Large-scale stochastic linear inversion using hierarchical matrices

Illustrated with an application to crosswell tomography in seismic imaging

Sivaram Ambikasaran* Judith Yue Li† Peter K Kitanidis*† Eric Darve *‡

Received: date / Accepted: date

Stochastic inverse modeling deals with the estimation of functions from sparse data, which is a problem with a non-unique solution, with the objective to evaluate best estimates, measures of uncertainty, and sets of solutions that are consistent with the data. As finer resolutions become desirable, the computational requirements increase dramatically when using conventional solvers. A method is developed in this paper to solve large-scale stochastic linear inverse problems, based on the hierarchical matrix (or $\mathcal{H}^2$ matrix) approach. The proposed approach can also exploit the sparsity of the underlying measurement operator, which relates observations to unknowns. Conventional direct algorithms for solving large-scale linear inverse problems, using stochastic linear inversion techniques, typically scale as $\mathcal{O}(n^2m + nm^2)$, where n is the number of measurements and m is the number of unknowns. We typically have $n \ll m$. In contrast, the algorithm presented here scales as $\mathcal{O}(n^2m)$, i.e., it scales linearly with the larger problem dimension m. The algorithm also allows quantification of uncertainty in the solution at a computational cost that also grows only linearly in the number of unknowns. The speedup gained is significant since the number of unknowns m is often large. The effectiveness of the algorithm is demonstrated by solving a realistic crosswell tomography problem by formulating it as a stochastic linear inverse problem. In the case of the crosswell tomography problem, the sparsity of the measurement operator allows us to further reduce the cost of our proposed algorithm from $\mathcal{O}(n^2m)$ to $\mathcal{O}(n^2\sqrt{m} + nm)$. The computational speedup gained by using the new algorithm makes it easier, among other things, to optimize the location of sources and receivers, by minimizing the mean square error of the estimation. Without this fast algorithm, this optimization would be computationally impractical using conventional methods.
Keywords. Stochastic inverse modeling Numerical linear algebra Hierarchical matrices Large-scale problems Subsurface imaging Tomography Geostatistical estimation

1 Introduction

Large-scale stochastic inverse modeling has been drawing substantial attention in recent times, for example in the context of subsurface modeling. Inverse modeling is an essential ingredient in subsurface modeling due to the fact that direct measurements of hydrogeologic parameters such as conductivity and specific storage and variables such as pressure and solute concentration are expensive and sometime impossible to obtain. Engineering applications, such as remediation, real-time monitoring of CO_2 sequestration sites, deep-well injection projects and hydrocarbon reservoir exploitation, demand ever more accurate prediction of the unknowns such as hydraulic conductivity, hydraulic head, solute concentrations, and others.

Such inverse problems are ill-posed because they involve few measurements and a large number of unknowns and require solving underdetermined linear systems. A popular technique that has found numerous applications in the context of subsurface imaging for solving such underdetermined inverse problems is the Bayesian geostatistic approach [12, 34, 35, 37–39, 49, 51]. The motivation behind this approach is to combine the data obtained through measurements with a stochastic model of the structure of the function to be estimated. The structure of the underlying function is characterized through an a priori given probability density function. The prior probability density function is parameterized through variograms and generalized covariance functions. The approach also accounts for the fact that the measurements include errors and thus must not be reproduced exactly. One of the advantages of this approach is that it not only gives the "best" estimate, but also allows

*Institute for Computational and Mathematical Engineering, Stanford University
†Department of Civil and Environmental Engineering, Stanford University
‡Department of Mechanical Engineering, Stanford University

us to quantify the uncertainty through confidence intervals or through functions that are samples from the posterior distribution (conditional realizations).

However, a major hurdle of this method is that, when the number of unknowns m to be estimated is large, the method becomes computationally expensive. The bottleneck arises due to the fact that storage and matrix operations involving large dense covariance matrices scale as $\mathcal{O}(m^2)$, making application of the approach next to impossible for large-scale problems.

There have been various approaches to circumvent this difficulty. For instance, [21, 43, 50] make use of the fast Fourier transform to address this problem on a regular grid. The fast Fourier transform takes advantage of the fact that for a regular grid the covariance matrix has a Toeplitz or block-Toeplitz structure and this structure can be exploited to construct fast matrix vector products in $\mathcal{O}(m \log m)$. However, one of the drawbacks of the method is that its extension to other grids is non-trivial and most often in applications, such as when using the finite-element approach, we rely on non-uniform unstructured grids.

To deal with unstructured grids, [40] proposes an algorithm that relies on functional parameterization of the spatial random field by the Karhunen-Loève (KL) expansion. The spatial random field is parameterized by weighted base functions that are derived from the covariance function. The KL expansion relies on the assumption that the covariance function is smooth and that the correlation length is large compared to the size of the domain. In terms of matrix algebra, the KL expansion can be interpreted as approximating the large dense covariance matrix as a low-rank matrix. One of the drawbacks of this method is that the assumption that the covariance matrix can be approximated by a low-rank matrix is not always valid.

Recently, [42] circumvented the enormous computational cost by reformulating the problem under certain assumptions and by employing a sparse formulation for the generalized inverse of the covariance matrix, so that there is no longer a need for using the large dense covariance matrix.

In contrast, in this article, we propose an algorithm that takes advantage of the structure of large dense prior covariance matrices. Previously, computationally exploitable structure of these covariance and generalized covariance matrices, like isotropy and separability, have been studied in [61, 62]. However, the structure exploited by our proposed algorithm is a hierarchical low-rank structure, which is different from the aforementioned studies.

As stated earlier, the main stumbling block in the application of the geostatistical theory for inversion for large-scale problems is that the cost of dense matrix vector products increases quadratically with the number of unknowns m. In the past, the cost of performing dense matrix-vector products, especially in the context of boundary integral equations, have been reduced using fast summation techniques like the fast multipole method (FMM) [4, 10, 25, 48], the Barnes-Hut algorithm [2], panel clustering [31], FFT [11], wavelet based methods [7, 44, 45], etc. The literature on the FMM is vast and we refer the readers to [3, 4, 8, 10, 15, 16, 19, 26, 48, 60] for more details.

Over the last decade, the fast multipole method has been studied in terms of matrices known as hierarchical matrices. There are many different possible hierarchical structures, the most common of them being hierarchical semiseparable matrices [59], denoted as HSS, and hierarchical matrices [6, 24, 28–30], denoted as $\mathcal{H}$ and $\mathcal{H}^2$ matrices (where $\mathcal{H}^2$ requires somewhat stricter assumptions than $\mathcal{H}$, i.e., $\mathcal{H}^2 \subset \mathcal{H}$). The FMM has been recognized as a method that takes advantage of properties of $\mathcal{H}^2$ matrices to accelerate matrix-vector products. The generalized covariance matrices, that we consider, possess this underlying $\mathcal{H}^2$ structure. This is due to the fact the far field covariance can be well-represented by a smooth function. As a result, we can employ the fast multipole method proposed in [19] to compute matrix-vector products.

One of the main advantages of the fast multipole method discussed in [19] is that the algorithm is applicable to a wide range of covariance and generalized covariance functions. By using the $\mathcal{H}^2$ matrix structure, the computational cost of these dense matrix vector products are reduced from $\mathcal{O}(m^2)$ to $\mathcal{O}(m)$. Another desirable feature of our implementation is that our algorithm takes into account the sparsity of the measurement operator as discussed in section 3. This fact also plays a significant role in reducing the time taken to estimate the unknowns.

Our fast, new, stable algorithm for large-scale linear inversion problem relies on $\mathcal{H}^2$-matrix algebra to accelerate matrix-vector products. The algorithm is applicable when the number of unknowns m is large — around $100,000$ — and the number of measurements is of the order of 100–500. The algorithm is applied to a linear inversion problem arising out of a real crosswell tomography application as discussed in [14]. The computational speedup gained by our algorithm allows us to (i) quantify the uncertainty in the solution (ii) optimize the capture geometry. Quantifying uncertainty is crucial in computational modeling and simulation, especially in the context of inverse problems, to enable accurate assessment of the solution and to take decisions. The computational speedup gained also allows us to analyze different capture geometries and determine the optimal capture geometry. In most of the linear inversion problems, it is imperative to have a good capture geometry to get the maximum possible information for a given number of measurement devices. However, the optimization procedure involves assessing a sequence of different computational geometries. Due to the computational expense of conventional algorithms for stochastic linear inversion, analyzing a single capture geometry in itself

is a computationally intensive task. Using our algorithm, optimizing the capture geometry becomes tractable. We illustrate this by optimizing the capture geometry for a real crosswell tomography application [14].

To sum up, the major contributions of this paper are the following. The algorithm proposed by us results in a significant reduction in computational cost compared to the conventional algorithms, thus enabling the Bayesian geostatistic approach for large-scale problems. The computational speedup gained enables us to make new estimates such as quantifying the uncertainty in the solution and optimizing the capture geometry, which were previously inaccessible due to its enormous computational cost. We highlight the performance of our algorithm in section 5 by comparing our implementation with a conventional algorithm for a real crosswell tomography problem discussed in [14]. The fast algorithm enables us to provide an optimal capture geometry for this crosswell tomography problem [14]. To give a quantitative sense of the capability and to highlight the potential of the proposed algorithm, the proposed algorithm can solve a crosswell tomography problem (Section 3) with $250,000$ unknowns on an ordinary desktop in less than 20 minutes. The conventional algorithm on the other hand takes nearly 2 hours to solve a problem with just $40,000$ unknowns. The algorithm was implemented in C++ and is made available at `http://www.stanford.edu/~sivaambi/Fast_Linear_Inversion_PACKage.html`.

The remainder of the article is organized as follows. Section 2 introduces the preliminary ideas, i.e., stochastic linear inversion in subsection 2.1 and the hierarchical matrices in subsection 2.2 to solve large-scale stochastic linear inversion problems. Section 3 discusses the crosswell tomography problem. Section 4 presents a discussion of the general algorithm and its computational cost. This is then followed by Section 5, which explains how the algorithm discussed in Section 4 is applied to solve the crosswell tomography problem. It also presents the numerical results obtained by the algorithm both on synthetic and real data sets.

2 Preliminary ideas

In this section, we discuss the two key ingredients for large-scale stochastic linear inversion.

2.1 Stochastic linear inversion

The stochastic Bayesian approach is briefly introduced. This is a general method [37] to solve underdetermined linear systems arising out of linear inverse problems.

2.1.1 Prior

Let $s(x)$ be the function to be estimated at location x. Its "structure" is represented through the prior probability density function. The basic model of the function to be estimated is taken as

$$s(x) = \sum_{k=1}^{p} f_k(x)\beta_k + \epsilon(x). \tag{1}$$

The first term is the prior mean, where $f_k(x)$ are known functions, typically step functions, polynomials, and β_k are unknown coefficients where $k \in \{1, 2, \ldots, p\}$. The second term $\epsilon(x)$ is a random function with zero mean and characterized through a covariance function. In a sense, the first part is the deterministic part and the second part is the stochastic part. This model is especially popular in geostatistics and in other data analysis approaches. The *linear model* is represented by equation (1). For instance, the zonation/regression approach is obtained by setting

$$f_k = \begin{cases} 1, & \text{if in zone } k \\ 0, & \text{otherwise} \end{cases},$$

where $k \in \{1, 2, \ldots, p\}$, p being the number of zones. The covariance of ϵ is set to zero. In the regression approach, the variability is described through deterministic functions. In stochastic approaches, one describes at least some of the variability through the stochastic part (while still retaining flexibility in the use of the deterministic part).

After discretization (e.g., through application of finite-difference and finite element models), $s(x)$ is represented by a vector $s \in \mathbb{R}^m$. The mean of s is given by

$$\mathbb{E}[s] = \mathbf{X}\beta \qquad \text{(notation: } \mathbb{E} \text{ is the expectation)},$$

where $\mathbf{X} \in \mathbb{R}^{m \times p}$ is the drift matrix and $\beta \in \mathbb{R}^{p \times 1}$ are the p unknown drift coefficients. The covariance matrix of s is given by

$$\mathbb{E}\left[(s-\mu)(s-\mu)^T\right] = Q.$$

The entries of the covariance matrix are given by $Q_{ij} = K(x_i, x_j)$, where $K(\cdot, \cdot)$ is a generalized covariance function, which must be conditionally positive definite. For a more detailed discussion on permissible covariance kernels, we refer the reader(s) to the following references: [5, 32, 33, 36, 56].

2.1.2 Measurement equation

The observation/measurement is related to the unknown by the linear relation

$$y = Hs + v, \tag{2}$$

where $H \in \mathbb{R}^{n\times m}$ is the *observation/measurement matrix*, $v \in \mathbb{R}^{n\times 1}$ is a Gaussian random vector of observation/measurement error independent from s, with zero mean and covariance matrix R, i.e., $v \sim \mathcal{N}(0, R)$. Typically, the matrix R is a diagonal matrix since each measurement is independent of other measurements.

The form mentioned in equation (2) is encountered frequently in practice, because many important inverse problems are linear "deconvolution problems." Furthermore, many nonlinear problems are solved through a succession of linearized problems.

The prior statistics of y are obtained as shown below. The mean is given by

$$\mu_y = \mathbb{E}[Hs + v] = H\mathbb{E}[s] + \mathbb{E}[v] = HX\beta = \Phi\beta,$$

and the covariance is given by

$$\begin{aligned}\Psi &= \mathbb{E}\left[(H(s - X\beta) + v)(H(s - X\beta) + v)^T\right] \\ &= HQH^T + R.\end{aligned}$$

The y to s cross-covariance

$$C_{ys} = \mathbb{E}\left[(H(s - X\beta) + v)(s - X\beta)^T\right] = HQ.$$

2.1.3 Bayesian Analysis

The prior probability distribution function is modeled as a Gaussian, i.e.,

$$p(s|\beta) \propto \exp\left(-\frac{1}{2}(s - X\beta)^T Q^{-1}(s - X\beta)\right).$$

To express that β is unknown a priori, its prior probability density function is modeled uniformly over all space, i.e.,

$$p(\beta) \propto 1,$$

and thus

$$p(s, \beta) \propto \exp\left(-\frac{1}{2}(s - X\beta)^T Q^{-1}(s - X\beta)\right).$$

The likelihood function is then given by

$$p(y|s) \propto \exp\left(-\frac{1}{2}(y - Hs)^T R^{-1}(y - Hs)\right).$$

Thus, the posterior probability density function is

$$\begin{aligned}p(s, \beta) \propto &\exp\left(-\frac{1}{2}(s - X\beta)^T Q^{-1}(s - X\beta)\right) \times \\ &\exp\left(-\frac{1}{2}(y - Hs)^T R^{-1}(y - Hs)\right),\end{aligned}$$

which is again a Gaussian. The negative log of the posterior probability density function is

$$\begin{aligned}\mathcal{L} &= -\ln(p(s, \beta)) \\ &= \frac{1}{2}(s - X\beta)^T Q^{-1}(s - X\beta) \\ &+ \frac{1}{2}(y - Hs)^T R^{-1}(y - Hs) + \text{ constant.}\end{aligned}$$

The posterior mean values, indicated by $\hat{s}$ and $\hat{\beta}$ minimize $\mathcal{L}$. Setting the partial derivative of $\mathcal{L}$ with respect to s and β to zero, we get the following equations:

$$\begin{aligned}(\hat{s} - X\hat{\beta})^T Q^{-1} - (y - H\hat{s})^T R^{-1} H &= 0 \\ -\left(\hat{s} - X\hat{\beta}\right)^T Q^{-1} X &= 0\end{aligned} \tag{3}$$

To bring the solution to a computationally convenient form, we introduce a vector $\xi \in \mathbb{R}^{n\times 1}$ defined through

$$y - HX\hat{\beta} = \Psi\xi. \tag{4}$$

We then have

$$\boxed{\hat{s} = X\hat{\beta} + QH^T\xi.} \tag{5}$$

Substituting the above into equation (3), we get

$$\xi^T HX = 0. \tag{6}$$

Combining the equations (4) and (6), we get that

$$\boxed{\begin{bmatrix} \Psi & \Phi \\ \Phi^T & 0 \end{bmatrix} \begin{bmatrix} \xi \\ \hat{\beta} \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix}.} \tag{7}$$

Thus, the solution is obtained by solving a system of $n+p$ linear equations. The key equations are equations (5) and (7).

To quantify the uncertainty in the solution, the posterior covariance matrix is given by

$$\begin{aligned} V = Q - QH^T P_{yy} HQ - XP_{bb}X^T - XP_{yb}^T HQ \\ - QH^T P_{yb} X^T, \end{aligned} \tag{8}$$

where P_{yy}, P_{yb}, P_{bb} are obtained as

$$\begin{bmatrix} P_{yy} & P_{yb} \\ P_{yb}^T & P_{bb} \end{bmatrix} = \begin{bmatrix} \Psi & \Phi \\ \Phi^T & 0 \end{bmatrix}^{-1}. \tag{9}$$

The diagonal entries of the matrix V enable us to quantify the uncertainty, since each diagonal entry represents the variance of each of the unknowns.

2.1.4 Computational cost

Our goal is to obtain the best estimate $\hat{s}$ and the diagonal entries of the matrix V efficiently. To do so, we first need to solve the linear system in equation (7) and then obtain $\hat{s}$ using equation (5). To quantify the uncertainty in the solution, we need to obtain the diagonal entries of V using equation (8) and equation (9).

In all these equations, **we need the matrix matrix product QH^T, which is the bottleneck in terms of computational cost.** A conventional direct algorithm, where the number of measurements, n, is much smaller than the number of unknowns, m, would proceed as shown in Algorithm 1.

Since we have $p \ll n \ll m$, the total computational cost is $\mathcal{O}(nm^2 + n^2m)$.

The cost to solve equation (7) is independent of m. Hence, once the linear system is constructed, it can be solved by any conventional direct method like Gaussian elimination, which costs $\mathcal{O}((n+p)^3)$. Since we are interested in the case where the number of measurements is relatively small, i.e., $n \approx 200$, we are not interested in optimizing the solving phase, since the cost is independent of m. Also, the cost to compute a single diagonal entry of V from equation (8) is independent of m. Hence, the bottleneck is the cost $\mathcal{O}(nm^2)$, which arises from the computation of the matrix-matrix product $Q_H = QH^T$. The storage cost is dominated by the dense matrix Q, which costs $\mathcal{O}(m^2)$ to store. **Hence, our focus will be on efficiently storing Q and efficiently constructing $Q_H = QH^T$, since this is the bottleneck in the algorithm.**Some typical values of p, n and m encountered in practice are $p \sim$ 1–3, $n \sim$ 100–200 and $m \sim 10^5$–10^6.

2.2 Hierarchical matrices

In this section, we briefly describe the analytic and computational foundations of hierarchical matrices, which we will be using in this article. Hierarchical matrices are data-sparse approximations of dense matrices arising in applications like boundary integral equations, interpolation, etc. Hierarchical matrices, introduced by Hackbusch, et al. [6, 24, 28–30], are generalizations of the class of matrices for which the fast multipole method is applicable. An example of a hierarchical matrix arising out of a two-dimensional application is shown in figure 1. Of the class of hierarchical matrices, $\mathcal{H}^2$-matrices are precisely matrices for which the fast multipole method (FMM) is applicable. The FMM is a numerical technique to calculate matrix-vector products (or) equivalently sums of the form

$$f(x_i) = \sum_{j=1}^{N} K(x_i, y_j)\sigma_j,$$

where $i \in \{1, 2, 3, \ldots, M\}$, in $\mathcal{O}(M+N)$ operations as opposed to $\mathcal{O}(MN)$ with a controllable error ϵ. The FMM was originally introduced by Greengard and Rokhlin [25] based on Legendre polynomial expansions and spherical harmonics for the kernel $K(\vec{x}, \vec{y}) = \dfrac{1}{|x-y|}$.

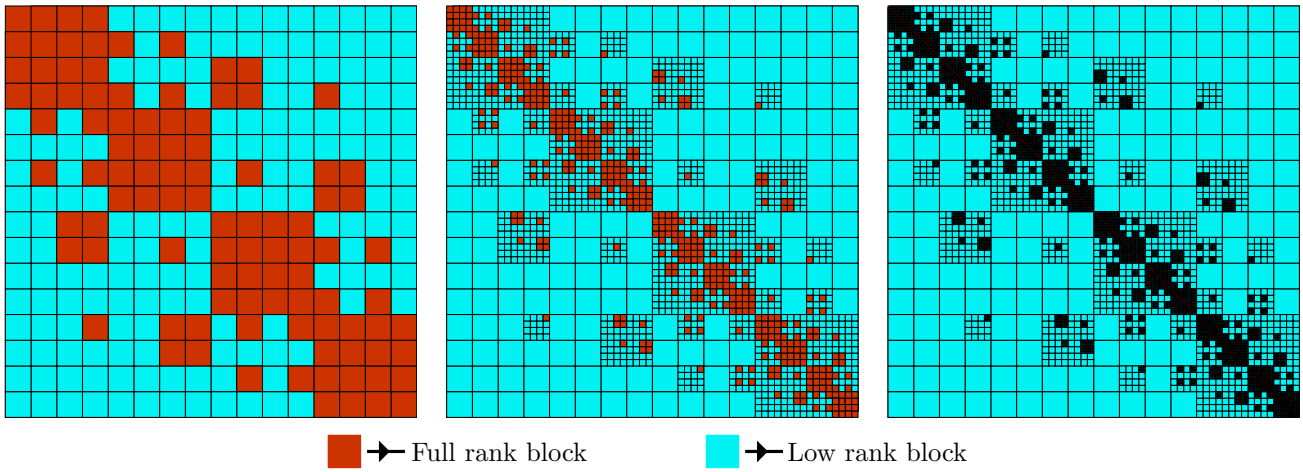


Figure 1: Hierarchical matrices arising out a 2D problem at different levels in the tree.

Historically the FMM has been used for the treatment of integral operators (boundary element method) and the solution of the resulting linear systems, which arise when solving partial differential equations, such as the Laplace, Helmholtz or Maxwell equations. It has been applied in a variety of fields including molecular dynamics, astrophysics, elastic materials, in graphics, etc. In all these cases, the FMM is used, as part of an iterative solution of linear systems (GMRES, conjugate gradient), to accelerate matrix-vector products. This is often the part of the calculation that is computationally the most expensive.

In our case though, the FMM will be used to construct the linear system, i.e., more precisely in constructing QH^T as discussed in the previous section. If the measurement matrix H is sparse, then the FMM can also be easily adapted to exploit the sparsity of the measurement matrix. Once the linear system has been constructed, since the size of the linear system is relatively small, the linear system can be solved directly using conventional direct solvers like the Gaussian elimination. One of the main advantages of FMMs, in general, is that they can be easily applied to non-uniform distributions of points. The FMM also comes with sharp *a-priori* error bounds.

As mentioned above, the original FMM is based on certain analytical expansions of the kernel $K(r)$, for example with spherical harmonics, powers of r, Bessel functions, etc. However it has since been realized that the FMM is based on a more general idea (see for example [19]), which is that certain matrices can be well-approximated using a low-rank matrix. More precisely, the key property behind the FMM is that the interactions $K(x_i, y_j)$ between far away clusters of points (far field interaction) can be efficiently represented using low-rank approximations. In general, matrices from boundary element method or the covariance matrix, Q, in our problem are full-rank. However, certain off-diagonal blocks (associated with well-separated clusters of points) can be efficiently represented using a low-rank matrix. This implies that the covariance matrix Q can be well-represented as a hierarchical matrix. An example of a hierarchical matrix is shown in figure 1.

Generally speaking, given a matrix $A \in \mathbb{R}^{M \times N}$ (in our case a sub-matrix of Q), the *optimal* rank p approximation can be obtained from its singular value decomposition (SVD) [22]. However, SVD is computationally expensive with a cost of $\mathcal{O}(MN \min(M, N))$. In recent years, there has been a tremendous progress [9, 20, 27, 47] in constructing fast approximate low-rank factorizations for matrices. Techniques like adaptive cross approximation [54] (ACA), pseudo-skeletal approximations [23], interpolatory decomposition [19], randomized algorithms [20, 41, 58] provide great ways for constructing efficient approximate low-rank representations.

For smooth kernels, Chebyshev polynomials have proven to be an attractive way to construct low-rank approximations and we refer the readers to [19, 46] for more details. In this article, we follow the fast multipole method discussed in [19]. One of the main reasons for choosing [19] is that the method is very general and is applicable for a wide range of kernels/covariance functions. Another advantage of [19] is that the precomputation cost is very small since the method is based on interpolating from Chebyshev nodes, for which efficient numerical algorithms are available. We will be making use of this fast multipole method coupled with the sparsity of the matrix H, to compute the product QH^T.

We briefly explain why the FMM is applicable to this class of problems. Given m points in space with position vector, $\{\vec{r}_k\}_{k=1}^m$, the covariance matrix Q is typically given as $Q(i,j) = K(\|\vec{r}_i - \vec{r}_j\|)$. For instance, some of the popular choices of $K(r)$ are $\exp(-r)$, $\frac{1}{r}$, $\exp(-r^2)$, $\frac{1}{1+r^2}$, etc. All the above mentioned covariance kernel functions are amenable to the use of the FMM since the interaction between well-separated clusters of

points can be well captured by modeling these interactions as low-rank interactions. This low-rank interaction enables us to reduce both the storage and computational cost. Theorems are available [19] to determine the rank p required to achieve an accuracy ε. Roughly speaking, the kernel K needs to be a smooth function, that is with a controllable growth in the complex plane away from the real line [19].

3 Problem specification

In this section, we describe the specific problem, which we solve by large-scale linear inversion. The problem deals with large-scale crosswell tomography to monitor the CO_2 plume in CO_2 sequestration sites. There are many techniques to perform geophysical imaging like electrical resistivity tomography, induced polarization technique, seismic tomography, etc. Of these techniques, crosswell tomography is an attractive technique for imaging in the context of CO_2 concentration. Typically, a compressional wave (P-wave) and/or a shear wave (S-wave) travel time measurements, using seismometers, are obtained. From these measurements, an inverse problem is solved to infer the properties of the subsurface. As more and more sensors are used and measurements become frequent and automated, and with the increasing need for finer resolutions, the need to process data sets of large sizes is of prime importance. Hence, efficient numerical algorithms to solve large-scale stochastic inversion problems is of utmost importance.

In our context, the crosswell tomography is used to image the seismic wave speed in the region of interest and thereby relate it to the evolution of the CO_2 plume in the domain. The problem configuration is shown in figure 2. The motivation for this capture geometry stems from [14]. We will compare the results we obtain using our fast large-scale linear inversion algorithm with the results in [14], which were obtained by running the forward model using TOUGH2, a multiphase-flow reservoir model [53] and patchy rock physical model [57].

A crosswell tomography is setup with n_s sources along the vertical line AB and n_r receivers along the vertical line CD. The sources emit a seismic wave and the receivers measure the time taken by the seismic wave from a source to hit the receiver. This is done for each source-receiver pair.

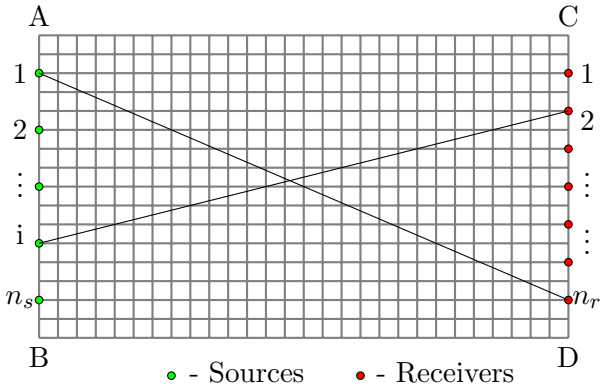


Figure 2: Discretization of the domain between the two wells. The line segment AB denotes the well where the n_s sources are placed, while the line segment CD denotes the well where the n_r receivers are placed. The seismic wave is generated by a source and the receiver measures the time taken for it to receive the seismic wave. Each source-receiver pair constitutes a measurement.

Our goal is to image the slowness of the medium inside the rectangular domain. Slowness is defined as the reciprocal of the speed of the seismic wave in the medium. In the context of CO_2 sequestration for example, the seismic wave travels considerably slower through CO_2 saturated rock [13, 14, 18] as opposed to the rest of the medium. Hence by measuring the slowness in the medium, we can estimate the CO_2 concentration at any point in the domain (with some uncertainty) and thereby the location of the CO_2 plume.

The above configuration is typical for most of the continuous crosswell seismic monitoring sites. We go about modeling the problem as follows. As a first order approximation, the seismic wave is modeled as traveling along a straight line from the sources to receivers with no reflections/refractions. The time taken by the seismic wave to travel from the source to the target is measured. Each source-receiver pair gives us a measurement and hence there are a total of $n = n_s \times n_r$ measurements. To obtain the slowness in the domain $ABDC$, the domain is discretized into m grid points (i.e., an $m_x \times m_y$ grid such that $m = m_x m_y$) and within each cell the slowness is assumed to be constant. Let y_{ij} denote the time taken by the seismic wave to travel from source i to receiver j. Let s_k be the slowness of the k^{th} cell. For every source-receiver (i, j) pair, we then have that

$$\sum_k l_{ij}^k s_k + v_{ij} = y_{ij}, \tag{10}$$

where l_{ij}^k denotes the length traveled by the ray from source i to the receiver j through the k^{th} cell and v_{ij} denotes the measurement error $i \in \{1, 2, \ldots, n_s\}$, $j \in \{1, 2, \ldots, n_r\}$, $k \in \{1, 2, \ldots, m\}$. Equation (10) can be written in a matrix-vector form:

$$Hs + v = y,$$

where $H \in \mathbb{R}^{n \times m}$, $s \in \mathbb{R}^{m \times 1}$, $v \in \mathbb{R}^{n \times 1}$ and $y \in \mathbb{R}^{n \times 1}$. Typically, the measurement error is modeled as a Gaussian white-noise. For most problems of practical interest, the number of measurements n is much smaller than the number of unknowns m, i.e., $n \ll m$. This under-determined linear system constitutes our inverse problem.

We will now analyze the matrix H to figure out the structure of the linear system. Each row of the matrix H corresponds to a source-receiver pair. Consider the source i and receiver j where $i \in \{1, 2, \ldots, n_s\}$ and $j \in \{1, 2, \ldots, n_r\}$. This corresponds to the $((i-1)n_r + j)^{th}$ row of the matrix H. Since we have modeled the wave traveling from a source to a receiver as a straight line without reflections/refractions, the non-zero entries along each row correspond to the cells hit by the ray from a source to the receiver. Since the wave from the source to receiver travels along a straight line, only the cells that lie on this straight line contribute to the non-zero entries. Hence, every row of H has only $\mathcal{O}(\sqrt{m})$ non-zeros. Hence, the matrix H is sparse since it has only $\mathcal{O}(n\sqrt{m})$ entries as opposed to $\mathcal{O}(nm)$ entries. We would hence like to take advantage of this sparsity to accelerate our computations. This underdetermined system is solved using Bayesian approach discussed in subsection 2.1. Since we would also like to characterize the fine-scale features, m can be much larger than n. Section 4 discusses the fast algorithm to solve the above problem using Bayesian approach.

4 Algorithm

In this section, we discuss in detail the fast algorithm for the large-scale linear inverse problem using Bayesian geostatistical approach and how this algorithm is applied to the crosswell tomography problem. We would like to point out that the algorithm is far more general and can be used for other linear inverse problems and not just the crosswell tomography case.

As discussed in section 2.1.4, the main bottleneck in the entire computation is constructing the matrix product QH^T, where $Q \in \mathbb{R}^{m \times m}$ is the covariance matrix and $H \in \mathbb{R}^{n \times m}$ is the measurement matrix. Let

$$H = \begin{bmatrix} h_1 & h_2 & \cdots & h_n \end{bmatrix}^T,$$

where $h_i \in \mathbb{R}^{m \times 1}$ are column vectors, each corresponding to a measurement. As mentioned in 2.2, the fast multipole method can be used to accelerate matrix-vector product computations, because the submatrices, of a dense covariance matrix Q, corresponding to the interaction between well-separated clusters can be efficiently represented using low-rank matrices.

Now note that

$$QH^T = \begin{bmatrix} Qh_1 & Qh_2 & \cdots & Qh_n \end{bmatrix}.$$

Hence, a straight-forward way to accelerate the computation of the matrix-matrix product QH^T is to accelerate the computation of each matrix vector product Qh_i, where $i \in \{1, 2, \ldots, n\}$. The black box FMM [19] reduces the cost of computing Qh_i for each i from $\mathcal{O}(m^2)$ to $\mathcal{O}(m)$. (For more details on the algorithm, see page 8716–17 of [19].) Hence, the total cost to compute the matrix-matrix product QH^T is just $\mathcal{O}(nm)$ as opposed to $\mathcal{O}(nm^2)$. As discussed in 2.1.4, this step is the bottleneck in the computation, since the number of measurements is much smaller than the number of unknowns, i.e., $n \ll m$. This is described in Algorithm 1.

With the FMM, the first step is no longer the bottleneck, while the second step becomes the limiting step. The overall computational cost scales as $\mathcal{O}(n^2 m)$. This computational cost cannot be further reduced if the measurement matrix H is assumed to be dense.

However, if the matrix H happens to be sparse, the fast multipole method can be easily adapted to exploit this sparsity and thereby further reduce the computational complexity. For instance, in the case of the crosswell tomography application, every row of H, i.e., $\{h_i^T\}_{i=1}^n$ has only $\mathcal{O}(\sqrt{m})$ non-zeros as opposed to $\mathcal{O}(m)$ entries. This is due to the fact that the non-zero entries along a row of the measurement matrix H are due to a seismic wave from a source to a receiver (see explanation in the previous section).

We can take advantage of this sparsity twice. First, in the FMM, although the asymptotic cost is $\mathcal{O}(nm)$ irrespective of whether H is sparse or not, the constant in front of nm is reduced with a sparse H. This is due to the fact that many source leaves in the FMM tree are empty (that is do not intersect the ray). Second, the computational cost for the second step, which was the bottleneck in algorithm 1, can be reduced from $\mathcal{O}(n^2 m)$ to $\mathcal{O}(n^2\sqrt{m})$ using a sparse matrix-vector product technique such as the compressed sparse row (CSR) format [17, 55] (e.g., since there are only $\mathcal{O}(\sqrt{m})$ non-zero entries in each row of H). See Algorithm 1.

Hence, the large-scale linear inversion problem has an overall computational complexity of $\mathcal{O}(nm + n^2\sqrt{m})$.

Algorithm 1 Computational cost for different steps in the large-scale linear inversion problem.

Operation	**Computational cost using different metho**		
	Conventional	FMM	FMM with spars
Compute the matrix-matrix product $Q_H = QH^T$ using the FMM	$\mathcal{O}(nm^2)$	$\mathcal{O}(nm)$	$\mathcal{O}(nm)$
Compute the matrix $\tilde{\Psi} = HQ_H$	$\mathcal{O}(n^2m)$	$\mathcal{O}(n^2m)$	$\mathcal{O}(n^2\sqrt{m})$
Compute the matrix $\Psi = \tilde{\Psi} + R$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
Compute the matrix $\Phi = HX$	$\mathcal{O}(nmp)$	$\mathcal{O}(nmp)$	$\mathcal{O}(n\sqrt{m}p)$
Solve the linear system directly in equation (7) to get ξ and β	$\mathcal{O}((n+p)^3)$	$\mathcal{O}((n+p)^3)$	$\mathcal{O}((n+p)^3)$
Compute $\hat{s} = X\beta + Q_H\xi$	$\mathcal{O}(mp+mn)$	$\mathcal{O}(mp+mn)$	$\mathcal{O}(mp+mn)$
Compute all the diagonal entries of V using the equation below	$\mathcal{O}(n^2m)$	$\mathcal{O}(n^2m)$	$\mathcal{O}(n^2m)$

$$V(i,i) = Q(i,i) - \underbrace{Q_H(i,:)P_{yy}(Q_H(i,:))^T}_{\mathcal{O}(n^2)} - \underbrace{X(i,:)P_{bb}(X(i,:))^T}_{\mathcal{O}(p^2)} - \underbrace{2X(i,:)P_{yb}^T(Q_H(i,:))^T}_{\mathcal{O}(pn)}$$

To quantify the uncertainty in the solution, we need to obtain the diagonal entries of V using equations (8) and (9). As before, once $Q_H = QH^T$ has been obtained, computing the uncertainty of a single cell costs us only $\mathcal{O}(n^2)$:

$$V(i,i) = Q(i,i) - \underbrace{Q_H(i,:)P_{yy}(Q_H(i,:))^T}_{\mathcal{O}(n^2)} - \underbrace{X(i,:)P_{bb}(X(i,:))^T}_{\mathcal{O}(p^2)} - \underbrace{2X(i,:)P_{yb}^T(Q_H(i,:))^T}_{\mathcal{O}(pn)}.$$

Hence, the total cost of computing the uncertainty for all the cells grows linearly with the number of cells and is given by $\mathcal{O}(n^2m)$.

5 Numerical benchmark

We now present numerical benchmarks for two cases. The first one is a synthetic data set for the crosswell tomography problem while the second one is a real data set for the crosswell tomography problem. The problem set up for both the cases is similar to the one shown in figure 2, i.e., the sources are distributed along the vertical line AB and the receivers are distributed along the vertical line CD. The goal is to image the slowness in both the cases.

5.1 Synthetic data set

We first present results for a synthetic data set. In the case of a synthetic data set, the domain we consider is a $70m \times 40m$ rectangular domain, i.e., the horizontal distance between the sources and receivers is $70m$. To benchmark our algorithm, we feed in a known slowness field as shown in figure 3. We compute our measurements as follows. We distribute 12 uniformly spaced sources along AB, i.e., $n_s = 12$ and 24 uniformly spaced receivers along CD, i.e., $n_r = 24$. This gives us a total of 288 measurements. We compute the time taken for the seismic wave to travel from each source to a receiver. We add a random Gaussian noise with mean 0 and variance 10^{-4} to each of the measurement to model the measurement error. The covariance function $K(r)$ is taken as the Gaussian covariance, i.e., $K(r) = \exp(-(r/\theta)^2)$ and the matrix X is taken as a vector with all 1's. The correlation length θ is set to 10. As a next step, the FMM requires the number of Chebyshev nodes along each direction as an input. In this case, for the Gaussian, the number of Chebyshev nodes in each direction is taken as 5.

We now discretize the domain into $\sqrt{m} \times \sqrt{m}$ grid points and reconstruct the image. The number of grid points m is varied from $2,500$ to $1,000,000$ and the time taken by the fast algorithm to image the slowness is compared against the conventional direct algorithm. The comparison of the time taken is tabulated in table 1. For $m = 1,000,000$, the reconstructed image using the fast algorithm is shown on the right side in figure 3. Note that there is a discrepancy between the true and reconstructed slowness for the synthetic data set. This discrepancy shows that in case of limited source-receiver pairs, it is highly important to optimize the capture geometry. This optimization of the capture geometry has been done for the real data set. Once the capture geometry has been optimized, grid refinement is crucial to study the true evolution of the plume. Also, the discrepancy between the true and reconstructed image will further be reduced if we have more measurements, in which case again refining the grid is important.

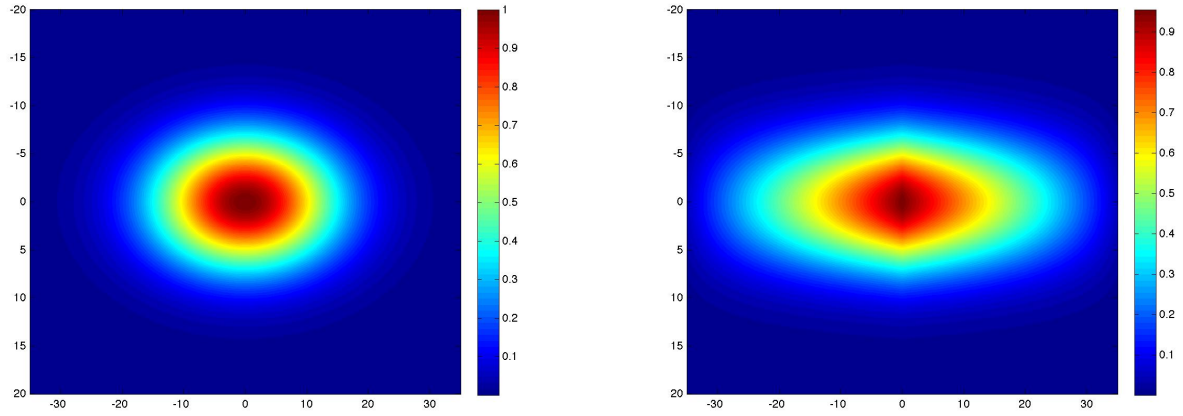


Figure 3: Left: known slowness field — synthetic data set; Right: reconstructed slowness field for the synthetic data set using a million grid points.

Table 1: Comparison of time taken and relative error for the synthetic test case shown in figure 3. The relative error is computed by comparing the reconstructed image (right panel in figure 3) and the true solution (left panel). We chose the approximation order in the FMM such that the FMM error is small compared to the inverse algorithm error. As a result the relative error for the direct and fast algorithms are comparable. A more complete error convergence is shown for the real test case in figure 4 and 9. The algorithm was implemented in C++. All the calculations were run on a machine with a single core 2.66 GHz processor and 8 GB RAM. There was no necessity to parallelize the implementation for the present purpose due to the speedup achieved with the proposed algorithm.

Grid size	Time taken in secs		Relative error	
m	Conventional direct algorithm	Fast algorithm	Conventional direct algorithm	Fast algorithm
$2,500$	$2.1 \cdot 10^{+1}$	$2.3 \cdot 10^{+1}$	$4.45 \cdot 10^{-1}$	$4.39 \cdot 10^{-1}$
$10,000$	$3.9 \cdot 10^{+2}$	$6.7 \cdot 10^{+1}$	$4.39 \cdot 10^{-1}$	$4.41 \cdot 10^{-1}$
$22,500$	$2.0 \cdot 10^{+3}$	$1.4 \cdot 10^{+2}$	$3.82 \cdot 10^{-1}$	$3.89 \cdot 10^{-1}$
$40,000$	$6.4 \cdot 10^{+3}$	$2.6 \cdot 10^{+2}$	$3.47 \cdot 10^{-1}$	$3.39 \cdot 10^{-1}$
$250,000$	–	$1.3 \cdot 10^{+3}$	–	$2.85 \cdot 10^{-1}$
$1,000,000$	–	$5.1 \cdot 10^{+3}$	–	$2.69 \cdot 10^{-1}$

Table 1 also compares the estimation error obtained by the stochastic linear inversion using the direct algorithm and our fast algorithm. The relative estimation error is defined as $\|s_{\text{reconstructed}} - s_{\text{true}}\|/\|s_{\text{true}}\|$. As the table indicates, there is very little difference in the estimation error between the conventional direct algorithm and our fast algorithm. This is in fact a strong argument in favor of fast algorithms. In the presence of large modeling errors, we can afford to approximate the covariance matrix, which is what our algorithm precisely does.

Figure 5 compares the time taken and storage cost of a conventional direct algorithm with the fast algorithm proposed by us. Note that the plots are on a log-log scale and clearly indicate that the time taken and storage cost for the conventional algorithm scales as $\mathcal{O}(m^2)$ whereas the time taken and storage cost for the novel fast algorithm scales as $\mathcal{O}(m)$.

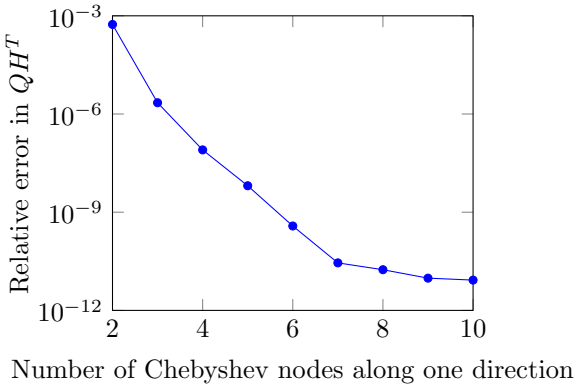


Figure 4: Relative error in QH^T versus the number of Chebyshev nodes along one direction for the Gaussian covariance for $m = 40,000$.

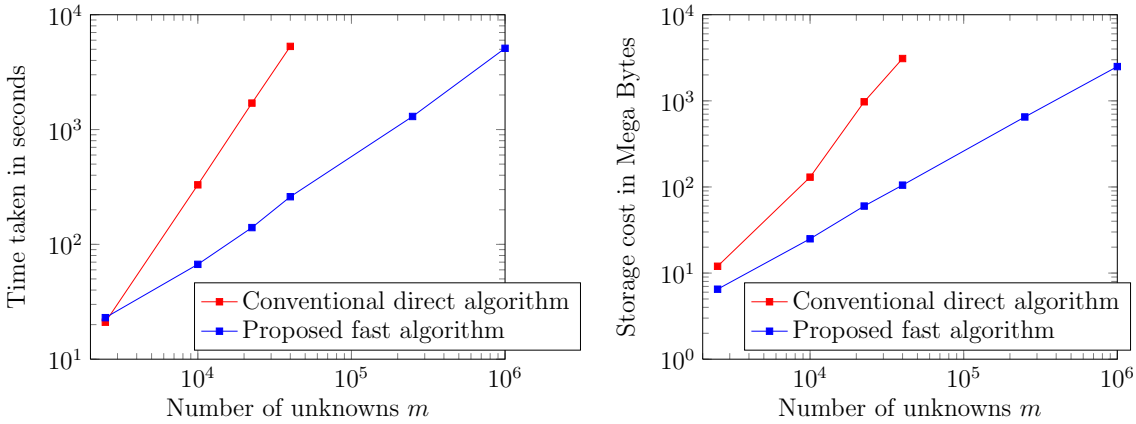


Figure 5: Left: comparison of the time taken by the fast algorithm vs the conventional direct algorithm; Right: comparison of the storage cost.

We also perform an analysis of the time taken and the storage cost by our fast algorithm when varying the number of measurements, n. The results are plotted in figure 6. This further validates the fact that the time taken and the storage cost scales only linearly with the number of unknowns m, irrespective of the number of measurements n. Note that all the plots are on a log-log scale.

5.2 Real data set

We now present detailed numerical benchmarking for a real data set. The linear inverse problem arises from monitoring a CO_2 plume in the subsurface for safe carbon storage. The site is the Frio II test site in southeast Texas near the Gulf of Mexico. The experiment is described in [14].

We briefly describe the capture geometry. There are two wells, an injection well and a monitoring well separated by 30 meters at a depth of around 1,600 m. To be precise, we consider a depth ranging from 1,620 m to 1,680 m. The CO_2 injection occurs at 1,665 m underground. The experiment measures the time taken by the seismic wave from a series of sources to the receivers. Using the seismic traveltime measured as constraints, a series of forward flow models governing the migration are produced by a state-of-the-art model, the TOUGH2/ECO2N system [52, 53]. This gives us the CO_2 plume induced slowness at different time

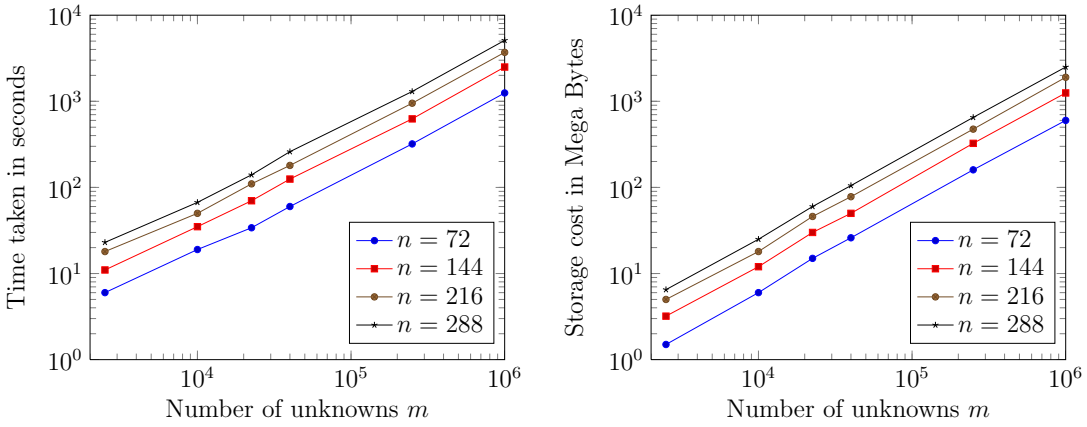


Figure 6: Left: time taken by the fast algorithm as a function of the number of unknowns; Right: storage cost by the fast algorithm as a function of the number of unknowns.

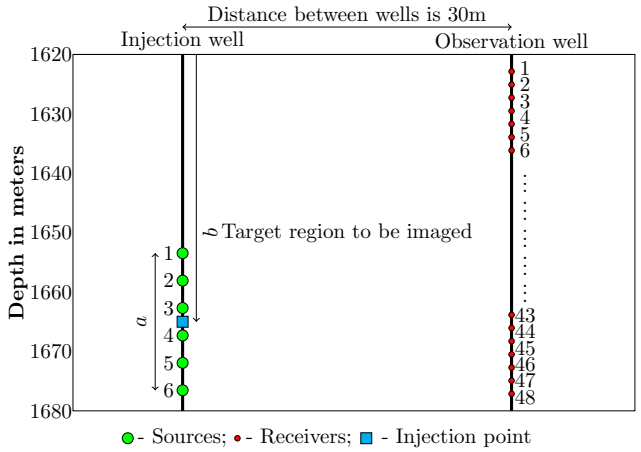


Figure 7: The schematic on the left is the actual crosswell geometry from [14].

instances. We obtained this data set, i.e., the location of the CO_2 plume after 120 hours, from one of our collaborators, Jonathan Ajo-Franklin, from Lawrence Berkeley National Laboratories. The data obtained is shown in figure 10. For more details on the test site and the capture geometry, we refer the readers to [14]. A pictorial representation of the test site is shown in figure 7.

From the data, i.e., the CO_2 plume after 120 hours of injection, the measurements to perform the linear inversion are obtained as follows. Given the CO_2 plume configuration after 120 hours, for a fixed capture geometry, we measure the time taken from a source to a receiver by modeling that the ray from the source to the receiver travels in a straight line. The signal to noise ratio of each of the measurements is typically around 65 dB. This dictates the variance of the measurement error in our measurements, which is set at 10^{-5}. These are the diagonal entries of the matrix R. The covariance function we chose here is $K(r) = -r/\theta$, where the correlation length, θ, is 90 m. The matrix, X, is taken as a vector with all 1's. As a next step, the fast multipole method requires the number of Chebyshev nodes along each direction as an input, which we set equal to 5. This means that the rank of interaction between well-separated clusters is 25. We then discretize the domain into m grid points to reconstruct the image. Once we reconstruct the image, we are able to obtain the uncertainty in the solution.

Note that in the above procedure, optimization of the capture geometry [1], i.e., the placement of the sources and receivers, is of prime importance in crosswell tomography applications, especially when the number of measurements is not large, since it has a direct bearing on the resolution of the image obtained. We place 6 sources and 48 receivers. The source transducers are typically more expensive around \$50,000. Hence, there is a large number of receivers, in our case 48, compared to the sources. Because of this, we place the receivers along the monitoring well such that they are uniformly spaced. However, since we only have a few sources (6), it is important for us to choose where exactly we need to place these 6 sources. Hence, we need to optimize the source locations. The unknown parameters we want to optimize are the array length of the sources, a, and the location of the source array center, b. Refer to figure 7 for more details. Given this set of 6 sources and 48 receivers, for a fixed a and b, we have a total of $6 \times 48 = 288$ measurements. We now use these 288 measurements to perform the linear inversion for each pair (a, b).

To optimize for the parameters 'a' and 'b', we proceed as follows. We consider a 237×217 grid. For a fixed 'a' and 'b', we estimate the relative error using

$$F(a, b) = \|s_{\text{computed}} - s_{\text{true}}\| \,/\, \|s_{\text{true}}\|.$$

The above can be computed since we already have access to the true data. This is the objective function we are minimizing, i.e.,

$$(a_{\text{opt}}, b_{\text{opt}}) = \text{argmin}_{a,b} F(a, b).$$

Note that in the current context, since we have access to the true slowness, we can compute the exact relative error. In applications where we do not have access to the true slowness, we could resort to the method of gradient desecent to find the local minimum of the relative error. Note that in the context of gradient descent as well, different capture geometries need to evaluated, which would be computationally impractical if we were to rely on the conventional algorithm.

The key point of this calculation is not to demonstrate an algorithm to optimize the placement of sensors but rather show how fast algorithms will be useful. Although computing one solution for a mid-size problem, e.g., a grid of size 200×200, is feasible without a fast algorithm, it would not be practical to calculate many solutions in the context of an optimization loop. Without the fast algorithm, **we would require close to** 2 **hours to obtain the solution and evaluate the objective function for a single capture geometry. With our fast algorithm in place, we are able to obtain the solution and evaluate the objective function for** 25 **different capture geometries in the same** 2 **hours.** The comparison will be even starker if we want a finer resolution. To find the optimal value for 'a' and 'b', we evaluate $F(a, b)$ for a range of a and b. Each evaluation requires computing QH^T, which is efficiently done using the FMM. The plot of $F(a, b)$ versus 'a' and 'b' is shown in figure 8.

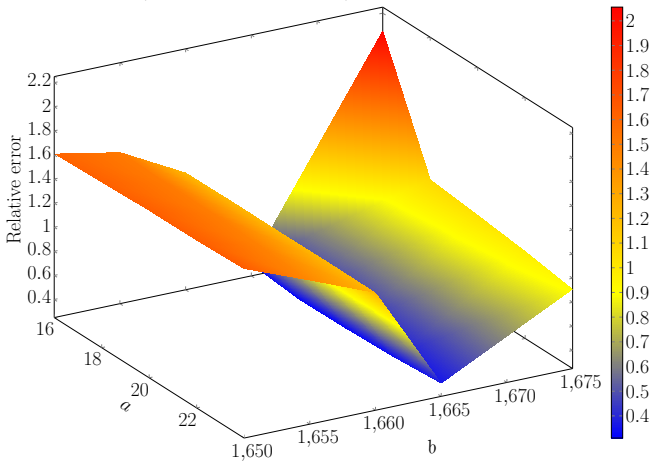
We also mention that if we were to tackle three dimensional problems, the size of the 3D mesh would probably make a single calculation using the direct method infeasible.

We obtained the optimal array length as $a_{\text{opt}} = 18$ m and the center of the source array to be at $b_{\text{opt}} = 1665$ m, i.e., this choice of 'a' and 'b' minimizes the relative error $F(a, b)$. We fixed the center of the source array at $b = 1665$ m $= b_{\text{opt}}$ and the array length at $a = 20$ m (this is very close to the optimum and there is no significant difference in the estimation relative error), and performed the following numerical benchmarks.

First, we performed validation on the matrix matrix product obtained using the FMM. The figure 9 shows the decay in the relative error in the matrix-matrix product QH^T using the FMM and the direct matrix-matrix product vs the number of Chebyshev nodes along one direction, i.e.,

$$\epsilon = \|QH^T_{\text{computed}} - QH^T_{\text{exact}}\| \,/\, \|QH^T_{\text{exact}}\|,$$

Figure 8: Relative error for the reconstructed image (compared to the exact answer in the left panel of Fig. 10) as a function of a, the length of the source array, and b, the location of the center of the source array. We used 25 evaluation points (5 along a and b).



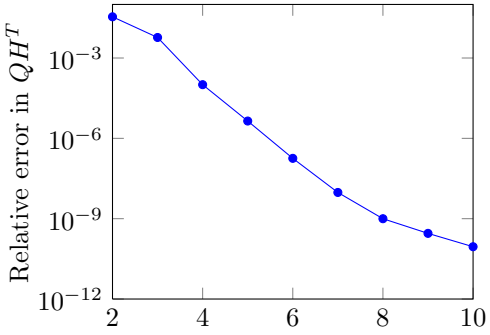


Figure 9: Relative error in QH^T versus the number of Chebyshev nodes along one direction for $m = 237 \times 217 = 51,429$.

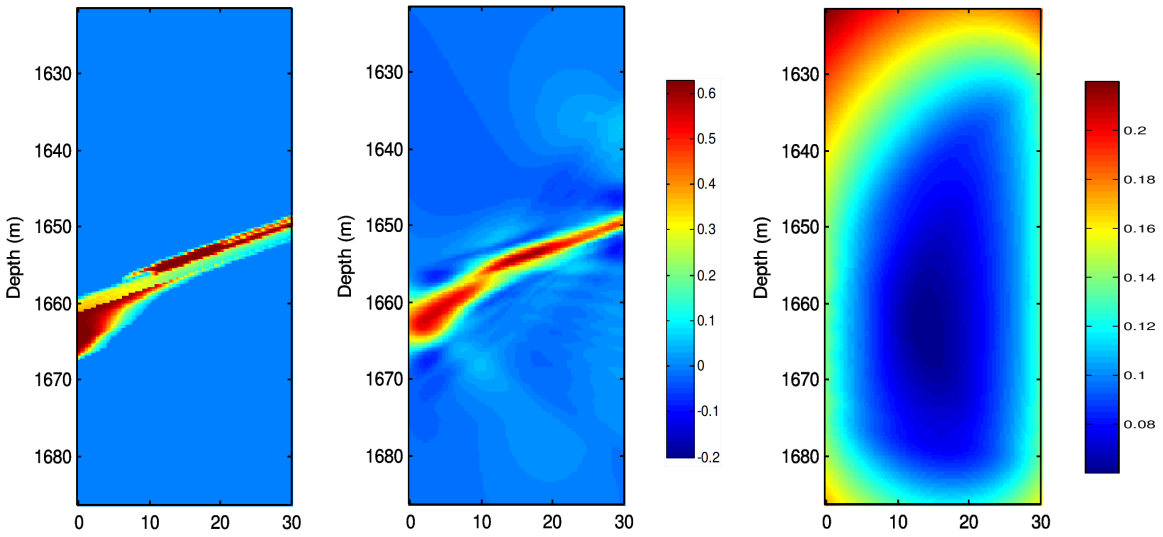


Figure 10: Left: true slowness using TOUGH2/ECO2N [52, 53] simulations, 120 hours after injection; Middle: reconstructed slowness using our proposed algorithm for the optimal choice of a and b; Right: uncertainty in the estimated solution. All these plots are for a 237×217 grid. Each unit of slowness corresponds to 10^4 sec/m.

Table 2: Comparison of the computational time for the real test case of figure 10

Grid size	Time taken in seconds	
m	Conventional direct algorithm	Fast algorithm
55×59	$2.7 \cdot 10^{+1}$	$3.1 \cdot 10^{+1}$
117×109	$4.0 \cdot 10^{+2}$	$9.2 \cdot 10^{+1}$
237×217	$6.4 \cdot 10^{+3}$	$2.8 \cdot 10^{+2}$
500×500	–	$1.1 \cdot 10^{+3}$

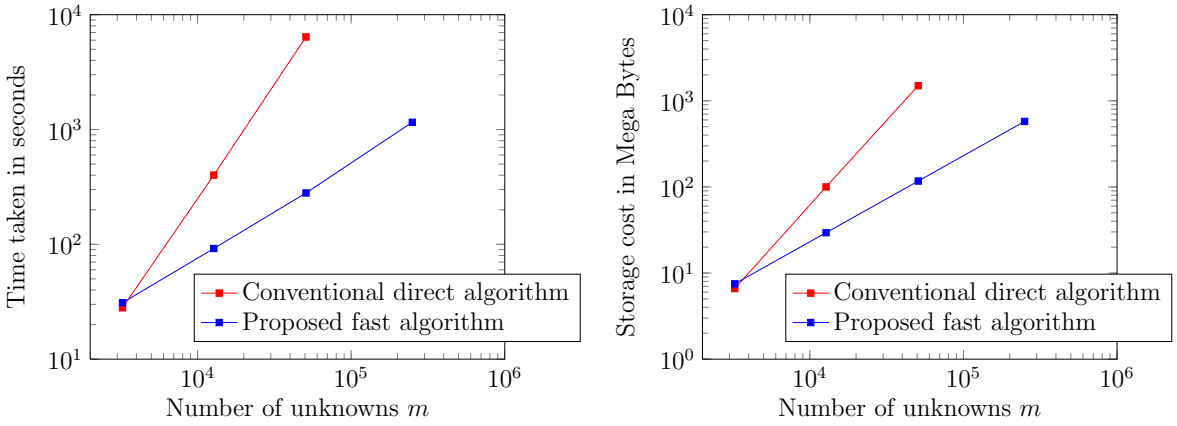


Figure 11: Left: time taken by the fast algorithm and the conventional direct algorithm; Right: storage cost.

for a 237×217 grid. The covariance kernel is given by $-r/\theta$. In the rest of our calculation we chose 5 Chebyshev nodes in each direction, which corresponds to an error around 10^{-6}. This is significantly below the reconstruction error due to the limited number of measurements.

For $a = 20$ m and $b = 1665$ m, we plot the image of the estimated slowness. This is shown in figure 10. There seems to be a good agreement with the true data shown in figure 10. The uncertainty in the image reconstructed is shown on the right panel of figure 10.

We now compare the time taken by the fast algorithm to image the slowness and the time taken by the conventional direct algorithm for different grid sizes keeping the array length and the center of the sources fixed at $a = 20$ m and $b = 1665$ m. The comparison of the time taken is tabulated in table 2 and the corresponding figure comparing the time taken and storage cost is also shown.

6 Conclusions and future work

We have presented a novel fast algorithm for large-scale linear inversion. In particular, we solve a problem with few measurements (n) and a large number of unknowns (m), i.e., $n \ll m$. The algorithm is illustrated by applying it to a synthetic data set (tomography problem), where we estimate one million unknowns. The fast algorithm is also applied for a real data set, from a crosswell tomography of a CO_2 sequestration site, to estimate 250,000 unknowns. The reconstruction using the fast linear inversion agrees well with the actual data. The computational speedup achieved by our algorithm allows us to, for example, (i) quantify the uncertainty in the solution; (ii) optimize the capture geometry. Optimizing the capture geometry would not have been possible without the fast algorithm, which highlights the potential impact of this type of technique. Further note that the entire algorithm is mostly a matrix-free approach, once the covariance kernel is given in an analytic form. This is of critical importance, especially in large-scale real-time monitoring applications, where the measurement matrix, H, might also be given in matrix free form. The matrix-matrix product QH^T, which is all we need for the inversion, can then be obtained using a matrix-free approach. This algorithm can also be combined with different filtering algorithms, thereby enabling large-scale real-time data-assimilation and inversion possible, which is the subject of our future work.

The fast algorithm for linear inversion hinges on the stochastic Bayesian approach and the hierarchical matrix approach. The algorithm can also exploit the underlying sparsity of the measurement operator. This in turn also results in a significant reduction in runtime. The new algorithm wins on two important counts. The speedup gained by using this algorithm is very significant, as demonstrated in the numerical benchmarks, and the storage is minimized tremendously. The speedup is especially significant when the number of unknowns m

is very large.

Acknowledgements

The authors were supported by "US Department of Energy, National Energy Technology Laboratory" (DOE, NETL) under the award number: DE-FE0009260, and also by the "National Science Foundation" —Division of Mathematical Sciences —under the award number: 1228275. The authors would also like to thank the "The Global Climate and Energy Project" (GCEP) at Stanford University, and the "Army High Performance Computing Research Center" (AHPCRC, sponsored by the U.S. Army Research Laboratory under contract No. W911NF-07-2-0027).

References

[1] Ajo-Franklin J (2009) Optimal experiment design for time-lapse traveltime tomography. Geophysics 74(4):Q27–Q40

[2] Barnes J, Hut P (1986) A hierarchical $\mathcal{O}(N \log N)$ force-calculation algorithm. Nature 324(4):446–449

[3] Beatson R, Greengard L (1997) A short course on fast multipole methods. Wavelets, multilevel methods and elliptic PDEs pp 1–37

[4] Beatson R, Newsam G (1992) Fast evaluation of radial basis functions: I. Computers & Mathematics with Applications 24(12):7–19

[5] Bjørnstad O, Falck W (2001) Nonparametric spatial covariance functions: estimation and testing. Environmental and Ecological Statistics 8(1):53–70

[6] Börm S, Grasedyck L, Hackbusch W (2005) Hierarchical matrices. Lecture notes 21

[7] Burrus C, Gopinath R, Guo H (1998) Introduction to wavelets and wavelet transforms: A primer. Recherche 67:02

[8] Cheng H, Greengard L, Rokhlin V (1999) A fast adaptive multipole algorithm in three dimensions. Journal of Computational Physics 155(2):468–498

[9] Cheng H, Gimbutas Z, Martinsson P, Rokhlin V (2005) On the compression of low-rank matrices. SIAM Journal on Scientific Computing 26(4):1389–1404

[10] Coifman R, Rokhlin V, Wandzura S (1993) The fast multipole method for the wave equation: A pedestrian prescription. Antennas and Propagation Magazine, IEEE 35(3):7–12

[11] Cooley J, Tukey J (1965) An algorithm for the machine calculation of complex fourier series. Math Comput 19(90):297–301

[12] Cornford D, Csató L, Opper M (2005) Sequential, bayesian geostatistics: a principled method for large data sets. Geographical Analysis 37(2):183–199

[13] Daley T, Ajo-Franklin J, Doughty C (2008) Integration of crosswell CASSM (Continuous Active Source Seismic Monitoring) and flow modeling for imaging of a CO2 plume in a brine aquifer. In: 2008 SEG Annual Meeting

[14] Daley TM, Solbau RD, Ajo-Franklin JB, Benson SM (2007) Continuous active-source seismic monitoring of co 2 injection in a brine aquifer. Geophysics 72(5):A57–A61

[15] Darve E (2000) The fast multipole method i: Error analysis and asymptotic complexity. SIAM Journal on Numerical Analysis 38(1):98–128

[16] Darve E (2000) The fast multipole method: numerical implementation. Journal of Computational Physics 160(1):195–240

[17] Davis T (2006) Direct methods for sparse linear systems, vol 2. Society for Industrial Mathematics

[18] Doughty C, Freifeld B, Trautz R (2008) Site characterization for CO_2 geologic storage and vice versa: the Frio brine pilot, Texas, USA as a case study. Environmental Geology 54(8):1635–1656

[19] Fong W, Darve E (2009) The black-box fast multipole method. Journal of Computational Physics 228(23):8712–8725

[20] Frieze A, Kannan R, Vempala S (2004) Fast Monte-Carlo algorithms for finding low-rank approximations. Journal of the ACM (JACM) 51(6):1025–1041

[21] Fritz J, Neuweiler I, Nowak W (2009) Application of FFT-based algorithms for large-scale universal kriging problems. Mathematical Geosciences 41(5):509–533

[22] Golub G, Van Loan C (1996) Matrix computations, vol 3. Johns Hopkins Univ Press

[23] Goreinov S, Tyrtyshnikov E, Zamarashkin N (1997) A theory of pseudoskeleton approximations. Linear Algebra and its Applications 261(1-3):1–21

[24] Grasedyck L, Hackbusch W (2003) Construction and arithmetics of $\mathcal{H}$-matrices. Computing 70(4):295–334

[25] Greengard L, Rokhlin V (1987) A fast algorithm for particle simulations. Journal of Computational Physics 73(2):325–348

[26] Greengard L, Rokhlin V (1997) A new version of the fast multipole method for the Laplace equation in three dimensions. Acta Numerica 6(1):229–269

[27] Gu M, Eisenstat S (1996) Efficient algorithms for computing a strong rank-revealing QR factorization. Society 17(4):848–869

[28] Hackbusch W (1999) A sparse matrix arithmetic based on $\mathcal{H}$-matrices. Part I: Introduction to $\mathcal{H}$-matrices. Computing 62(2):89–108

[29] Hackbusch W, Börm S (2002) Data-sparse approximation by adaptive $\mathcal{H}^2$-matrices. Computing 69(1):1–35

[30] Hackbusch W, Khoromskij B (2000) A sparse $\mathcal{H}$-matrix arithmetic. Computing 64(1):21–47

[31] Hackbusch W, Nowak Z (1989) On the fast matrix multiplication in the boundary element method by panel clustering. Numerische Mathematik 54(4):463–491

[32] Kitanidis P (1983) Statistical estimation of polynomial generalized covariance functions and hydrologic applications. Water Resources Research 19(4):909–921

[33] Kitanidis P (1993) Generalized covariance functions in estimation. Mathematical Geology 25(5):525–540

[34] Kitanidis P (1995) Quasi-linear geostatistical theory for inversing. Water Resources Research 31(10):2411–2419

[35] Kitanidis P (1996) On the geostatistical approach to the inverse problem. Advances in Water Resources 19(6):333–342

[36] Kitanidis P (1999) Generalized covariance functions associated with the Laplace equation and their use in interpolation and inverse problems. Water Resources Research 35(5):1361–1367

[37] Kitanidis P, Vomvoris E (1983) A geostatistical approach to the inverse problem in groundwater modeling (steady state) and one-dimensional simulations. Water Resources Research 19(3):677–690

[38] Kitanidis PK (1997) Introduction to geostatistics: applications to hydrogeology. Cambridge Univ Pr

[39] Kitanidis PK (2007) On stochastic inverse modeling. Geophysical Monograph Series 171:19–30

[40] Li W, Cirpka O (2006) Efficient geostatistical inverse methods for structured and unstructured grids. Water Resources Research 42(6):W06,402

[41] Liberty E, Woolfe F, Martinsson P, Rokhlin V, Tygert M (2007) Randomized algorithms for the low-rank approximation of matrices. Proceedings of the National Academy of Sciences 104(51):20,167

[42] Liu X, Kitanidis P (2011) Large-scale inverse modeling with an application in hydraulic tomography. Water Resources Research 47(2):W02,501

[43] Liu X, Illman W, Craig A, Zhu J, Yeh T (2007) Laboratory sandbox validation of transient hydraulic tomography. Water Resources Research 43(5):W05,404

[44] Mallat S (1989) A theory for multiresolution signal decomposition: The wavelet representation. Pattern Analysis and Machine Intelligence, IEEE Transactions on 11(7):674–693

[45] Mallat S (1999) A wavelet tour of signal processing. Academic Press

[46] Messner M, Schanz M, Darve E (2012) Fast directional multilevel summation for oscillatory kernels based on chebyshev interpolation. Journal of Computational Physics 231(4):1175–1196

[47] Miranian L, Gu M (2003) Strong rank revealing LU factorizations. Linear Algebra and its Applications 367:1–16

[48] Nishimura N (2002) Fast multipole accelerated boundary integral equation methods. Applied Mechanics Reviews 55(4):299–324

[49] Nowak W, Cirpka O (2006) Geostatistical inference of hydraulic conductivity and dispersivities from hydraulic heads and tracer data. Water Resources Research 42(8):8416

[50] Nowak W, Tenkleve S, Cirpka O (2003) Efficient computation of linearized cross-covariance and auto-covariance matrices of interdependent quantities. Mathematical geology 35(1):53–66

[51] Pollock D, Cirpka O (2010) Fully coupled hydrogeophysical inversion of synthetic salt tracer experiments. Water Resources Research 46(7):W07,501

[52] Pruess K (2005) ECO2N: A TOUGH2 fluid property module for mixtures of water, NaCl, and CO_2. Lawrence Berkeley National Laboratory

[53] Pruess K, Oldenburg C, Moridis G (1999) TOUGH2 user's guide, version 2.0

[54] Rjasanow S (2002) Adaptive cross approximation of dense matrices. IABEM 2002, International Association for Boundary Element Methods

[55] Saad Y (1996) Iterative methods for sparse linear systems, vol 20. PWS publishing company Boston

[56] Starks T, Fang J (1982) On the estimation of the generalized covariance function. Mathematical Geology 14(1):57–64

[57] White J (1975) Computed seismic speeds and attenuation in rocks with partial gas saturation. Geophysics 40(2):224–232

[58] Woolfe F, Liberty E, Rokhlin V, Tygert M (2008) A fast randomized algorithm for the approximation of matrices. Applied and Computational Harmonic Analysis 25(3):335–366

[59] Xia J, Chandrasekaran S, Gu M, Li X (2010) Fast algorithms for hierarchically semiseparable matrices. Numerical Linear Algebra with Applications 17(6):953–976

[60] Ying L, Biros G, Zorin D (2004) A kernel-independent adaptive fast multipole algorithm in two and three dimensions. Journal of Computational Physics 196(2):591–626

[61] Zimmerman D (1989) Computationally efficient restricted maximum likelihood estimation of generalized covariance functions. Mathematical Geology 21(7):655–672

[62] Zimmerman D (1989) Computationally exploitable structure of covariance matrices and generalized covariance matrices in spatial models. Journal of Statistical Computation and Simulation 32(1-2):1–15