Adaptive finite volume method for distributed non-smooth parameter identification

Eldad Haber* Stefan Heldmann † Uri Ascher ‡

June 21, 2007

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 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^{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 β 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. [15, 5].

Numerical optimization problems of this kind are often solved in geophysics [23], medical physics [14], computer vision [12] and other fields which involve data fitting. To be more

*This work is supported by NSF grant CCF-0427094 and DOE grant DE FG02-05ER25696

†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 [37, 32, 36] and references therein) the elliptic problems associated with

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

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. [25]).

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 [5, 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 [19, 39, 13]. 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 [3, 4, 19]. Such an approach is particularly beneficial in this context because it enables us to obtain a good approximation to the regularization parameter, β, 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 [3]. 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 function 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 [34]). 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. [24, 30], 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 [18, 17, 28].

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 [7]. Similar work in the context of finite elements is reported in [8, 9, 2, 26, 10]. Some work using finite differences is reported in [11]; see also [29, 42] 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 facilitates "zooming 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 a finite volume discretization on OcTrees for the forward and inverse problems, as well as the choice of the function ρ 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 perform numerical experiments that demonstrate the advantage of our approach. The latter two are the major sections of this article: we show that rather significant gains can be made provided that the regularization operator is adequately formulated and evaluated. Finally, in Section 6 we summarize our findings.

2 Problem discretization

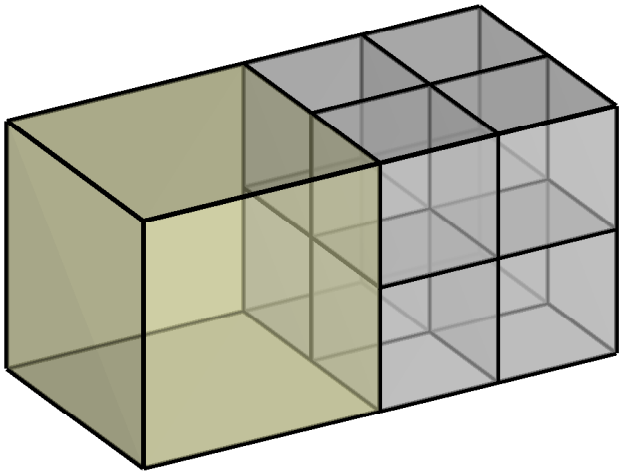
In this section we discuss the discretization of the problem (1.1). As in [3] we envision a uniform underlying coarse orthogonal grid with cell width h_c and a uniform underlying fine orthogonal grid of size $2^{N_1} \times 2^{N_2} \times 2^{N_3}$ with cell width $h_f = 2^{-nlevel} h_c$. The fine grid 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 [3] 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 forward and inverse problems on it. While the discretization of the forward problem on OcTrees is common using finite differences (see for example [38]), we take here a different, variational approach, mimicking finite elements and arriving at a finite volume method. This is well

known on uniform grids, but the details for more general grids, particularly those that can be implemented by an OcTree, do not appear to be available, so a short exposition is warranted. The discretization of the inverse problem is then developed in a similar manner.

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. Each entry in the array is the cell size.

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 [24]. 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 [3] we choose to use staggered discretization and discretize u at the nodes and m at cell centers. As shown in [4] this discretization of the optimization problem is h-elliptic. For the importance of h-elipticity, see [38].

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.

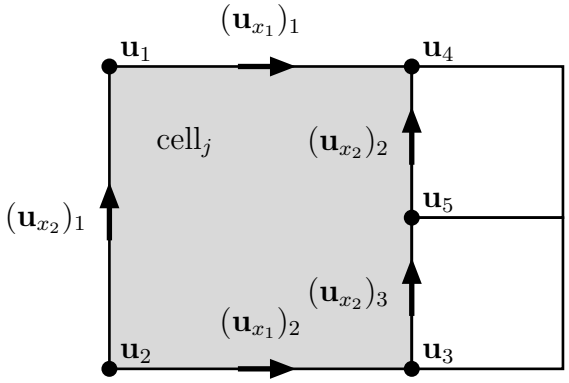


Figure 2: Discretization of $\nabla \mathbf{u}$

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 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 σ 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

$$\begin{aligned}
\sigma_{\text{cell}} \int_{\text{cell}} (u_{x_1})^2 \, d\mathbf{x} &= \frac{\sigma_{\text{cell}} V_{\text{cell}}}{8h^2} \left((u_4 - u_1)^2 + (u_3 - u_2)^2 \right) + \mathcal{O}(h^2), \\
\sigma_{\text{cell}} \int_{\text{cell}} (u_{x_2})^2 \, d\mathbf{x} &= \frac{\sigma_{\text{cell}} V_{\text{cell}}}{8h^2} \left(\frac{1}{2}(u_4 - u_5)^2 + \frac{1}{2}(u_5 - u_3)^2 + (u_1 - u_2)^2 \right) + \mathcal{O}(h^2),
\end{aligned}$$

where V_{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

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

where $\mathbf{u}$ and $\mathbf{b}$ are nodal OcTree grid functions, σ 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

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

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

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

Note that given a particular grid, the matrices GRAD, $\mathbf{A}_n^c$ and $\mathbf{A}_e^c$ are computed only once. We can then reuse them for different grid functions σ 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 gradient 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 centered 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 [18] and recently used in [31].

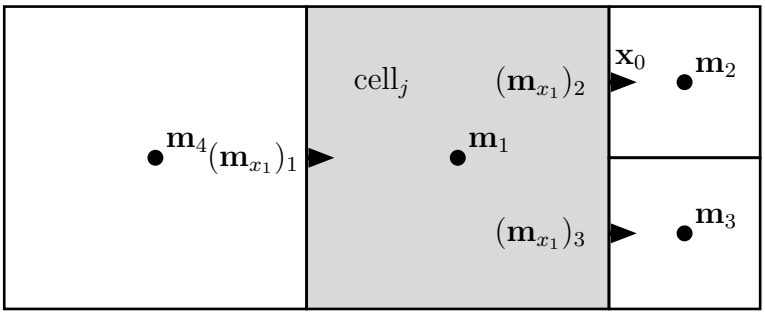


Figure 3: Discretization of ∇m

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 [6]. But here the resulting approximation 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. Approximations to the gradient of m in any other direction are obtained in a similar way.

Next we replace the integral of $\rho(|\nabla m|)$ by its approximation at the cell center multiplied by the cell volume

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

Once again, we rewrite the expression using matrices and vectors. For this, let 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

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

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 γ

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

The parameter γ was selected adaptively in [5] 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 γ depending on the finest expected grid scale. We do not modify ρ during the grid refinement process.

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

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

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

$$\mathbf{v}^\top\mathbf{u}_j = 0, \quad j = 1, \ldots, n. \tag{2.10c}$$

3 Solving the optimization problem

In this Section we quickly review solution techniques for the optimization problem. For a more in-depth discussion see [22].

We use variants of Reduced Hessian Sequential Quadratic Programming. 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 [40, 5].

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 [21, 20].

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 obtained by substituting the fields $\mathbf{u}_j$ and the Lagrange multiplier functions $\mathbf{p}_j$ 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 evaluate $\mathbf{g}$, observe that we can break the computation over each term in the sum. Starting from $\mathbf{g}=\mathbf{0}$, assume that the k^{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 completes the evaluation of the reduced gradient.

Unfortunately, one must store $\mathbf{u}_j$ when a product of the reduced Hessian and a vector is evaluated. 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, it is possible to effectively use Quasi-Newton techniques without the necessity of excessive storage. In our applications we often need to deal with a very large number of sources. We therefore use the reduced space L-BFGS [33] approach.

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 good, low-cost 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 too large to be acceptable. Unfortunately, finding a unique refinement criterion that works for different problems is rather difficult (e.g. [38]). As we see next, further complications can arise from the fact that we solve a constrained optimization problem with noisy data.

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 $\mathbf{u}_j$, $\mathbf{m}$ and $\mathbf{p}_j$. At least in principle, it is possible to 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, this may give rise to other difficulties.

- If the grids for the fields $\mathbf{u}_j$ and for the Lagrange multipliers $\mathbf{p}_j$ are not identical then the forward operator A is not the adjoint of the discrete operator for the Lagrange multiplier. This implies a loss of symmetry in the saddle point equations. Indeed, it may happen that no discrete objective function is decreased, thus causing even simple unconstrained optimization algorithms to potentially 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 [16, 27]. 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 relatively flat 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 flat, 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.

In the next subsections we discuss our refinement criteria for each of the variables.

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, ζ and ν. Since the discretization for $\mathbf{m}$ is based on the evaluation of integrals in an optimization functional, we derive the refinement based on the estimated errors in those integrals.

Assume first that $m(\mathbf{x})$, and therefore $\rho(|\nabla m(\mathbf{x})|)$, are known functions. Then, for the solution of the inverse problem we need to estimate the integral

$$\int_{\Omega_j} \rho \, d\mathbf{x},$$

where Ω_j is an OcTree cell centered at $\mathbf{x}_j$.

Let us use the shorthand $\hat{\rho}(\mathbf{x}) = \rho(|\nabla m(\mathbf{x})|)$. We can first write

$$\begin{aligned} \int_{\Omega_j} \hat{\rho}(\mathbf{x}) \, d\mathbf{x} &= \int_{\Omega_j} \left[\hat{\rho}(\mathbf{x}_j) + \nabla \hat{\rho}(\boldsymbol{\xi}(\mathbf{x}))^\top (\mathbf{x} - \mathbf{x}_j) \right] \, d\mathbf{x} \\ &= V_j \hat{\rho}(\mathbf{x}_j) + \int_{\Omega_j} \nabla \hat{\rho}(\boldsymbol{\xi}(\mathbf{x}))^\top (\mathbf{x} - \mathbf{x}_j) \, d\mathbf{x}, \end{aligned}$$

where $\boldsymbol{\xi}$ is a point within the domain Ω_j and V_j is the volume of the cell. Thus, we can estimate the integral by $V_j \hat{\rho}(\mathbf{x}_j)$ and bound the error of integration at each OcTree cell by

$$\left| \int_{\Omega_j} \hat{\rho} \, d\mathbf{x} - V_j \hat{\rho}(\mathbf{x}_j) \right| \leq V_j \max_{\mathbf{x} \in \Omega_j} \|\mathbf{x} - \mathbf{x}_j\| \max_{\mathbf{x} \in \Omega_j} |\nabla \hat{\rho}(\mathbf{x})| = \frac{\sqrt{3}}{2} h_j V_j \max_{\mathbf{x} \in \Omega_j} |\nabla \hat{\rho}(\mathbf{x})|, \qquad (4.13)$$

where h_j is the length of the cell edge.

From the error expression we see that if $|\nabla \rho(|\nabla m|)|$ is large then the integral is inaccurate. Since the solution of the optimization problem depends on the accurate evaluation of the integral we would like to refine in areas that $|\nabla \rho(|\nabla m|)|$ is large. To estimate it we use the computed m, denoted m_h. Thus, in areas where $|\nabla_h \rho(|\nabla_h m_h|)| > \zeta$ we refine the grid. In other areas the approximation to ρ is relatively flat so no further refinement is needed. The refinement process is terminated when $|\nabla_h \rho(|\nabla_h m_h|)| \leq \zeta$ for all cells.

Note that ζ bounds the truncation error in our approximation of the regularization term. If we allow a large ζ then the evaluation of the regularization term could be rather inaccurate. If, on the other hand, we choose ζ too small then the integral evaluation is accurate but may not benefit the overall estimation of $\mathbf{m}$.

The other parameter, ν, checks convergence with respect to successive $\mathbf{m}$ refinements, see Algorithm 1.

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 $\mathbf{u}_j$ grids given the $\mathbf{m}$ grid.

There are two different but related issues when designing an appropriate refinement criterion to the $\mathbf{u}$ grid. For the optimal model, m^*, and a $\mathbf{u}$ grid S, we can compute the simulated data $\mathbf{d}(S, m^*)$. We can then write

$$\mathbf{d} - \mathbf{d}(S, m^*) = \text{measurement noise} + \text{numerical noise}.$$

While we are able in principle to control the numerical noise, the measurement noise is part of the given problem.

If the measurement noise level in the data is not very small then the discretization for u on the grid S should approximate the data well below the noise level. That is, given $\xi < 1$, at the solution, the $\mathbf{u}$-grid is required to satisfy

$$\|\mathbf{d}(S, m) - \mathbf{d}\| \leq \xi \|\text{measurement noise}\|. \tag{4.14}$$

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 $\mathbf{u}$ grid width must go to 0.

Assume now that the numerical noise dominates the measurement noise, hence the $\mathbf{u}$ grid should be refined. We now suggest an approach to the discretization of the u-grid that allows a gradual improvement of the misfit.

Assume that at the k^{th} iteration we have a model m_k and its discrete approximation $\mathbf{m}_k$. Let $m^* = m_k + \delta m$, let S_k represent the grid for u and p, and let $A(\mathbf{m}_k; S_k)$ be the discretization of the forward problem for $\mathbf{m}_k$ on S_k. Further, assume for simplicity that the number of sources is $n = 1$ and that Q is the identity in (1.1a). Finally, let $\mathbf{d}(\mathbf{m}_k, S_k) = A(\mathbf{m}_k; S_k)^{-1}\mathbf{b}$. We can write

$$\delta\mathbf{d} = \mathbf{d} - \mathbf{d}(m_k) = \delta\mathbf{d}(S_k) + (\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)),$$

where $\delta\mathbf{d}(S_k) = \mathbf{d} - \mathbf{d}(\mathbf{m}_k, S_k)$ and $\mathbf{d}(m_k)$ is the simulated data for the continuous model m_k (i.e. the data obtained without numerical errors). Since we evaluate our solution on the grid S_k the term $\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)$ can be interpreted as the numerical noise in our

evaluation of the data difference $\delta\mathbf{d}$. Thus, if $\|\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)\| > \|\delta\mathbf{d}(S_k)\|$ we have added a substantial amount of highly correlated noise to the problem and there is no chance to recover a good approximation to δm. Therefore, our strategy for the choice of the grid for u and p is to require that

$$\|\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)\| \leq \xi\|\delta\mathbf{d}(S_k)\|. \tag{4.15}$$

Equation (4.15) is similar to (4.14) and it requires that on the current grid we are able to evaluate a meaningful approximation to the simulated data. Evaluating the left hand side of (4.15) may not be trivial. In a few cases $\mathbf{d}(m_k)$ is known analytically, but in most cases it requires the numerical evaluation of u given m_k, which can be done by standard adaptive grid refinement techniques for the forward problem (see for example [38] and references therein).

We summarize our approach for adaptive grid refinement in Algorithm 1.

Algorithm 1 Adaptive Grid Refinement

given m_0 evaluate $\mathbf{m}_0$ and the m grid S_0^m;
set $\xi, \zeta, \nu < 1$
for $k = 0, 1, 2, \ldots$ **do**
  evaluate S_k such that

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

  if $\mathbf{m}_k$ is an acceptable model **and** $\|\mathbf{d}(\mathbf{m}_k, S_k) - \mathbf{d}(m_k)\| \leq \xi\|\text{noise}\|$ **then**
    return
  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**
    return
  end if
  given $\mathbf{m}_{k+1}$ obtain S_{k+1}^m with tolerance ζ
end for

4.3 Discussion

Unfortunately, our multilevel refinement requires three parameters. However, these parameters have a rather intuitive meaning.

ζ: Controls the truncation error in the discretization of the regularization operator. In many cases this parameter can be estimated by a priori knowledge about the model we are after.

ξ: Controls the error level in the approximated data. If the data contains large amount of noise then setting ξ below the noise level is desired. In (mainly synthetic) cases where the noise level is negligible, setting ξ to the finest discretization error prevents attempting to obtain computed data that are too accurate.

ν: Controls the change in the recovered model. In most cases it is easy to pick this parameter by understanding the dynamical range of the model. For example, in geophysical setting where m is the log conductivity a change of less than 0.1 is considered not meaningful.

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* 128^3 *uniform grid* and add 1% Gaussian noise to it. To avoid "inverse crimes" we allow in the sequel only grid refinements of up to 64^3. We also set $\gamma = .01$ in (2.9) for all experiments, which is commensurate with the width h of the finest grid. We then use Algorithm 1 to recover the model. Unless stated otherwise the following choice of parameters is employed: $\xi = 0.1, \nu = 0.05, \zeta = 0.01$. For each optimization problem on different grids, the optimization process terminates 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. The experiments are conducted on a 2.5 intel GHz processor with 2Gb of RAM.

5.1 Experiment I

In our first experiment we use 4 dipole sources located at

$$\begin{aligned} s_1^+ &= [0.1.0.1.0.1],\ s_1^- = [0.9, 0.9, 0.9] \\ s_2^+ &= [0.1.0.9.0.1],\ s_2^- = [0.9, 0.1, 0.9] \\ s_3^+ &= [0.9.0.1.0.1],\ s_3^- = [0.1, 0.9, 0.9] \\ s_4^+ &= [0.9.0.9.0.1],\ s_4^- = [0.1, 0.1, 0.9]. \end{aligned}$$

We sample using 1000 points equally spaced in $[0.2, 0.8]^3$. Here, the model has a 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.

5.1.1 Experiment I - regular grid

In order to have a point of comparison to the adaptive grid refinement we solve the problem using the method described in [4] on uniform grids. Our coarsest grid is set to 16^3 for the u, p grids and to 8^3 for the m grid. Our finest grid is 64^3 for all unknowns. We then save the final model obtained and record the CPU time to be compared against adaptive grid techniques.

5.1.2 Experiment I - adaptive grid

Here the grid for u and p is set by choosing a uniform 16^3 grid as the coarsest one and refining twice close to the sources. When comparing the data obtained on this first grid with a half space data [41] we find an agreement to roughly 0.1%, 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 the model m. The resulting m on different grids is presented in Figure 4. Interestingly, each of the models within the multilevel process fits the data to within 1% error. Thus, from a data fitting point of view all these models are valid! Nevertheless, while the 8^3 grid does not yield a satisfactory result, locally refining it, 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. For this example, the information provided by the regularization gives a more accurate reconstruction of the model we are after. In Figure 4 we also plot the recovered model obtained on a uniform 64^3 grid. It is evident that the finest OcTree model gives virtually the same result. When comparing the 2 norm of the interpolated OcTree model to the one obtained on uniform grids we find that

$$h^3 \|\mathbf{m}_{\text{uniform}} - \mathbf{m}_{\text{OcTree}}\| = 8.2 \times 10^{-3}.$$

Thus, our locally refined grid has allowed good approximation to the results obtained on the uniform 64^3 grid.

Comparing the computational effort, we find that for this reconstruction the 64^3 uniform grid requires roughly 397 minutes while the adaptive mesh refinement takes only 25 CPU minutes.

The OcTree for m for different levels is plotted in Figure 5. 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!

5.1.3 Experiment I - adaptive grid with different parameters

In this set of experiments we test the sensitivity of our results to the different parameters of the adaptive algorithm. We vary all the parameters and record the number of cells and the difference between the solutions obtained at the finest level. When doing so we vary one parameter and leave the rest fixed in their default values. The results are displayed in Table 2.

We now discuss the results for each parameter.

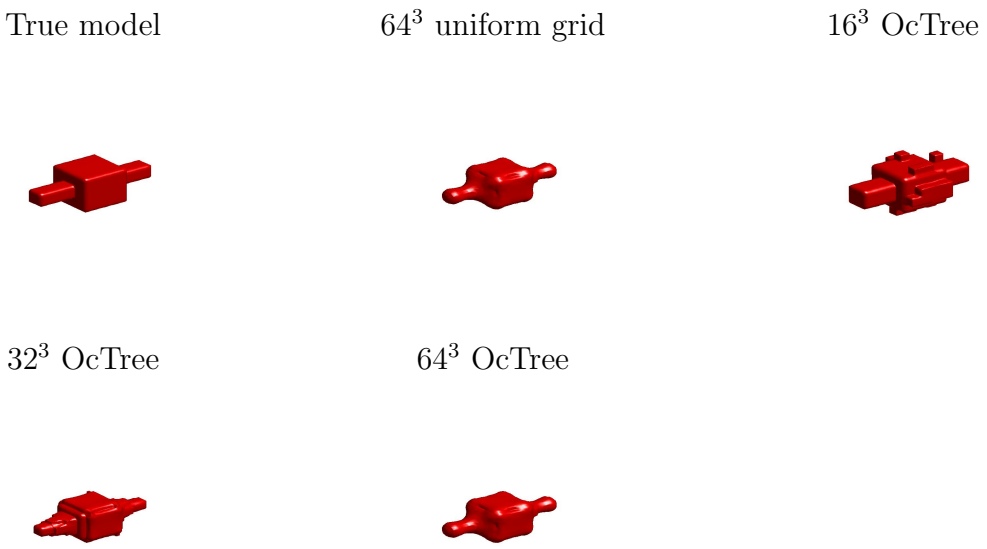


Figure 4: True and Recovered models on different grids.

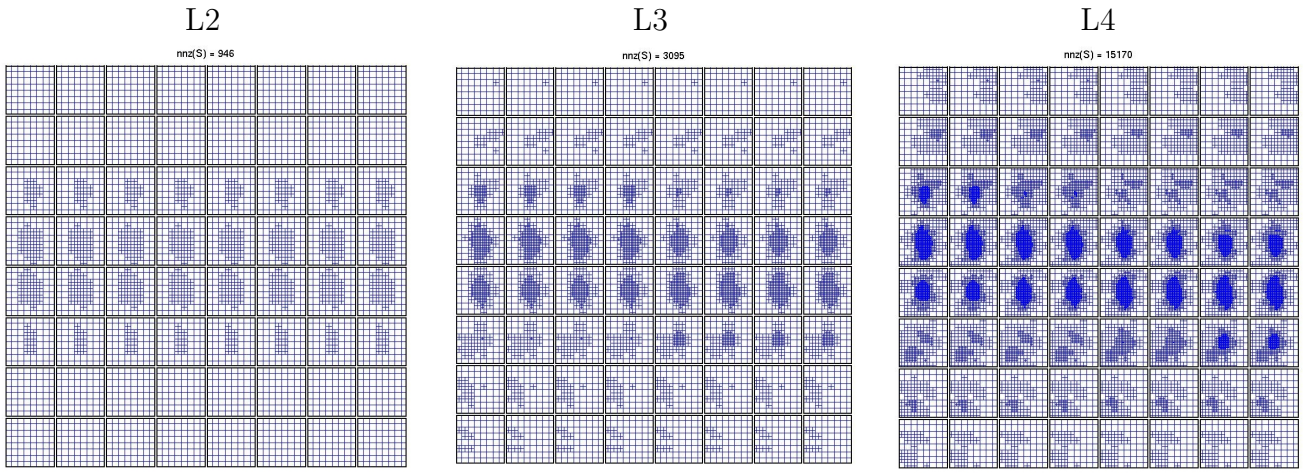


Figure 5: Grids for m at different levels. The 3D grid is visualized by slicing it along the z axis. The grid on the first level is equidistant.

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

ζ	$\|\mathbf{m}-\mathbf{m}^*\|$	u grid	m grid
10^{-1}	6.4×10^{-2}	13312	5194
10^{-2}	8.2×10^{-3}	19547	15170
10^{-3}	1.1×10^{-3}	33682	27322
10^{-4}	2.7×10^{-4}	62856	42257
ξ	$\|\mathbf{m}-\mathbf{m}^*\|$	u grid	m grid
10^{-1}	8.2×10^{-3}	19547	15170
10^{-2}	7.9×10^{-3}	21332	16005
10^{-3}	7.9×10^{-3}	21332	16005
10^{-4}	7.9×10^{-3}	21332	16005
ν	$\|\mathbf{m}-\mathbf{m}^*\|$	u grid	m grid
10^{-1}	9.2×10^{-3}	14322	12180
10^{-2}	8.3×10^{-3}	24357	16172
10^{-3}	7.8×10^{-3}	28981	19345
10^{-4}	7.8×10^{-3}	28981	19345

Table 2: Sensitivity of the results with respect to the different parameters

- **Sensitivity with respect to ζ:** This parameter controls the accuracy of the discretized regularization and therefore the smaller it is, the finer the m grid is. We observe that as the m grid gets refined the error in the model is smaller. However, for most practical applications a very accurate representation of m^* is not required. This is because the optimal model, m^*, is rarely a very good approximation to the true model, so only a rough estimation of this model is needed.

- **Sensitivity with respect to ξ:** This parameter controls the discretization of u such that the simulated data is sufficiently accurate. Note that when $\xi < 10^{-2}$ we obtain the same results as for $\xi = 10^{-2}$. This is because although we have more information in the data, it is insufficient to generate changes in the recovered models.

- **Sensitivity with respect to ν:** This parameter terminates the iteration if the difference between models on two consecutive OcTrees is small. As expected, when ν is smaller more cells are required. However, for $\nu < 10^{-3}$ we obtain the same results as for $\nu = 10^{-3}$. This is because $|\nabla\rho|$ was smaller than ζ.

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 [35], 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 given in Figure 6.

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

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

where $-3 \leq \{x, y, z\} \leq 3$ and $a = 500$.

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 7 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.

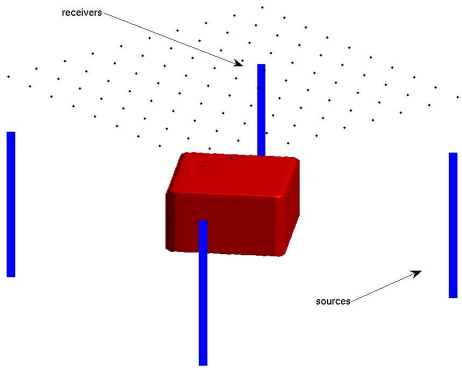


Figure 6: The model and the experimental setting of our second experiment. The model is a block tilted in a 45 degree angle with respect to the grid. The sources are placed in boreholes and the receivers on a plain above.

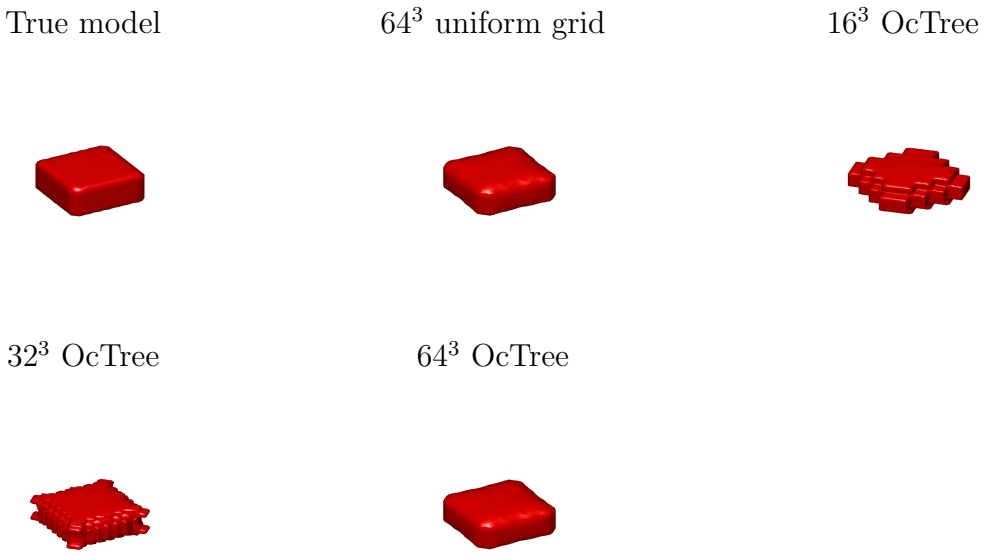


Figure 7: True and Recovered models on different grids.

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

Table 3: Number of unknowns on different levels

In Figure 8 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.

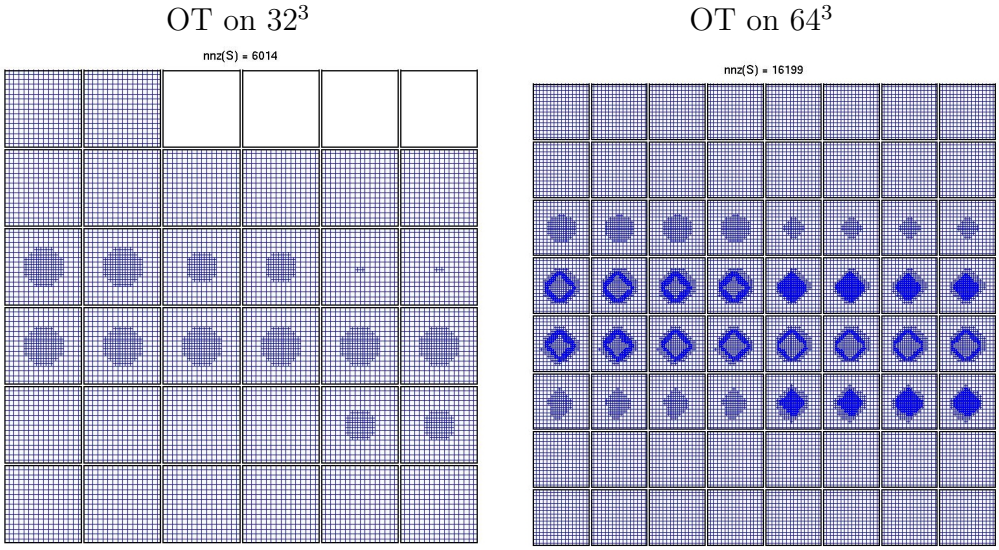


Figure 8: OcTree grids for m obtained for the second experiment. The 3D grids for the model are visualized by slicing along the z axis.

In Table 3 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.

To evaluate the computational saving obtained using adaptive mesh we solve the problem on a uniform 64^3 grid and compare the results. Once again, the error in the recovered model is very small

$$h^3 \|\mathbf{m}_{\text{uniform}} - \mathbf{m}_{\text{OcTree}}\| = 1.1 \times 10^{-2}.$$

The computational saving here are significant: while the computation on the full grid took roughly 123 hours, the computation on the OcTree grid took only 13 hours.

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 we note the following:

- 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 γ 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.

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] H. Ben Ameur, G. Chavent, and J. Jaffré. Refinement and coarsening indicators for adaptive parametrization: application to the estimation of hydraulic transmissivities. *Inverse Problems*, 18:775–794, 2002.

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

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

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

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

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

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

[9] 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.

[10] L. Beilina and C.R. Johnson. A posteriori error estimation in computational inverse scattering. *Mathematical Models & Methods In Applied Sciences*, 15:23–35, 2005.

[11] L. Borcea and V. Druskin. Optimal finite difference grids for direct and inverse Sturm Liouville problems. *Inverse Problems*, 18:979–1001, 2002.

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

[13] 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.

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

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

[16] 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.

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

[18] 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.

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

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

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

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

[23] 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.

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

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

[26] C.R. Johnson and R.S. MacLeod. Adaptive local regularization methods for the inverse ECG problem. *Progress in Biophysics & Molecular Biology*, 69:405–423, 1998.

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

[28] 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.

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

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

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

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

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

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

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

[36] A. Pidlisecky, E. Haber, and R. Knight. RESINVM3D: A MATLAB 3-D resistivity inversion package. *Geophysics*, 72(2):H1–H10, 2007.

[37] 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.

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

[39] 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.

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

[41] S.H. Ward and G.W. Hohmann. Electromagnetic theory for geophysical applications. *Electromagnetic Methods in Applied Geophysics*, 1:131–311, 1988. Soc. Expl. Geophys.

[42] N. Yan and W.B. Liu. A posteriori error estimates for control problems governed by nonlinear elliptic equations. *Applied Numerical Mathematics*, 47:173–187, 2003.