discretized on the grid $S_m$. We further assume that each of the $u_j$-grids is *either*

*finer or the same* as the $m$-grid. This discretization results in $n$ grid functions shaped into vectors $\mathbf{u}_j$ for $u_j$ and a grid function likewise shaped into a vector $\mathbf{m}$ for $m$. We also use an interpolation matrix $P_j$ in order to transfer $\mathbf{m}$ to the grid $S_{u_j}$. With these grid functions we can now discretize the optimization problem.

Using the discrete forward problem (2.5), we approximate $\mathcal{A}(m)u_j = \nabla \cdot (e^m \nabla u_j)$ in our model problem (1.2) with

$$\mathrm{GRAD}^\top \left(\mathrm{diag}\left((\mathbf{A}_e^c)^\top \exp(-P_j\mathbf{m})\right)\mathrm{diag}\left(\mathbf{v}\right)\right) \mathrm{GRAD}\mathbf{u}_j \;=\; A(P_j\mathbf{m})\mathbf{u}_j = \mathbf{b}_j. \tag{2.8}$$

To make the forward modeling unique we also require that the integral of $\mathbf{u}_j$ vanish, which leads to the discrete constraint $\mathbf{v}^\top \mathbf{u}_j = 0$, where $\mathbf{v}$ is a vector that contains the volume of each cell on our grid.

The regularization functional in (1.1a) employs a function $\rho(|\nabla m|)$. For this we use a Huber function depending on a switching parameter $\gamma$

$$\rho(\tau) \;=\; \begin{cases} \tau, & \tau \geq \gamma \\ \tau^2/(2\gamma) + \gamma/2, & \tau < \gamma \end{cases}. \tag{2.9}$$

The parameter $\gamma$ was selected adaptively in [4] based on resolution considerations with the overall desire to stay as close as it makes sense to total variation. Here the purpose is somewhat different because the local grid refinement may be viewed as stretching steep gradients on a local scale, and we select $\gamma$ depending on a user's notion of the typical size of a jump in the model.

The discretization of the misfit term is straightforward. Given the approximate field vector $\mathbf{u}$ we use linear interpolation to approximate it at the measurement location.

Collecting the above discretized quantities we obtain a discrete optimization problem

$$\min_{\mathbf{m},\mathbf{u}} \quad \frac{1}{2}\sum_{j=1}^{n} \|Q_j\mathbf{u}_j - \mathbf{d}_j\|^2 + \beta\mathbf{v}^\top \rho\left(\sqrt{A_f^\top(\mathrm{GRAD}_c\mathbf{m})^2}\right) \tag{2.10a}$$

$$\text{s.t} \quad A(P_j\mathbf{m})\mathbf{u}_j - \mathbf{b}_j = \mathbf{0}, \quad j = 1, \ldots, n. \tag{2.10b}$$

# 3 Solving the optimization problem

In this Section we quickly review solution techniques for the optimization problem. For a more in-depth discussion see [20].

We use variants of Reduced Hessian Sequential Quadratic Programming to solve the optimization problem. The Lagrangian can be written as

$$\begin{aligned} \mathcal{L} \;&=\; \frac{1}{2}\sum_{j=1}^{n} \|Q_j\mathbf{u}_j - \mathbf{d}_j\|^2 + \beta\mathbf{v}^\top \rho\left(\sqrt{A_f^\top(\mathrm{GRAD}\mathbf{m})^2}\right) \\ &+\; \sum_{j=1}^{n} \mathbf{p}_j^\top \mathrm{diag}(\mathbf{v})(A(P_j\mathbf{m})\mathbf{u}_j - \mathbf{b}_j). \end{aligned} \tag{3.11}$$

Upon differentiation we obtain the following system of equations:

$$A(P_j\mathbf{m})^\top \mathbf{p}_j = Q_j^\top(\mathbf{d}_j - Q_j\mathbf{u}_j), \quad j = 1,\ldots,n \tag{3.12a}$$

$$A(P_j\mathbf{m})\mathbf{u}_j = \mathbf{b}_j, \quad j = 1,\ldots,n \tag{3.12b}$$

$$\sum_k P_k^\top G(P_k\mathbf{m},\mathbf{u}_k)^\top \mathbf{p}_k + \beta\,\mathrm{GRAD}^\top\,\Sigma(\mathbf{m})\,\mathrm{GRAD}\,\mathbf{m} = \mathbf{0}, \tag{3.12c}$$

where for a grid function $\mathbf{w}$

$$G(\mathbf{w},\mathbf{u}_j) = \frac{\partial(A(\mathbf{w})\mathbf{u}_j)}{\partial\mathbf{w}},$$

and $\Sigma(\mathbf{m})$ is an approximation to the Hessian of the regularization term. Here we use the lagged diffusivity approach, see [37, 4].

We can then use an inexact Newton method for the solution of the system. For details on the linear systems and preconditioning of the Newton iteration, see [19, 18].

When facing a very large number (thousands) of sources and a limited computer memory one may consider working with a reduced space approach rather than the full space approach. The reduced space approach implies that $\mathbf{u}_j$ and $\mathbf{p}_j$ are eliminated first, and then we solve for $\mathbf{m}$. The advantage of a reduced space approach, for this particular problem structure, is that we are able to store only a single $\mathbf{u}_j$ and $\mathbf{p}_j$ at a time. To see this note first that we are able to compute the solution of the forward problem (3.12a) and the adjoint problem (3.12b) without any communication between the two or between any other forward or adjoint problems. The reduced gradient is then obtain by substituting the fields $\mathbf{u}_j$ and the Lagrange multipliers $\mathbf{p}_j$ in (3.12c). We define the reduced gradient of the data misfit

$$\mathbf{g} := \sum_k P_k^\top G(\mathbf{m},\mathbf{u}_k(\mathbf{m}))^\top \mathbf{p}_k(\mathbf{m}).$$

To compute $\mathbf{g}$, observe that we can break the computation over each term in the sum. Starting from $\mathbf{g} = \mathbf{0}$, assume that the $k^{\text{th}}$ forward and adjoint problems have been solved. We can then calculate the product $\mathbf{g}_k = P_k^\top G(\mathbf{m},\mathbf{u}_k)^\top \mathbf{p}_k$ and set

$$\mathbf{g} \leftarrow \mathbf{g} + \mathbf{g}_k.$$

After evaluating $\mathbf{g}_k$ we can write over both $\mathbf{u}_k$ and $\mathbf{p}_k$ in order to compute the next field and Lagrange multiplier function. Finally, adding the regularization term we complete the evaluation of the reduced gradient.

Unfortunately, one cannot "get away" without storing $\mathbf{u}_j$ when a product of the reduced Hessian and a vector is computed. Therefore, the reduced Hessian approach is not as attractive in this case and one would benefit working with an all-at-once approach. Nevertheless, if we are limited in memory, given the reduced gradient, we are able to effectively use Quasi-Newton techniques without the necessity of excessive storage. In our application we often need to deal with a very large number of sources. We therefore use the reduced space L-BFGS [30] approach for the solution of the problem.

# 4 Adaptive multilevel refinement

The cost of the optimization process is directly impacted by the size of the problem and our initial guess for the solution. Adaptive multilevel refinement methods are targeted to achieve a low-cost good starting guess by using coarse grids, and to reduce the size of the discrete fine grid problem by using adaptive nested grids that refine only in areas where the error in the solution is large. Unfortunately, finding a unique refinement criterion that works for different problems is rather difficult (e.g. [35]). As we see next, further complications can arise from the fact that we solve a constrained optimization problem.

Before describing our refinement strategy, let us first discuss the choice for our grids. The solution of the optimization problem requires solving the discretized Euler-Lagrange equations (3.12) for the unknowns $\mathbf{u}_j$, $\mathbf{m}$, $\mathbf{p}_j$. At least in principle, one could envision different grids for each of these variables. While different grids could be used and they may be cheaper for solving each of the Euler-Lagrange equations, they may generate other difficulties.

- If the grid for the fields $\mathbf{u}_j$ and the grid for the Lagrange multipliers $\mathbf{p}_j$ are not identical then the forward operator $A$ is not the discrete adjoint of the operator for the Lagrange multiplier. In this case the *discrete* Euler-Lagrange equations do not evolve from a discrete Lagrangian and therefore the *discrete* reduced gradient is not a gradient of anything. This implies that no discrete objective function is decreased and even simple unconstrained optimization algorithms may fail. We therefore keep the grids for $\mathbf{p}_j$ and $\mathbf{u}_j$ identical.

- If the grid for the model $\mathbf{m}$ is strictly finer than the grid for $\mathbf{u}_j$ then homogenization is needed in order to accurately evaluate $A(\mathbf{m})$ on the $\mathbf{u}_j$ grid. It is well known that homogenization is not a trivial process [13, 24]. In fact, some of the better homogenization processes are highly nonlinear with respect to the coefficients. This can generate further complications when solving the optimization problem. We therefore insist that the $\mathbf{u}_j$ grid contain the grid for $\mathbf{m}$.

- It is possible, and can be desirable, to have a coarser grid for $\mathbf{m}$ than for $\mathbf{u}_j$ and $\mathbf{p}_j$. In fact, one may claim that if a field $\mathbf{u}_j$ is more or less constant then we cannot obtain "real" information from it about $\mathbf{m}$. Nevertheless, structure in $\mathbf{m}$ may appear even in areas where $\mathbf{u}_j$ is more or less constant, forced by the regularization term. Since the regularization term describes crucial a priori information about our highly ill-posed problem, we should attempt not to dilute it by using a very coarse grid. Thus, the grid for the fields may contain refinement regions that would not normally arise if it were not for the present inverse problem context.

- Although it may seem that we are able to choose uniform starting grids, experiments show that such an approach can perform poorly. This is because information about the sources and data locations is not utilized. In such a case the coarse grid is often too coarse and the data is not accurately simulated. The approach leads to high bias in the

forward model, which in turn leads to artificial structures in the recovered model. This can be prevented or improved by using the available information about the sources and receivers when designing initial grids.

## 4.1 Refinement criteria for m

Let us assume for the moment that we have sufficiently fine grids for the fields and concentrate on refining a grid for the model $\mathbf{m}$ alone. We next develop a refinement algorithm using two termination parameters, $\zeta$ and $\nu$.

Note that our smoothness assumptions on the model are embodied in the regularization operator, which in turn depends directly on $|\nabla m|$, providing anisotropic smoothing. Thus, in areas where the gradient is large we may assume that the grid should be finer, while in areas where the model is relatively flat no further refinement is needed.

Consider solving the problem on grid $S_k^m$, obtaining $\mathbf{m}_k$. We evaluate the regularization operator given the grid and the model as

$$R(m) = \sum_{\text{cells}} \int_{\text{cell}} \rho(|\nabla m|)\, d\mathbf{x} \;=\; \sum_{\text{cells}} \mathbf{r}_{\text{cell}}.$$

In Section 2 we have discussed how to evaluate this integral over each cell in the OcTree grid yielding a vector $\mathbf{r}(\mathbf{m}_k, S_k^m)$. We then refine all cells that satisfy $|\mathbf{r}_{\text{cell}}(\mathbf{m}_k, S_k^m)| > \zeta$. The refinement process is terminated when $|\mathbf{r}_{\text{cell}}| \leq \zeta$ for all cells.

Once a grid is refined we must re-solve on it. In order to terminate this iteration we use a simple criterion. Assume that the solution process on grid $S_k$ starts from the model $\mathbf{m}_k^0$. This model is an interpolated version of the one obtained using the previous grid. Let $\mathbf{m}(S_k)$ be the obtained optimal model on this grid. If

$$\|\mathbf{m}_k^0 - \mathbf{m}(S_k)\|_\infty \leq \nu$$

then we conclude that further refinement is unnecessary and terminate the iteration.

The choice of $\zeta$ and $\nu$ typically depends on what is considered to be a large gradient and changes in material properties.

## 4.2 Initializing and refining the grid for u and p

Having a good criterion for the refinement of the model is insufficient: further complications arise because the problem is strongly coupled with respect to the fields. We now discuss how to refine the $u_j$ grids given the refinement in $m$.

It may seem that one could use any coarse grid for the discretization of (1.1b) for $u_j$. However, our experience shows that carelessness could lead to catastrophic results. In order to explain why, let $\mathbf{m}_0$ be the discretization of the initial model $m_0$ (for example, assume that $m_0$ is a constant), and let $\mathbf{d}(m_0) = \mathcal{Q}\mathcal{A}(m_0)^{-1}b$ be the corresponding data predicted by the initial model. Writing the model that minimizes the functional as $m^* = m_0 + m_1$, our algorithm should be able to retrieve a discrete approximation to $m_1$. We cannot expect this

to happen if the initial data we start the optimization from does not contain any signal, that is, we expect to obtain $m_1$ only if $\|\mathbf{d} - \mathbf{d}(m_0)\|$ is large enough. If this distance is smaller than the noise level then there is no point in solving the inverse problem and we can set $\mathbf{m}_0$ as the solution. In short, an overly coarse grid for the forward problem may lead to a poor model reconstruction that nonetheless fits the data satisfactorily.

Let $S_0$ represent the grid for $u$ and $p$, and let $A(\mathbf{m}_0; S_0)$ be the discretization of the forward problem for $\mathbf{m}_0$ on $S_0$. Further, assume for simplicity that $n = 1$ and that $Q$ is the identity in (1.1a), and let $\mathbf{d}(\mathbf{m}_0, S_0) = A(\mathbf{m}_0; S_0)^{-1}\mathbf{b}$. We can write

$$\delta\mathbf{d} = \mathbf{d} - \mathbf{d}(m_0) = \delta\mathbf{d}(S_0) + (\mathbf{d}(\mathbf{m}_0, S_0) - \mathbf{d}(m_0)).$$

Since we evaluate our solution on the grid $S_0$ the term $\mathbf{d}(\mathbf{m}_0, S_0) - \mathbf{d}(m_0)$ can be regarded as noise in our evaluation of $\delta\mathbf{d}$. Thus, if $\|\mathbf{d}(\mathbf{m}_0, S_0) - \mathbf{d}(m_0)\| > \|\delta\mathbf{d}(S_0)\|$ we have added a substantial amount of highly correlated noise to the problem and there is no chance to recover a good approximation to $m_1$. Furthermore, since the noise is highly correlated, methods such as Generalized Cross Validation (GCV) [16] cannot be used reliably, and the algorithm over-fits the data, which in turn generates excessive structure in $\mathbf{m}$.

Therefore, our strategy for the choice of the initial grid for $u$ and $p$ is to require that

$$\|\mathbf{d}(\mathbf{m}_0, S_0) - \mathbf{d}(m_0)\| \leq \eta\|\delta\mathbf{d}(S_0)\|, \tag{4.13}$$

where $\eta < 1$ is yet another parameter. Evaluating the left hand side of (4.13) may not be trivial. In some cases $\mathbf{d}(m_0)$ is known analytically, but in other cases it requires the numerical evaluation of $u$ given $m_0$, which can be done by standard adaptive grid refinement techniques for the forward problem (see for example [35] and references therein).

In principle, a good choice for the parameter $\eta$ ensures that

$$\|\mathbf{d}(\mathbf{m}_0, S_0) - \mathbf{d}(m_0)\| \ll \|\text{noise}\|.$$

Such a choice guarantees that we are able to pull out most of the information from the given data without polluting it with numerical noise. However, this requirement can lead to over-discretization if the data is very accurate. Indeed, for the noiseless case the $u$ grid width must go to 0 if we use this criterion alone. In practice, for very low noise levels the coarsest grid tends to be very fine and no real benefit from a later multilevel refinement approach is obtained. We therefore use an iterative method in order to refine our $u$-grid.

Assume now that we have used the initial grid to solve the inverse problem, obtaining a model $\mathbf{m}_1$. Given this new model, we now evaluate the new grid $S_1$. There are two requirements for this new grid. First, letting $S_1^m$ be the $m$-grid which corresponds to $m_1$, we need $S_1$ to contain $S_1^m$. Second, we require that

$$\|\mathbf{d}(\mathbf{m}_1, S_1) - \mathbf{d}(m_1)\| < \eta\|\delta\mathbf{d}(S_1)\|,$$

where $m_1$ is a function obtained by piecewise constant interpolation of $\mathbf{m}_1$. If $\|\mathbf{d}(\mathbf{m}_1, S_1) - \mathbf{d}(m_1)\| \ll \|\text{noise}\|$ then the discretization of $u$ is sufficient and no further discretization is needed. Therefore, if $\mathbf{m}_1$ is an acceptable model then the algorithm is terminated. However,

**Algorithm 1** Adaptive Grid Refinement

given $m_0$ evaluate $\mathbf{m}_0$ and the $m$ grid $S_0^m$;
set $\eta, \xi, \nu, \zeta < 1$
set $k = 0$
**while** not STOP **do**
  evaluate $S_k$ such that

  - $S_k^m \in S_k$
  - $\|\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)\| \leq \eta \|\delta \mathbf{d}(S_k)\|$

  **if** $\mathbf{m}_k$ is an acceptable model **then**
    **if** $\|\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)\| \leq \xi \|\text{noise}\|$ **then**
      STOP
    **end if**
  **end if**
  Solve the optimization problem on grid $S_k$ and obtain $\mathbf{m}_{k+1}$
  **if** $\|\mathbf{m}_{k+1} - \mathbf{m}_k\|_\infty \leq \nu$ **then**
    STOP
  **end if**
  given $\mathbf{m}_{k+1}$ obtain $S_{k+1}^m$ with tolerance $\zeta$
  set $k \leftarrow k + 1$
**end while**

if this is not the case then one may want to re-solve the problem on $S_1$ starting from $\mathbf{m}_1$. We summarize our approach for adaptive grid refinement in Algorithm 1.

Next we choose the initial grid in our numerical experiments. When we start with a constant $\mathbf{m}_0$, $\mathbf{d}_0$ is known analytically. We have found it sufficient to then use 2 or 3 levels of refinement close to the sources and receivers as an initial grid $S_0$. We have further found that this grid hardly needs to be refined in order to obtain very low levels of data accuracy.

# 5 Numerical experiments

In this section we experiment with our algorithm and show that using adaptive grids is beneficial in the context of inverse problems. We generate data for different models on a fine uniform grid and add 1% noise to it. We then use Algorithm 1 with the following choice of parameters: $\eta = 0.1, \xi = 0.5, \nu = 0.05, \zeta = 0.01$. For each optimization problem on different grids, we terminate the optimization process if the norm of the gradient is reduced by two orders of magnitudes from its initial value. We use L-BFGS(20) for the solution of each optimization problem. All optimization problems have converged to the specified tolerance in less than 50 steps.

We have experimented with different parameter values and observed that the results are

| Level | $u$ grid | $m$ grid |
|---|---|---|
| L1 | 6448 | 512 |
| L2 | 6910 | 946 |
| L3 | 8974 | 3095 |
| L4 | 19547 | 15170 |

Table 1: Number of unknowns on different levels

not very sensitive to these choices.

## 5.1 Experiment I

In our first experiment we use only a single source located at [0.1.0.1.0.1] and $[0.9, 0.9, 0.9]$, but we sample the space using 1000 points equally spaced in $[0.2, 0.8]^3$. Here, the model has some nontrivial shape presented in Figure 4 top left. Note that this shape is well represented on a $128^3$ uniform grid (which the data is generated from, but we allow only refinement of up to $64^3$). The grid for $u$ and $p$ is set by choosing a uniform $16^3$ grid as the coarsest and refining twice close to the sources. When comparing the data obtained on this first grid with a half space data we see that the agreement is roughly 1%, which is within the noise levels. To initialize the grid for $m$ we use an $8^3$ uniform grid. We then solve the problem on a sequence of grids and use the refinement criteria specified above for $m$. The resulting $m$ on different grids is presented in Figure 4. Interestingly, each of the models depicted fits the data to within 1% error. That is, from a data fitting point of view all the above models are valid! Nevertheless, it is obvious that the $8^3$ grid does not yield a satisfactory result and that locally refining the grid, even though the data does not justify doing so, does give better reconstructions. This is because the regularization adds valid and useful information to the inverse problem. If we treat our regularization seriously (and we do) then its discretization needs to be faithful to the continuous formulation. For this example, the information provided by the regularization gives a more accurate reconstruction of the model we are after. In Figure 4 top right we plot the model on a uniform $64^3$ grid. It is evident that the finest OcTree model gives virtually the same result. Thus, our locally refined grid has allowed recovery of the results obtained on the uniform $64^3$ grid but with far less computational effort.

In Figure 5 we plot a slice from the model. The OcTree for $m$ for different levels is plotted in Figure 6. The number of unknowns for each level is summarized in Table 1. Note that although our final resolution is equivalent to the one obtained on a $64^3$ grid we used roughly a factor of 13 fewer cells! The computational time dramatically scales by at least that factor.

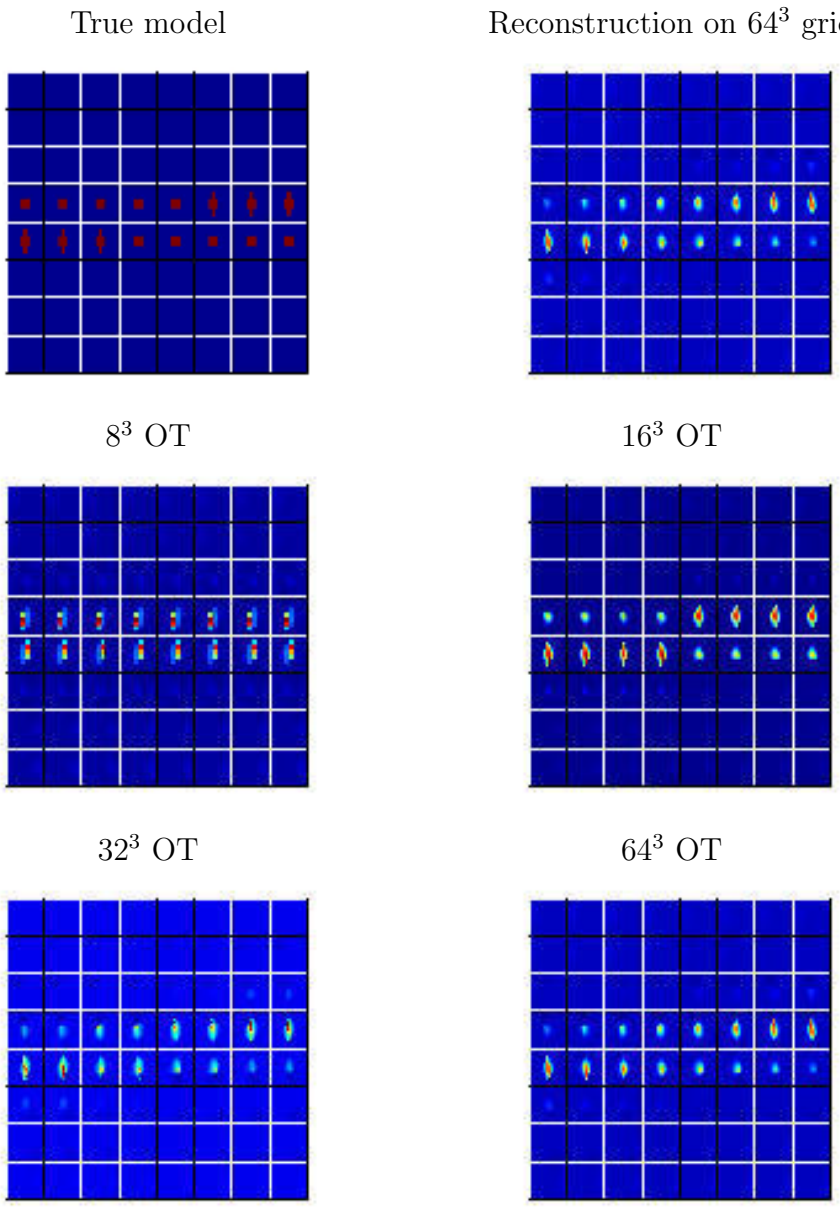



Figure 4: Recovered models on different OcTree grids. Top left is the true model and top right is the reconstruction on a uniform $64^3$ grid. The $8^3$ OT grid is uniform. The other OT grids are adaptive refinements of one, two and three levels. The reconstructions on the $64^3$ and on the $64^3$ OT grids appear virtually the same even though the latter as 13 times fewer cells.

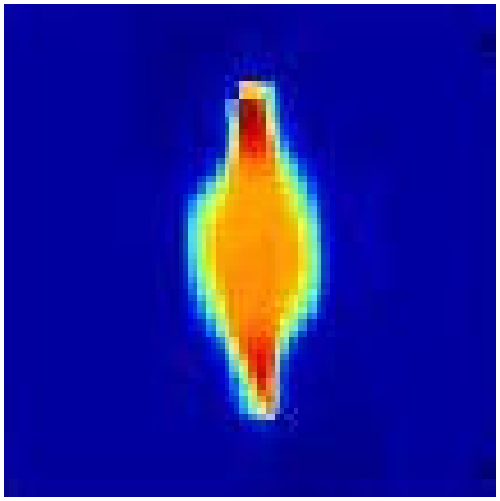


Figure 5: A slice through the recovered model.

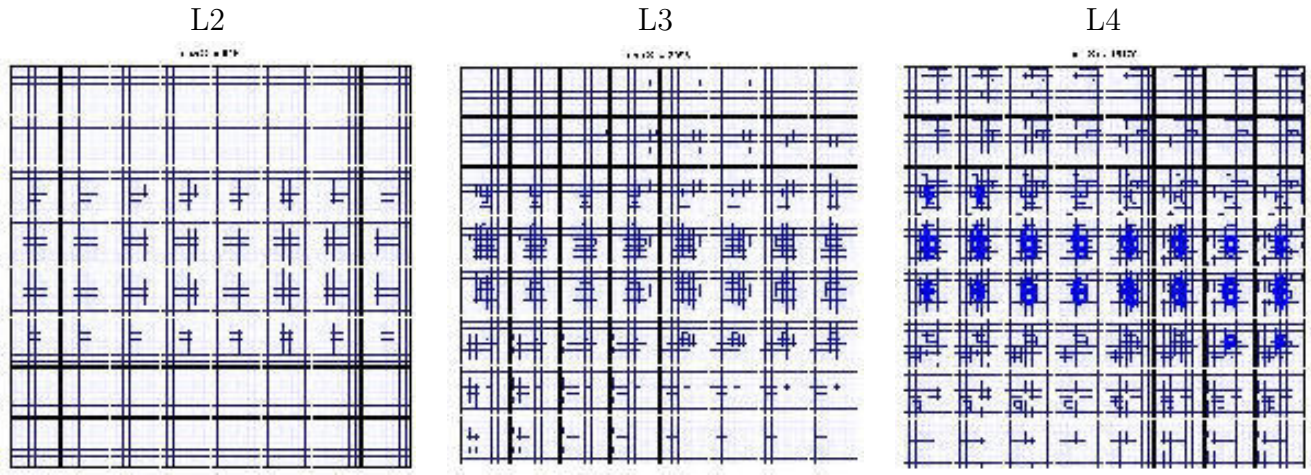



Figure 6: Grids for $m$ at different levels. The grid on the first level is equidistant.

## 5.2 Experiment II

In our second experiment we demonstrate how a better resolution can be obtained by using many sources. Of course, this is obtained at the price of a substantial increase in computational cost to solve the problem. Our experimental setting is similar to the field setting presented in [32], where sources are placed in boreholes and data is measured on the surface of the earth. In our setting we have a grid of $64^2$ receivers on the surface of the earth. We decree the existence of 4 boreholes, where 40 sources are placed. The total number of data from this experimental setting is $64^2 \times 40 = 163,840$. A sketch of the experimental setting is plotted in Figure 7.

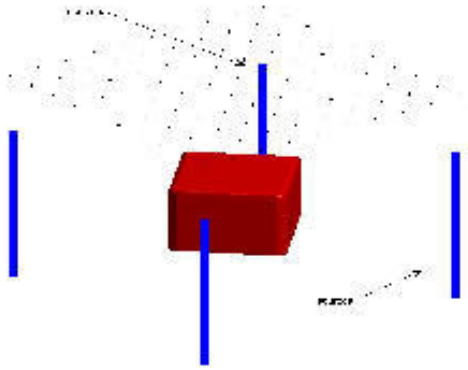


Figure 7: The model and the experimental setting of our second experiment.

In this case the model is a cube that *does not* coincide with the axes and is generated by the function

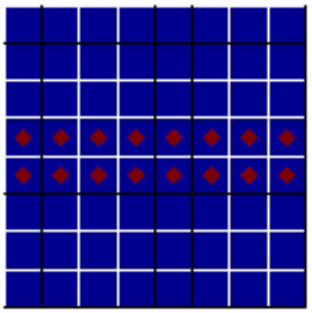
$$\begin{aligned} m(x,y,z) &= (\text{atan}(a(x+y+0.75)) - \text{atan}(a(x+y-0.75))) \times \\ &\quad (\text{atan}(a(x-y+0.75)) - \text{atan}(a(x-y-0.75))) \times \\ &\quad (\text{atan}(a(z+0.75)) - \text{atan}(a(z-0.75))) \end{aligned}$$

where $-3 \leq \{x, y, z\} \leq 3$.

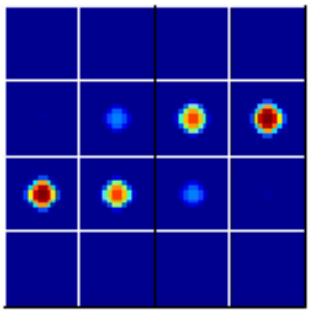
We start by solving the problem on a uniform $16^3$ grid for $m$ and an OcTree of 8756 cells for $u$. The OcTree for $u$ is obtained again from the known analytical solution of the problem. We then continue to refine the grid. Figure 8 displays the models obtained using 3 levels of refinement.

Once again, all models on all grids fit the data within 1% of the noise level. Therefore, if our interest was data fitting only then there is no preference to the fine resolution model over the coarse resolution one. However, since the regularization operator is appropriate for this particular problem, using finer grids and letting the regularization operator "sharpen" the model yields very pleasing results.

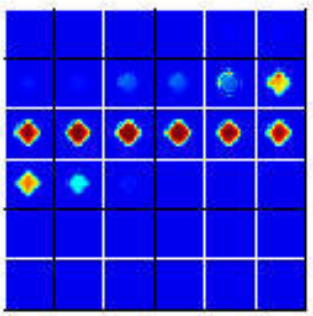
True model



Reconstruction on $16^3$ grid



Reconstruction on $32^3$ OT grid



Reconstruction on $64^3$ OT grid

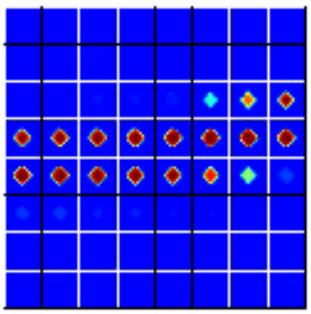


Figure 8: Recovered models for the second experiment.

| Level | $u$ grid | $m$ grid |
|---|---|---|
| L1 | 5656 | 4096 |
| L2 | 10356 | 6014 |
| L3 | 21342 | 16199 |

Table 2: Number of unknowns on different levels

In Figure 9 we plot the $m$ grids obtained in the second experiment. We can observe that the code has automatically refined the grid close to the cube edges to obtain higher resolution.

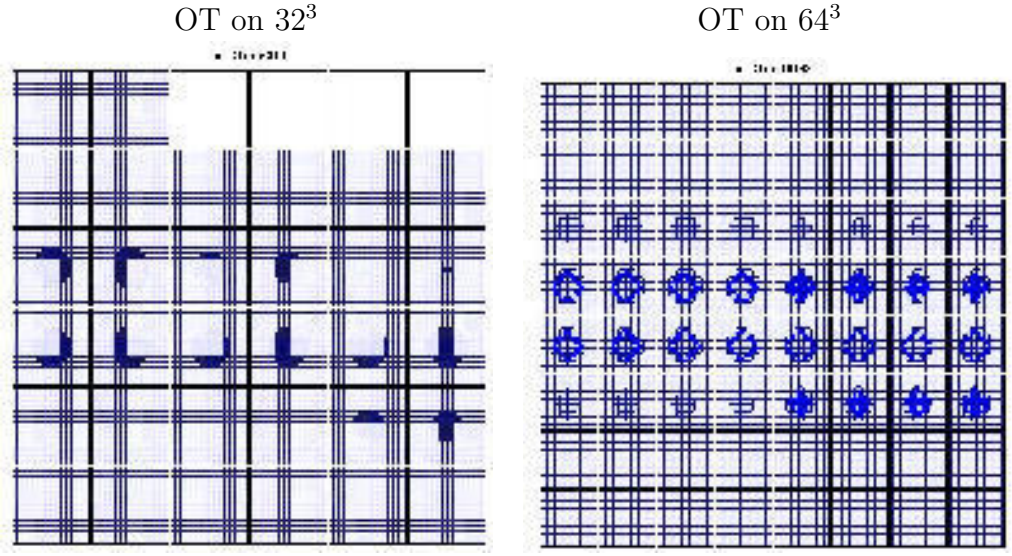



Figure 9: OcTree grids obtained for the second experiment.

In Table 2 we present the number of unknowns obtained on each level. The number of unknowns on the finest grid is roughly 12 times smaller than the number of unknowns we would obtain by using the full $64^3$ grid. Since the overhead of the OcTree is relatively small compared to the time required for the solution of the forward problem, we obtain an order of magnitude improvement in the computational time.

# 6 Discussion and Summary

In this paper we have developed an adaptive multilevel refinement method for the solution of distributed parameter estimation problems using a finite volume approach. Both forward and inverse problems are carefully discretized on OcTree grids, and a Tikhonov-type regularization with a Huber switching function on the magnitude of the model's gradient is applied.

We have experimented with the DC resistivity problem. Our experiments reveal that substantial computational savings can be obtained using the proposed multilevel adaptive refinement strategy. In particular, the total computational time can routinely be reduced by an order of magnitude compared to a uniform grid with the same resolution.

Most ingredients from which our algorithm is constructed may look very familiar at first, but their composition in the present context leads to potential surprises. In particular:

- the suitable grids one ends up using for the fields (solutions of the constraint equations (1.1b) or (1.2)) are not necessarily similar to those that would be suited for solving the forward problem in a stand-alone fashion;
- the suitable grids one ends up composing for the model can improve the reconstructed model without necessarily improving the fit to the data;
- a good choice of the Huber switching parameter $\gamma$ is more suitable, in principle as well as in practice, than either the least squares or total variation options from which (2.9) is constructed;
- contrary to common folklore, detailed models may better be reconstructed on locally fine grids, where coarse uniform grids are not sufficient, even though the forward problem is diffusive.

The latter point is particularly interesting. It demonstrates that using adaptive grid refinement one is able to learn data vs regularization driven features in the model. If one obtains a satisfactory data misfit on a coarse grid then we conclude that any further local resolution is obtained by the regularization operator. Thus, adaptation can be beneficial also as a tool to study the resolution of the problem.

Finally, we have shown that care should be exercised when choosing the initial coarse grid. For the particular application we have here, starting with too coarse a grid can be a disadvantage.

Next, we plan to combine our grid refinement tools with level sets for shape optimization.

# References


[1] A. Alessandrini and S. Vessella. Lipschitz stability for the inverse conductivity problem. *Adv. Appl. Math.*, 35:207–241, 2005.

[2] U. Ascher and E. Haber. Grid refinement and scaling for distributed parameter estimation problems. *Inverse Problems*, 17:571–590, 2001.

[3] U. Ascher and E. Haber. A multigrid method for distributed parameter estimation problems. *ETNA*, 15:1–15, 2003.

[4] U. Ascher, E. Haber, and H. Haung. On effective methods for implicit piecewise smooth surface recovery. *SIAM J. Scient. Comput.*, 28:339–358, 2006.

[5] U. Ascher, R. Mattheij, and R. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. SIAM, Philadelphia, 1995.

[6] W. Bangerth. Adaptive finite element methods for the identification of distributed coefficients in partial differential equations. *Ph.D Thesis*, University of Hiedelberg:Germany, 2002.

[7] R. Becker. Adaptive finite element methods for optimal control problems. *Habilitation Thesis*, University of Hiedelberg:Germany, 2001.

[8] R. Becker, H. Kapp, and R. Rannacher. Adaptive finite element methods for optimal control of partial differential equations. *SIAM J. Control Optim.*, 39:113–132, 2000.

[9] A. Borzi and K. Kunisch. Optimal contol formulation for determining optical flow. *SIAM J. Scient. Comput.*, 24:818–847, 2002.

[10] M. Burger and S. J. Osher. A survey on level set methods for inverse problems and optimal design. *European J. Appl. Math.*, 16:263–301, 2005.

[11] M. Cheney, D. Isaacson, and J.C. Newell. Electrical impedance tomography. *SIAM Review*, 41:85–101, 1999.

[12] J. Claerbout and F. Muir. Robust modeling with erratic data. *Geophysics*, 38:826–844, 1973.

[13] L. J. Durlofsky. Coarse scale models of two phase flow in heterogeneous reservoirs: Volume averaged equations and their relationship to existing upscaling techniques. *Computational Geosciences*, 2:73–92, 1998.

[14] M. Edwards. Elimination of adaptive grid interface errors in the discrete cell centered pressure equation. *J. of Comp. Phys.*, 126:356–372, 1996.

[15] R.E. Ewing, R.D. Lazarov, and P.S. Vassilevski. Local refinement techniques for elliptic problems on cell-centered grids I, error analysis. *Math. Comp.*, 56:437–461, 1991.

[16] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–223, 1979.

[17] E. Haber. A multilevel, level-set method for optimizing eigenvalues in shape design problems. *J. Comp. Phys.*, 198:518–534, 2004.

[18] E. Haber. Quasi-newton methods methods for large scale electromagnetic inverse problems. *Inverse Peoblems*, 21, 2005.

[19] E. Haber and U. Ascher. Preconditioned all-at-one methods for large, sparse parameter estimation problems. *Inverse Problems*, 17:1847–1864, 2001.

[20] E. Haber, U. Ascher, and D. Oldenburg. On optimization techniques for solving nonlinear inverse problems. *Inverse problems*, 16:1263–1280, 2000.

[21] E. Haber, U. Ascher, and D. Oldenburg. Inversion of 3d electromagnetic data in frequency and time domain using an inexact all-at-once approach. *Geophysics*, 69:1216–1228, 2004. n5.

[22] G. R. Hjaltason and H. Samet. Speeding up construction of QuadTrees for spatial indexing. *The VLDB Journal*, 11:109–137, 2002.

[23] V. Isakov. *Inverse Problems for Partial Differential Equations*. Springer, 2006.

[24] S. Knapek. Matrix-dependent multigrid homogenization for diffusion problems. *SIAM J. Sci. Comp.*, 20(2):515–533, 1999.

[25] K. Lipnikov, J. Morel, and M. Shashkov. Mimetic finite difference methods for diffusion equations on non-orthogonal AMR meshes. *J. Comp. Phys.*, 199:589–597, 2004.

[26] J. Liu. A multiresolution method for distributed parameter estimation. *SIAM J. Scient. Comp.*, 14:389–405, 1993.

[27] F. Locasso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM trans. on Graphics*, Siggraph:457–462, 2004.

[28] F. Losasso, R. Fedkiw, and S. Osher. Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids*, 35:457–462, 2006.

[29] P. R. McGillivray. *Forward Modelling and Inversion of DC Resistivity and MMR Data*. PhD thesis, University of British Columbia, 1992.

[30] J. Nocedal and S. Wright. *Numerical Optimization*. New York: Springer, 1999.

[31] P.O. Persson and G. Strang. A simple mesh generator in matlab. *SIAM Review*, 46 (2):329–345, 2004.

[32] A. Pidlisecky, E. Haber, and R. Knight. Cone-based electrical resistivity tomography. *Geophysics*, 71, n4:157–167, 2006.

[33] A. Pidlisecky, E. Haber, and R. Knight. RESINVM3D: A MATLAB 3-D resistivity inversion package. *Geophysics*, To appear, 2006.

[34] N.C. Smith and K. Vozoff. Two dimensional DC resistivity inversion for dipole dipole data. *IEEE Trans. on geoscience and remote sensing*, GE 22:21–28, 1984. Special issue on electromagnetic methods in applied geophysics.

[35] U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, 2001.

[36] K. van den Doel and U. Ascher. On level set regularization for highly ill-posed distributed parameter estimation problems. *J. Comp. Phys.*, 216:707–723, 2006.

[37] C. Vogel. *Computational methods for inverse problem*. SIAM, Philadelphia, 2001.