AGU PUBLICATIONS

Water Resources Research

RESEARCH ARTICLE
10.1002/2016WR020299

Large-scale inverse model analyses employing fast randomized data reduction

Youzuo Lin[1], Ellen B. Le[2], Daniel O'Malley[1], Velimir V. Vesselinov[1], and Tan Bui-Thanh[2]

[1]Earth and Environmental Sciences Division, Los Alamos National Laboratory, Los Alamos, New Mexico, USA, [2]Institute for Computational Sciences and Engineering, University of Texas at Austin, Austin, Texas, USA

Key Points:
- We have developed a computationally efficient, scalable, and implementation-friendly randomized geostatistical inversion method
- Our method is especially suitable for inverse modeling with a large number of observations
- Our method yields a comparable accuracy to other geostatistical inverse methods

Correspondence to:
Y. Lin,
ylin@lanl.gov

Citation:
Lin, Y., E. B. Le, D. O'Malley, V. V. Vesselinov, and T. Bui-Thanh (2017), Large-scale inverse model analyses employing fast randomized data reduction, *Water Resour. Res.*, *53*, 6784–6801, doi:10.1002/2016WR020299.

Received 19 DEC 2016
Accepted 29 JUN 2017
Accepted article online 6 JUL 2017
Published online 12 AUG 2017

Abstract When the number of observations is large, it is computationally challenging to apply classical inverse modeling techniques. We have developed a new computationally efficient technique for solving inverse problems with a large number of observations (e.g., on the order of 10^7 or greater). Our method, which we call the randomized geostatistical approach (RGA), is built upon the principal component geostatistical approach (PCGA). We employ a data reduction technique combined with the PCGA to improve the computational efficiency and reduce the memory usage. Specifically, we employ a randomized numerical linear algebra technique based on a so-called "sketching" matrix to effectively reduce the dimension of the observations without losing the information content needed for the inverse analysis. In this way, the computational and memory costs for RGA scale with the information content rather than the size of the calibration data. Our algorithm is coded in Julia and implemented in the MADS open-source high-performance computational framework (http://mads.lanl.gov). We apply our new inverse modeling method to invert for a synthetic transmissivity field. Compared to a standard geostatistical approach (GA), our method is more efficient when the number of observations is large. Most importantly, our method is capable of solving larger inverse problems than the standard GA and PCGA approaches. Therefore, our new model inversion method is a powerful tool for solving large-scale inverse problems. The method can be applied in any field and is not limited to hydrogeological applications such as the characterization of aquifer heterogeneity.

1. Introduction

The permeability of a porous medium is of great importance for predicting flow and transport of fluids and contaminants in the subsurface [*Carrera and Neuman*, 1986; *Sun*, 1994; *Carrera et al.*, 2005]. A well-understood distribution of permeability can be crucial for many different subsurface applications, such as (1) forecasting production performance of geothermal reservoirs, (2) extracting oil and gas, (3) estimating pathways of subsurface contaminant transport, and many others.

Various hydraulic inversion methods have been proposed to obtain subsurface permeability [*Neuman and Yakowitz*, 1979; *Neuman et al.*, 1980; *Carrera and Neuman*, 1986; *Sun*, 1994; *Kitanidis*, 1997a; *Zhang and Yeh*, 1997; *Carrera et al.*, 2005], of which geostatistical inversion approaches are the most widely used [*Kitanidis*, 1995; *Zhang and Yeh*, 1997; *Kitanidis*, 1997a, 1997b; *Vesselinov et al.*, 2001a]. Geostatistical inversion can be more advantageous than many other subsurface inverse modeling methods in that not only can it provide uncertainty estimates, but it is also suitable for sequential data assimilation [*Vesselinov et al.*, 2001a, 2001b; *Illman et al.*, 2015; *Yeh and Simunek*, 2002]. However, as pointed out in *Vesselinov et al.* [2001b] and *Illman et al.* [2015], one drawback of geostatistical inversion methods is its high-computational cost when the number of observations is large and the model is highly parameterized. In recent years, with the help of regularization techniques [*Tarantola*, 2005; *Engl et al.*, 1996], there is a trend to increase the number of model parameters [*Hunt et al.*, 2007]. It has been suggested that these highly parameterized models have great potential for characterizing subsurface heterogeneity [*Tonkin and Doherty*, 2005; *Hunt et al.*, 2007]. Meanwhile, as the theory and computational tools for subsurface characterization quickly move into the new era of "big data," many existing methodologies are facing the challenge of handling a large number of unknown model parameters and observations. Therefore, it is important to address the theoretical and computational issues of the geostatistical inversion methods.

The costs related to geostatistical inversion methods can be broken into two parts: the computational cost and the memory cost. A number of computational techniques have been proposed to alleviate the costs of computation [*Saibaba and Kitanidis*, 2012; *Liu et al*., 2013; *Ambikasaran et al*., 2013; *Liu et al*., 2014; *Lee and Kitanidis*, 2014; *Lin et al*., 2016] or memory [*Nowak et al*., 2003; *Schoniger et al*., 2012; *Saibaba and Kitanidis*, 2012; *Kitanidis and Lee*, 2014; *Lee and Kitanidis*, 2014]. Some studies targeted both computation and memory costs [*Saibaba and Kitanidis*, 2012; *Kitanidis and Lee*, 2014; *Lee and Kitanidis*, 2014].

One major direction to reduce the computational cost is based on subspace approximations, i.e., solving a small-size approximated problem residing in a lower-dimensional subspace. Several types of subspaces have been utilized, including principle component subspaces [*Kitanidis and Lee*, 2014; *Lee and Kitanidis*, 2014; *Tonkin and Doherty*, 2005], Krylov subspaces [*Lin et al*., 2016; *Liu et al*., 2014; *Saibaba and Kitanidis*, 2012], subspaces spanned by reduced-order models [*Liu et al*., 2014], hierarchical matrix decompositions [*Ambikasaran et al*., 2013; *Saibaba and Kitanidis*, 2012], and active subspaces [*Constantine et al*., 2014].

In geostatistical inversion methods, a majority of the memory is used in storing the matrices, such as the Jacobian matrix and the covariance matrix [*Kitanidis and Lee*, 2014; *Lee and Kitanidis*, 2014]. In situations with a large number of measurements and model parameters, it is prohibitively expensive to store these matrices. To overcome the memory issues, matrix-free or low-rank approximation methods have been developed. Specifically, *Kitanidis and Lee* [2014] and *Lee and Kitanidis* [2014] developed a matrix-free Jacobian to approximate the multiplication of the Jacobian matrix with a vector by finite-difference operations. To further reduce the memory cost associated with storing the covariance matrices, various computational methods have been developed. *Nowak et al*. [2003] developed FFT-based geostatistical inversion method, which is restricted to regular grids, but it only needs to store the first line of the covariance matrix. Ensemble Kalman filters (EnKFs) and related methods have also been proposed for geostatistical inversion to avoid the storage and handling of large covariance matrices [*Schoniger et al*., 2012]. Low-rank matrix approximation-based techniques have also been employed, such as hierarchical decomposition [*Ambikasaran et al*., 2013; *Saibaba and Kitanidis*, 2012] and principal component decomposition [*Kitanidis and Lee*, 2014; *Lee and Kitanidis*, 2014]. Recent work [*Lee et al*., 2016] reported a computationally efficient method to generate a preconditioner by using Generalized Eigenvalue Decomposition and the Sherman-Morrison-Woodbury formula. Another popular computational method to reduce the data size and computational cost is based on the extraction of temporal moments from large data sets [*Yin and Illman*, 2009; *Zhu and Yeh*, 2006; *Nowak and Cirpka*, 2006; *Cirpka and Kitanidis*, 2000].

Randomized algorithms have received a great deal of attention in recent years [*Drineas and Mahoney*, 2016]. They can be seen as either sampling or projection procedures [*Mahoney*, 2011]. Their main idea is to construct a sketch matrix of the input matrix. The sketch matrix is usually a smaller matrix that yields a good approximation and represents the essential information of the original input. In essence, a sketching matrix is applied to the data to obtain a sketch that can be employed as a surrogate for the original data to compute quantities of interest [*Drineas and Mahoney*, 2016]. Randomized algorithms have been successfully applied to various scientific and engineering domains, such as scientific computation and numerical linear algebra [*Le et al*., 2017; *Meng et al*., 2014; *Drineas et al*., 2011; *Lin et al*., 2010; *Rokhlin and Tygert*, 2008], seismic full-waveform inversion and tomography [*Moghaddam et al*., 2013; *Krebs et al*., 2009], and medical imaging [*Huang et al*., 2016; *Wang et al*., 2015; *Zhang et al*., 2012].

Here we present a new geostatistical inversion method using a randomization-based data reduction technique to reduce both the computation and memory costs. We use Gaussian projection to produce the sketching matrix [*Johnson and Lindenstrauss*, 1984] in a matrix and a direct linear solver to obtain the solution of the surrogate problem. A numerical cost analysis will show that our new randomized geostatistical inversion method improves the computational efficiency and reduces memory cost significantly. To evaluate the performance of our new randomized geostatistical inversion method, a test case is presented where a transmissivity field is estimated from observations of hydraulic head. By comparing the results with those obtained using the conventional principal component geostatistical approach, we show that our method significantly reduces the computational and memory costs while maintaining the accuracy of the inversion results.

In the following sections, we first briefly describe the fundamentals of inverse modeling and geostatistical inversion methods (section 2). We then develop and discuss a randomized geostatistical inversion method (section 3). We further elaborate on the computational and memory costs of our method (section 4). We then apply our method to test problems and discuss the results (section 5). Finally, concluding remarks are presented in section 6.

2. Theory

2.1. Inverse Modeling

We consider a transient groundwater flow equation. The forward modeling problem can be written as:

$$\mathbf{h}=f(\mathbf{T})+\varepsilon, \tag{1}$$

where **h** is the hydraulic head, **T** is the transmissivity, $f(\mathbf{T})$ is the nonlinear forward operator mapping from the transmissivity to the hydraulic head, and ε is a term representing additive noise that follows a normal distribution:

$$\varepsilon \sim N(0,\mathbf{R}), \tag{2}$$

where **R** is the error covariance matrix.

The problem of hydrogeologic inverse modeling is to estimate the transmissivity from available measurements. Usually, such a problem is posed as a minimization problem:

$$\hat{\mathbf{m}}=\arg\min_{\mathbf{m}}\left\{||\mathbf{d}-f(\mathbf{m})||_2^2\right\}, \tag{3}$$

where **d** represents a hydraulic head data set and **m** is the vector of model parameters, $||\mathbf{d}-f(\mathbf{m})||_2^2$ measures the data misfit, and $||\cdot||_2$ stands for the L_2 norm. Minimizing equation (3) yields a model $\hat{\mathbf{m}}$ that minimizes the mean-squared difference between observed data and model predictions. However, inverse problems are often severely ill-posed. Moreover, because of the nonlinearity of the forward modeling operator *f*, the solution of the inverse problem may be nonunique and null sets of parameters might provide acceptable inverse solutions. Regularization techniques can be used to address the nonuniqueness of the solution and reduce the ill-posedness of the inverse problem. A general regularization term can be incorporated into equation (3) as [*Vogel*, 2002; *Hansen*, 1998]:

$$\hat{\mathbf{m}}=\arg\min_{\mathbf{m}}\{l(\mathbf{m})\}, \tag{4}$$

$$=\arg\min_{\mathbf{m}}\left\{||\mathbf{d}-f(\mathbf{m})||_2^2+\lambda\mathcal{R}(\mathbf{m})\right\}, \tag{5}$$

where $\mathcal{R}(\mathbf{m})$ is a general regularization term and the parameter λ is the regularization parameter, which controls the amount of regularization in the inversion.

2.2. Geostatistical Inverse Modeling

To account for the errors in the observations and the model, we follow the work of *Kitanidis and Lee* [2014] and *Lee and Kitanidis* [2014], and employ the generalized least squares approach that produces weights to the data misfit and regularization terms in equation (5) using covariance matrices:

$$\hat{\mathbf{m}}=\arg\min_{\mathbf{m}}\{g(\mathbf{m})\}=\arg\min_{\mathbf{m}}\left\{||\mathbf{d}-f(\mathbf{m})||_R^2+\lambda\mathcal{R}(\mathbf{m})\right\}, \tag{6}$$

The weighted data misfit and regularization terms are defined as:

$$||\mathbf{d}-f(\mathbf{m})||_\mathbf{R}^2=(\mathbf{d}-f(\mathbf{m}))^T\,\mathbf{R}^{-1}(\mathbf{d}-f(\mathbf{m})), \tag{7}$$

and

$$\mathcal{R}(\mathbf{m})=(\mathbf{m}-(\mathbf{X}\boldsymbol{\beta}))^T\,\mathbf{Q}^{-1}(\mathbf{m}-(\mathbf{X}\boldsymbol{\beta})), \tag{8}$$

where **X** is a drift (trend) matrix, **Q** is the covariance matrix of the model parameters, and **R** is defined in equation (2).

Using the Jacobian matrix **H** of the forward modeling operator *f* defined as:

$$\mathbf{H}=\left.\frac{\partial f}{\partial \mathbf{m}}\right|_{\mathbf{m}=\tilde{\mathbf{m}}}, \tag{9}$$

the linearized function of the forward modeling operator *f* can be defined as:

$$f(\hat{\mathbf{m}}) \approx f(\bar{\mathbf{m}}) + \mathbf{H}(\hat{\mathbf{m}} - \bar{\mathbf{m}}), \tag{10}$$

where $\hat{\mathbf{m}}$ is the current solution and $\bar{\mathbf{m}}$ is the previous solution. Following *Kitanidis* [1997b] and *Nowak and Cirpka* [2004], the current solution $\hat{\mathbf{m}}$ in equation (10) is given by:

$$\hat{\mathbf{m}} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Q}\mathbf{H}^T \xi, \tag{11}$$

where the vectors of $\boldsymbol{\beta}$ and ξ are solutions to the linear system of equations:

$$\begin{bmatrix} \mathbf{HQH}^T + \mathbf{R} & \mathbf{HX} \\ (\mathbf{HX})^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \xi \\ \boldsymbol{\beta} \end{bmatrix} = \begin{bmatrix} \mathbf{y} - f(\bar{\mathbf{m}}) + H\bar{\mathbf{m}} \\ \mathbf{0} \end{bmatrix}. \tag{12}$$

2.3. Computational Approaches for Solving Geostatistical Inverse Modeling

The most computational and memory intensive parts of solving the geostatistical inverse model in equation (12) are the construction of the Jacobian matrix, **H**, and the matrix products involves the Jacobian, particularly **HQ** in equation (12). Various techniques are employed to address these issues. In *Kitanidis and Lee* [2014] and *Lee and Kitanidis* [2014], the principal component geostatistical approach (PCGA), a seminal computational method in solving the geostatistical inverse model, is proposed and developed. To bypass the expensive explicit construction of the Jacobian matrix, a finite difference scheme is used to approximate a generic Jacobian-vector multiplication of **Hx**, i.e.:

$$\mathbf{Hx} \approx \frac{1}{\delta}[f(\mathbf{x} + \delta\mathbf{x}) - f(\mathbf{x})], \tag{13}$$

where **x** is a n-dimensional vector and δ is the finite difference interval. Furthermore, a low-rank approximation of the covariance matrix **Q** is used

$$\mathbf{Q} \approx \mathbf{Q_k} = \mathbf{Z}^T\mathbf{Z} = \sum_{i=1}^{k} \zeta_i \zeta_i^T, \tag{14}$$

where $\mathbf{Q_k}$ is the rank-k approximation of the covariance matrix **Q**, **Z** is the square root of $\mathbf{Q_k}$ obtained using eigen decomposition, and ζ_i is the ith column vector of **Z**. Based on equations (13) and (14), the expensive matrix-matrix operations of **HQ** and $\mathbf{HQH}^T$ can be reformulated as matrix-vector operations

$$\mathbf{HQ} \approx \mathbf{HQ}_k = \mathbf{H}\sum_{i=1}^{k} \zeta_i \zeta_i^T = \sum_{i=1}^{k} (\mathbf{H}\zeta_i)\zeta_i^T, \tag{15}$$

$$\mathbf{HQH}^T \approx \mathbf{HQ_kH}^T = \sum_{i=1}^{k} (\mathbf{H}\zeta_i)(\mathbf{H}\zeta_i)^T. \tag{16}$$

Another computational technique to reduce the cost of matrix products with the Jacobian matrix is to use a hierarchical representation of the covariance matrix [*Saibaba and Kitanidis*, 2012]. The hierarchical representation of a matrix is accomplished by having split the given matrix into a hierarchy of rectangular blocks and approximating each of the blocks by a low-rank matrix [*Saibaba and Kitanidis*, 2012; *Bebendorf*, 2008; *Borm et al.*, 2003].

With the Jacobian matrix obtained approximately, two main categories of numerical methods have been developed to solve the linear system of equations in equation (12). One is based on direct solvers [*Lee and Kitanidis*, 2014; *Kitanidis and Lee*, 2014] and the other is based on iterative solvers [*Liu et al.*, 2014; *Saibaba and Kitanidis*, 2012; *Nowak and Cirpka*, 2004]. Direct solvers are mostly used in situations when the size of problems ranges from small to medium scale and the system matrix in equation (12) can therefore be explicitly constructed [*Lee and Kitanidis*, 2014; *Kitanidis and Lee*, 2014]. As pointed out in *Lee and Kitanidis* [2014], direct solvers can be used to solve dense linear systems of dimension up to $n \sim \mathcal{O}(10^4)$. For large-scale problems (dimension $n > \mathcal{O}(10^4)$), matrix-free representations can be used, and Krylov-subspace based iterative solvers such as GMRES [*Saad and Schultz*, 1986] or MINRES [*Paige and Saunders*, 1975] are favored over direct methods to solve equation (12) [*Liu et al.*, 2014; *Saibaba and Kitanidis*, 2012].

The use of direct solvers or iterative solvers to solve equation (12) can be memory bound [*Lee and Kitanidis*, 2014; *Kitanidis and Lee*, 2014]. Such a limitation can significantly reduce the computational efficiency when a large number of measurements are available. In particular, it can be observed from equation (12) that the number of equations is of the same order as the number of the measurements. In many subsurface applications, it is increasingly common to calibrate models using a very large number of observations (e.g., $\mathcal{O}(10^7)$ or more). Using the computational techniques discussed above to solve linear system of equations of such a scale is beyond the computability and storage capacity of any method regardless of the choice of direct or iterative solvers. As pointed out in *Kitanidis and Lee* [2014], the computational methodologies discussed so far work best for problems with a modest number of observations. Therefore, there is a need to develop computational methods that allow an efficient solution of equation (12) with a large number of measurements.

There have been some studies addressing this important need for data reduction to reduce computational and storage costs. In *Lee et al.* [2016], PCGA was extended to handle data-intensive inverse problems by constructing a fast preconditioner of the cokriging matrix leading to accelerated iterative matrix inversion. Specifically, using a similar notation as *Lee et al.* [2016], $\Psi=\mathbf{HQH}^T+\mathbf{R}$, $\Phi=\mathbf{HX}$, and $\mathbf{S}=\Phi^T\Psi^{-1}\Phi$, the exact inversion of the system matrix in equation (12) can be written as:

$$\begin{bmatrix} \mathbf{HQH}^T+\mathbf{R} & \mathbf{HX} \\ (\mathbf{HX})^T & \mathbf{0} \end{bmatrix}^{-1} = \begin{bmatrix} \Psi & \Phi \\ \Phi^T & \mathbf{0} \end{bmatrix}^{-1} = \begin{bmatrix} \Psi^{-1}-\Psi^{-1}\Phi\mathbf{S}^{-1}\Phi^T\Psi^{-1} & \Psi^{-1}\Phi\mathbf{S}^{-1} \\ \mathbf{S}^{-1}\Phi^T\Psi^{-1} & -\mathbf{S} \end{bmatrix}. \tag{17}$$

By further employing the Sherman-Morrison-Woodbury formula and Generalized Eigenvalue Decomposition (GED) [*Golub and Van Loan*, 1996], the dominant cost of solving Ψ^{-1} can be significantly reduced by low-rank approximation, while the overall accuracy is well maintained. It has been pointed out in *Lee et al.* [2016] that GED can be efficiently implemented by using either the sequential Lanczos-based method or the parallelized randomized SVD method. *Lee et al.* [2016] concluded that this computational technique can be either used as a direct solver or as a preconditioner for iterative solution of equation (12). In the numerical examples therein, the authors estimated the hydraulic conductivity field of a laboratory-scale sand box using 6 million MRI-scanned tracer concentration observations directly within a reasonable time.

Another popular computational method to reduce the data size and computational cost is based on extraction of temporal moments from large data sets. Researchers have applied such a technique to various data sets such as transient pressure [*Yin and Illman*, 2009; *Zhu and Yeh*, 2006] and concentration breakthrough curves [*Nowak and Cirpka*, 2006; *Cirpka and Kitanidis*, 2000]. Temporal moment based data reduction methods have been shown to be very efficient in reducing the data. Their major drawback, however, is that the system response must be integrable (except when using truncated temporal moments), so this approach cannot be applied to dynamic systems with fluctuating drivers.

In the next section, we describe our approach to reduce the dimensionality of the data while maintaining the accuracy of the inverse results based on randomization theory. We will demonstrate that our method has no restrictions with respect to the mathematical properties of the data.

3. Randomized Geostatistical Inverse Modeling

3.1. Randomized Geostatistical Approach

We develop a new randomized geostatistical inversion method to reduce the data dimensionality and maintain the accuracy of the inversion result. The basic idea of this approach is to construct a sketching matrix $\mathbf{S}$, then replace the data $\mathbf{d}$ with $\mathbf{Sd}$, replace the forward model, $f(\mathbf{T})$, with $\mathbf{S}f(\mathbf{T})$, and the additive noise, ϵ, with $\mathbf{S}\epsilon$; and use the PCGA method for inversion. By multiplying all vectors by $\mathbf{S}$, we reduce the dimensionality ($\mathbf{S}$ has many columns, but not that many rows). At a high level, multiplying by the sketching matrix solves the problems associated with a high-dimensional observation space and the use of the PCGA method solves the problems associated with a high-dimensional parameter space. By combining these methods, we solve both problems. Additionally, if a PCGA implementation is available, the randomized geostatistical approach is extremely easy to implement in high-level languages such as Julia, Matlab and Python (our Julia implementation consists of three lines of code).

The misfit function of the randomized geostatistical inversion is given by

$$\hat{\mathbf{m}} = \arg\min_{\mathbf{m}} \left\{ ||\mathbf{Sd} - \mathbf{S}f(\mathbf{m})||_2^2 + \lambda \mathcal{R}(\mathbf{m}) \right\}, \tag{18}$$

where $\mathbf{S} \in \mathbb{R}^{k_{\mathrm{red}} \times n}$ is the sketching matrix and $k_{\mathrm{red}} \ll n$ is a tunable reduced dimension. The sketching matrix is also referred to as a Johnson-Lindenstrauss Transform [*Kane and Nelson*, 2014; *Woodruff*, 2014; *Mahoney*, 2011; *Dasgupta et al.*, 2010; *Clarkson and Woodruff*, 2009; *Sarlos*, 2006]. With the new misfit function defined in equation (18) and following a similar derivation as in the previous section, the following randomized linear system of equations is obtained

$$\begin{bmatrix} \mathbf{SHQH}^T\mathbf{S}^T + \mathbf{R} & \mathbf{SHX} \\ (\mathbf{SHX})^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \xi \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbf{S}(\mathbf{y} - f(\bar{\mathbf{m}}) + \mathbf{H}\bar{\mathbf{m}}) \\ 0 \end{bmatrix}. \tag{19}$$

At this point, we need to specify *R*. As discussed above, the forward model can be formulated as:

$$\mathbf{Sh} = \mathbf{S}f(\mathbf{T}) + \mathbf{S}\varepsilon, \tag{20}$$

we can therefore derive the data error covariance matrix **R** in equation (19) as:

$$\mathbf{R} = \mathbb{E}[\mathbf{S}\varepsilon(\mathbf{S}\varepsilon)^T] = \mathbf{S}\mathbb{E}[(\mathbf{h} - f(\mathbf{T}))(\mathbf{h} - f(\mathbf{T}))^T]\mathbf{S}^T = \mathbf{SRS}^T. \tag{21}$$

With the randomized linear system of equations given in equation (19) and the covariance matrix in equation (21), we will have the corresponding solution iterate, which is similar to equation (11)

$$\hat{\mathbf{m}} = \mathbf{X}\beta + \mathbf{QH}^T\mathbf{S}^T\xi. \tag{22}$$

Correspondingly, the posterior covariance matrix can be derived similar to *Kitanidis and Lee* [2014]

$$\mathbf{V} = \mathbf{Q} - \mathbf{F}, \tag{23}$$

where

$$\mathbf{F} = \mathbf{XQ}_{\beta\beta}\mathbf{X}^T + \mathbf{QH}^T\mathbf{S}^T\mathbf{Q}_{\mathbf{yy}}^{-1}\mathbf{SHQ}^T. \tag{24}$$

3.2. Selection of the Sketching Matrix

Random projection is one class of methods for low-rank matrix approximation [*Yang et al.*, 2016; *Mahoney*, 2011]. The idea of random projection is based on the Johnson-Lindenstrauss Lemma [*Johnson and Lindenstrauss*, 1984]. In particular, *Johnson and Lindenstrauss* [1984] pointed out that random projection yields the property of subspace embedding. *Johnson and Lindenstrauss* [1984] further provided a strategy to generate the random projection matrix. With the significant increase of data volumes, recent years have witnessed an explosion of research on so-called randomized numerical linear algebra algorithms, which use the power of randomization in order to perform standard matrix computations [*Yang et al.*, 2016; *Drineas and Mahoney*, 2016; *Iyer et al.*, 2016; *Mahoney*, 2011; *Avron et al.*, 2010].

3.2.1. Subspace Embedding and Johnson-Lindenstrauss Lemma

Subspace embedding is the core of all randomization-based methods. It is a particular property of any randomization projection built upon the definition of column space, which is provided in the appendix. The randomized projection matrix (or sketching matrix) **S** is critical in reducing data dimensionality and preserving solution accuracy. The role of the sketching matrix can be seen as preconditioning the input data to spread out or uniformize the information contained in the data [*Drineas and Mahoney*, 2016]. With an appropriately selected sketching matrix, the solution to equation (18) yields a highly accurate approximation to the original problem in equation (3).

In *Johnson and Lindenstrauss* [1984], theoretical work is provided (through the proof of the Johnson-Lindenstrauss Lemma) to demonstrate the existence of a projection matrix (sketching matrix) that allows subspace embedding. *Johnson and Lindenstrauss* [1984] described the subspace embedding and further proved that a specially constructed sketching matrix **S** exists that allows to project with high (asymptotic) probability, *N* points in high-dimensional space to a much lower dimension without losing essential information. We provide the visualization of the Johnson-Lindenstrauss bounds to better illustrate the relation between the number of observations and the value of k_{red} with respect to the distortion rate ϵ, which is defined in definition A.2 of the Appendix A. Specifically, Figure 1a shows a plot of the minimum k_{red} versus the number of observation, *n*, for different distortion rates ϵ. Figure 1b shows a plot of the minimum

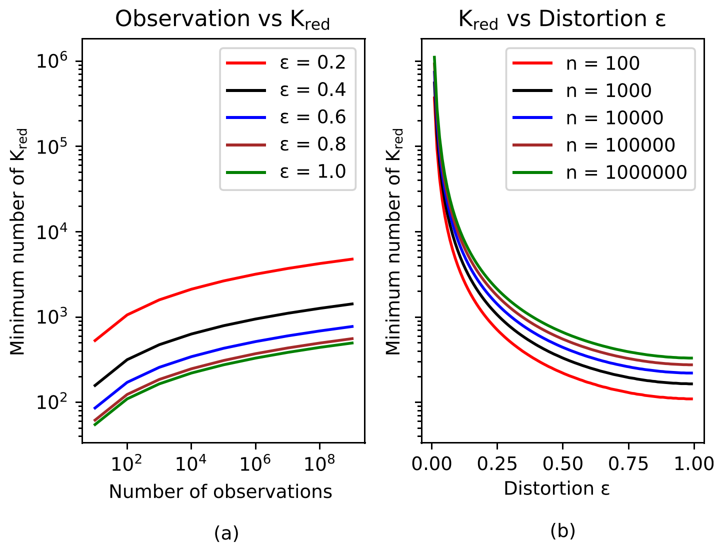


Figure 1. Johnson-Lindenstrauss bounds: (a) minimum k_{red} versus the number of observation, n, for different distortion rates ϵ; (b) minimum number of k_{red} versus the distortion rate ϵ for different number of observations, n. Clearly, an increase in n requires an increase in k_{red} to preserve a given distortion rate ϵ.

number of k_{red} versus the distortion rate ϵ for different number of observations. From Figure 1a, we can see the larger the number of observations, the larger the value of k_{red} required to preserve a given distortion rate. Similarly, in Figure 1b, with the number of observations fixed, the larger the value of k_{red}, the smaller the distortion rate becomes.

3.2.2. Construction and Selection of Sketching Matrix

Practically, various methods have been proposed to construct the sketching matrix, **S** [*Drineas and Mahoney*, 2016; *Mahoney*, 2011]. The most important criteria for construction and selection of the sketching matrix are based on computational complexity, the ability to apply it to arbitrary data, and the quality of data reduction.

Sketching matrices can be categorized into two types, those based on projection and those based on sampling [*Drineas and Mahoney*, 2016; *Mahoney*, 2011]. Sampling-based sketching matrices are easy to implement. However, these methods are data dependent, and, therefore, may not provide robust reduction performance. Projection-based sketching matrices can be applied to arbitrary data. Two of the widely used sketching matrices are obtained using Gaussian projection or a randomized Hadamard transform. The Gaussian projection sketching matrix can be represented by independent identically distributed (i.i.d.) Gaussian random variables, i.e., matrix values drawn from the standard Gaussian distribution [*Drineas and Mahoney*, 2016]. The randomized Hadamard transform sketching matrix is represented by a product of two matrices, a random diagonal matrix with $+1$ or -1 on each diagonal entry, each with probability 1/2, and the Hadamard-Walsh matrix [*Ailon and Chazelle*, 2010]. There are other construction methods of sketching matrices designed for specific cases. In *Avron et al.* [2010], the construction of the sketching matrix is the product of a random diagonal matrix and the discrete cosine transform. In *Iyer et al.* [2016], a specific sketching matrix is designed for solving large-scale and sparse systems. In this work, we employ the randomization matrix scheme similar to the one used in *Le et al.* [2017] considering its simplicity to implement, its independence from the data, and stronger conditioning properties than other sketching matrices [*Drineas and Mahoney*, 2016]. Hence, the Gaussian random projection matrix $\mathbf{S} \in \mathbb{R}^{k_{red} \times n}$ can be represented by

$$\mathbf{S} = \frac{1}{\sqrt{n}} \mathbf{G}, \tag{25}$$

where **G** is sampled i.i.d. from $\mathcal{N}(0, 1)$.

3.3. Randomized Geostatistical Inversion Algorithm

To summarize our new randomized geostatistical inversion algorithm, we provide a detailed description of the algorithm below.

Algorithm 1: Randomized Geostatistical Approach (RGA)

```
Require: $k_{\text{red}}$, $\xi_0$, and $\boldsymbol{\beta}_0$, $IterCount_{\max}$;
Ensure: $\mathbf{m}^{(k)}$
 1: Initialize $found$=false;
 2: Initialize $k_{\text{red}}$, $\xi_0$, and $\boldsymbol{\beta}_0$;
 3: Generate the sketching matrix according to section 3.2;
 4: Obtain the data-reduced problem according to equation (20);
 5: Update the data covariance matrix $\mathbf{R}$ according to equation (21);
 6: while $\{(\textbf{not } found)$ and $(IterCount < IterCount_{\max})\}$ do
 7:     Solve for the solution of the reduced linear system of equations in equation (19);
 8:     Update the iterate according to equation (22);
 9:     if {Stopping criterion are satisfied} then
10:         $found$=true;
11:         Return with current iterate $\hat{\mathbf{m}}$;
12:     end if
13: end while
```

Both direct linear solvers and iterative solvers can be utilized to solve the reduced linear system of equations in equation (19). Considering that, in most cases, the reduced linear system of equations usually yields relatively small system matrices, we use a direct solver to solve the reduced linear system of equations.

4. Computational and Memory Cost Analysis

To better understand the cost of our new randomized geostatistical inversion algorithm, we provide both the computational and memory cost analysis of our method. We assume that the number of model parameters is $\tilde{m}$, the number of observations is $\tilde{n}$, hence the size of the Jacobian matrix $\mathbf{H} \in \mathbb{R}^{\tilde{n}\times\tilde{m}}$ and the covariance matrix $\mathbf{Q} \in \mathbb{R}^{\tilde{m}\times\tilde{m}}$. We also denote the rank of the sketching matrix by k_{red}. The drift matrix $\mathbf{X} \in \mathbb{R}^{\tilde{m}\times\tilde{p}}$. As a reference method, we select the method of PCGA, which is developed in *Kitanidis and Lee* [2014] and *Lee and Kitanidis* [2014].

4.1. Computational Cost

Considering that most of the numerical operations in Algorithm 1 involve only matrix and vector operations, we use the floating point operations per second (FLOPS) and the big-$\mathcal{O}$ notation to quantify the computational cost [*Golub and Van Loan*, 1996]. In numerical linear algebra, basic linear algebra subprograms (BLAS) are categorized into three levels. Level-1 operations involve an amount of data and arithmetic that is linear in the dimension of the operation. Those operations involving a quadratic amount of data and a quadratic amount of work are Level-2 operations [*Golub and Van Loan*, 1996]. Following this notation and given a vector of length n and a matrix size of $n \times n$, vector dot-product, addition and subtraction are examples of BLAS Level-1 operations (BLAS 1). They involve $\mathcal{O}(n)$ amount of data and $\mathcal{O}(n)$ amount of arithmetic operations. Matrix-vector multiplication is a BLAS Level-2 operation and it involves $\mathcal{O}(n^2)$ amount of data and $\mathcal{O}(n^2)$ amount of arithmetic operations. Matrix-matrix multiplication is a BLAS Level-3 operation and it involves $\mathcal{O}(n^2)$ amount of data and $\mathcal{O}(n^3)$ amount of arithmetic operations.

First, we provide the computational cost of the PCGA method followed by the computational cost of RGA method, PCGA employs a matrix-free iterative approach (equations (13–16)) for solving the cokriging system in equation (12). The total computational cost is [*Lee and Kitanidis*, 2014]:

$$\text{COMP}_{\text{PCGA}} \approx \mathcal{O}(\tau \tilde{n}\, k), \tag{26}$$

where τ is the iteration number, and k is the rank of the approximated covariance matrix Q_k in equation (14). Because of the randomization technique used in the RGA method, the size of the system matrix in equation (19) has been significantly reduced. Therefore, direct linear solver such as QR factorization is feasible for solving the linear system of equations in equation (19) [*Golub and Van Loan*, 1996]. Using QR-factorization to solve equation (19), the computational cost of the RGA method is given by:

$$\text{COMP}_{\text{RGA\_Direct}} \approx \mathcal{O}((k_{\text{red}}+\tilde{p})^3)+\mathcal{O}((k_{\text{red}}+\tilde{p})^2), \tag{27}$$

where the first term corresponds to the cost of QR factorization, and the second term is the cost to form the right-hand side and the cost to perform the back substitution. As an alternative, we can also employ the matrix-free iterative approach to solve equation (19) as done in the PCGA method. The resulting computational cost then is given by:

$$\text{COMP}_{\text{RGA\_Iterative}} \approx \mathcal{O}(\tau\, k\, k_{red}). \tag{28}$$

By comparing to equations (27) and (28), we observe that the RGA method is more efficient. However, it should be noted here that this analysis only explores the computational cost of the linear algebra associated with performing an iteration of the inverse analysis. The overall computational cost should also include the computational cost for solving the forward model repeatedly. However, when PCGA is used and $\tilde{n}$ is sufficiently large, the computational cost associated with the linear algebra operations dominate. By reducing the cost of the linear algebra operations, RGA results in a situation where the computational cost of repeatedly solving the forward model is the dominant cost in the inverse analysis.

4.2. Memory Cost

Both the RGA and PCGA methods rely on dense matrix storage. Hence, the major memory cost is that used to store the matrices. Out of all these matrices, the largest matrices required to be stored are Z and HZ in equations (14) and (16) for the PCGA method or the matrix in equation (19) for our method. The dimensions of system matrices Z and HZ are $Z \in \mathcal{R}^{\tilde{m}\times k}$ and $HZ \in \mathbb{R}^{\tilde{n}\times k}$. Hence, the total memory cost of the PCGA method will be:

$$\text{MEM}_{\text{PCGA}} \approx \mathcal{O}((\tilde{m}+\tilde{n})\cdot k). \tag{29}$$

Similarly, we can also calculate the dimension of the corresponding linear system of equations in equation (19) for our method. Provided with a rank k_{red} sketching matrix, the dimension of the resulting linear system of equations will be $\mathcal{R}^{(k_{\text{red}}+\tilde{p})\times(k_{\text{red}}+\tilde{p})}$. Hence, the total memory cost including both the sketching matrix and linear system of equations is:

$$\text{MEM}_{\text{RGA}} \approx \mathcal{O}((k_{\text{red}}+\tilde{p})\times(k_{\text{red}}+\tilde{p})+\tilde{n}\times k_{red}). \tag{30}$$

Comparing equation (30) to equation (29), we see that the memory cost of our method is approximately $\kappa \approx k_{\text{red}}/k$ of that of the PCGA method. Practically, the sketching matrix can be generated "on-the-fly." It means that instead of constructing the sketching matrix explicitly, we can generate the elements of the sketching matrix implicitly, therefore, the storage of the sketching matrix can mostly be saved, which results in a cost of

$$\text{MEM}_{\text{RGA}} \approx \mathcal{O}((k_{\text{red}}+\tilde{p})\times(k_{\text{red}}+\tilde{p})). \tag{31}$$

Despite the considerably lower memory cost, the implementations of our method on top of PCGA are straightforward.

5. Numerical Results

In this section, we provide numerical examples to demonstrate the efficiency of our new randomized geostatistical inversion algorithm. A synthetic model study using transient groundwater flow is developed where the "observed" hydraulic heads were taken from a solution of the groundwater equation using a reference transmissivity field with the addition of noise. To have a comprehensive comparison, we provide four sets of tests. In section 5.1, we provide a convergence test of our method. In section 5.2, we report the performance of our method as a function of the number of rows, k_{red}, in the sketching matrix. In section 5.3, we test the robustness of our method with a view on the randomness of the sketching matrix. In section 5.4, we test our method on inverse problems with an increasing number of measurements up to 10^7. An

Table 1. Settings for the Calibration Used for the RGA Calibration

Parameter	Value(s)	Notes
Observation noise	$N(\mu=0, \sigma=0.01)$	
Observations per well per pumping test	1000	
Number of observation/pumping wells	4–10	See Figures 3 and 9 for locations
Prior covariance for **T**	$\sigma \approx 4.5,\ \lambda=0.2$	Exponential model
True heterogeneity of $\log_{10}\mathbf{T}$	$\mu=0.5,\ \sigma=1/2$	Fractal model

important parameter in the PCGA method is the number of principal components (rank), R_{PCGA}. In all the tests using the PCGA method, we set $R_{\text{PCGA}}=120$.

We select Julia as our programming tool because of its efficiency and simplicity. Julia is a high-level programming language designed for scientific computing [*Bezanson et al.*, 2014]. The Julia code for our RGA algorithm is available as a part of the open-source release of the Julia version of MADS (Model Analysis and Decision Support) at "http://mads.lanl.gov" [*Vesselinov et al.*, 2015]. The methods of the QR factorization and fundamental BLAS operations are all implemented using the system routines provided in the Julia packages. As for the computing environment, we run the first three sets of tests on a computer with 40 Intel Xeon E5–2650 cores running at 2.3 GHz, and 64 GB memory, and the last set of tests on a higher-memory machine with 64 AMD Opteron 6376 cores running at 2.3 GHz and 256 GB of memory.

The stopping criterion is an important issue for any iterative method including our method. We use the following two stopping criteria

$$||\mathbf{m}^{(k+1)}-\mathbf{m}^{(k)}||_2^2/||\mathbf{m}^{(k)}||_2^2 \leq \text{TOL}, \tag{32}$$

and

$$Iter \leq Iter_{MAX}, \tag{33}$$

where $\text{TOL}=10^{-6}$, and $Iter$ is the iteration count. $Iter_{MAX}=50$ is the maximum number of iterations. If either equation (32) or equation (33) is satisfied, the iteration procedure is stopped and convergence is declared.

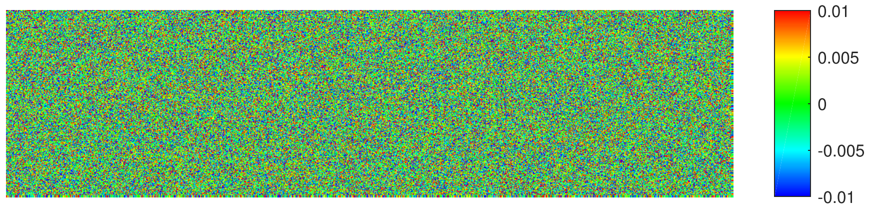


Figure 2. Illustration of the randomization matrix used in the presented analyses with dimension $k_{\text{red}}\times n=256\times16,000$. The elements of the randomization matrix follow equation (25), a scaled normal distribution with mean 0 and standard deviation 1. Because of the width limitation of the page, we only show the first 1000 columns of the randomization matrix.

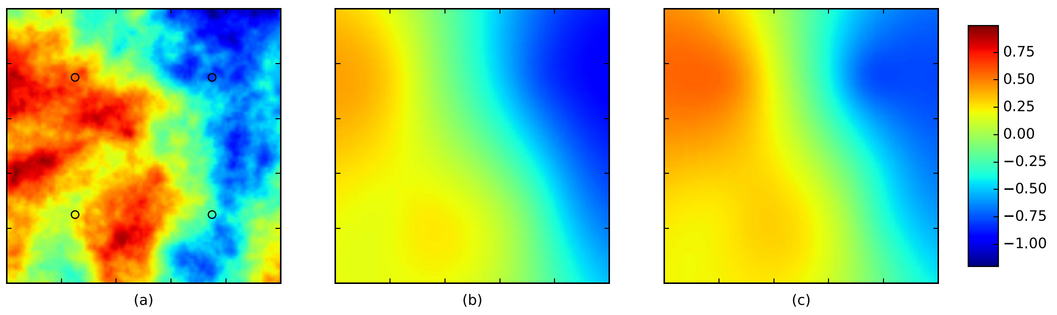
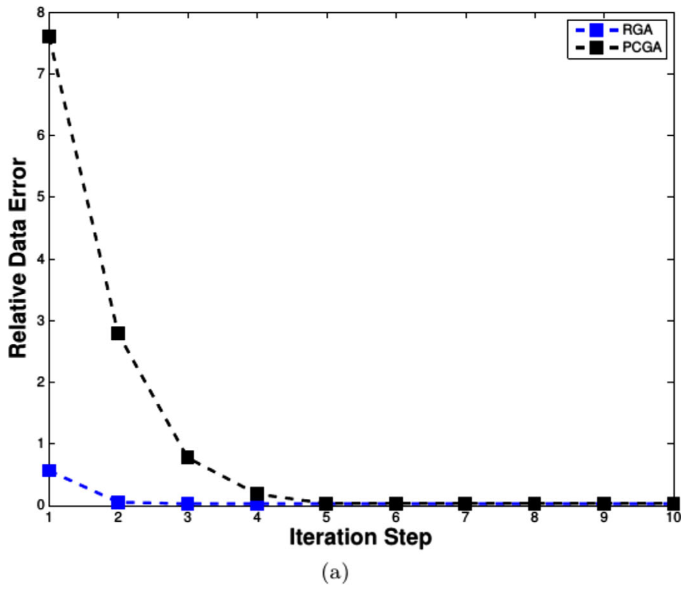


Figure 3. Synthetic log-transmissivity field (a) with variance $\sigma^2_{\mathbf{m}}=0.5$ and power $\beta_{\mathbf{m}}=-3.5$. Hydraulic conductivity and hydraulic head observation locations are indicated with circles. The results of the inverse modeling solved by (b) PCGA and (c) our RGA algorithm are shown. They are visually similar to each other. The RME values of the results in Figures 3b and 3c are 0.28 and 0.33, respectively. Hence, our RGA method yields comparable result to that obtained using the PCGA method.



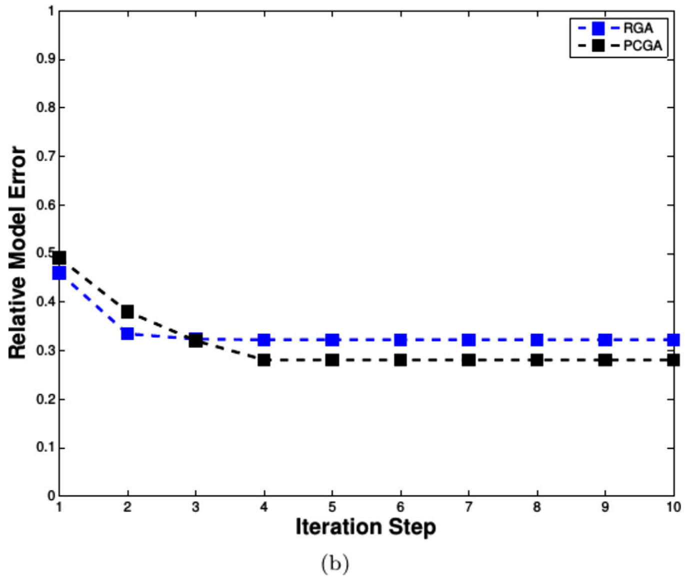


Figure 4. Convergence of the PCGA (in black) and our RGA (in blue) algorithms in terms of iteration steps. The rates of convergence for these two methods are very close to each other. However, the computational time of two methods to reach convergence are very different. In this case, PCGA converged for about 32,000 s, while RGA convergence took only 1020 s. The RGA speed-up is about 31 times.

5.1. Test of the Convergence

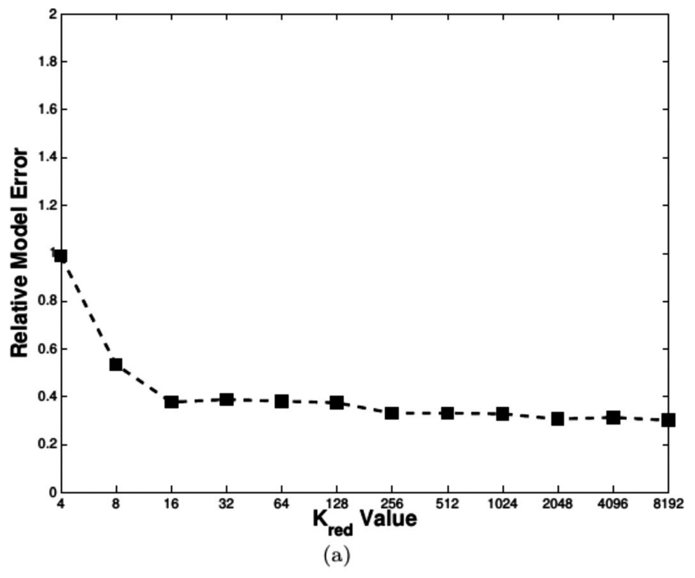
In our first numerical example, we test the convergence of our new method. The reference model is solved on a grid containing 2-D 100 $\times$ 100 head nodes and a total of 20,200 model parameters (100 $\times$ 101 log-transmissivities along the *x* axis, 101 $\times$ 100 log-transmissivities along the *y* axis). Table 1 describes the model setup in more detail. We generate a ground truth, which is shown in Figure 3a. We utilize the variance ($\sigma^2_\mathbf{m}$) and an exponent ($\beta_\mathbf{m}$—related to the fractal dimension of the field and the power law of the field's spectrum) to characterize the heterogeneity of the considered fields [*Peitgen and Saupe*, 1988]. In this example, we set the variance to $\sigma^2_\mathbf{m}=0.5$ and the power exponent to $\beta_\mathbf{m}=-3.5$. The total number of measurements generated in this test is 16,000, which come from running the transient simulation to simulate pumping tests at each well (a total of four tests) and acquiring data at all four locations (four sets of data for each test). In each test, 1000 hydraulic head observations are recorded at each well.

We illustrate one of the randomization matrices in Figure 2. The dimension of the randomization matrix is $k_{\text{red}}\times n=256\times 16,000$. The elements of the randomization matrix follow equation (25), a scaled normal distribution with mean 0 and standard deviation 1. Because of the width limitation of the page, we only show the first 1000 columns of the randomization matrix. We set the color scale of Figure 2 in the range $(-0.01, 0.01)$ to enhance the visualization of the randomized matrix.

Figure 3b illustrates the result using the PCGA method and the Figure 3c shows the results for our method. Compared to the true model in Figure 3a, our method obtains a good result, representing both the high and low log-transmissivity regions. Visually, our method yields a comparable result to the one obtained using the PCGA method in Figure 3b.

To further quantify the error of the two inverse modeling methods, we calculate both the relative-model-error (RME) and relative-data-error (RDE) of the inversion results

$$\text{RME}(\mathbf{m})=\frac{||\mathbf{m}-\mathbf{m}_{\text{ref}}||_2}{\sqrt{m}\times \text{std}_{field}}, \tag{34}$$



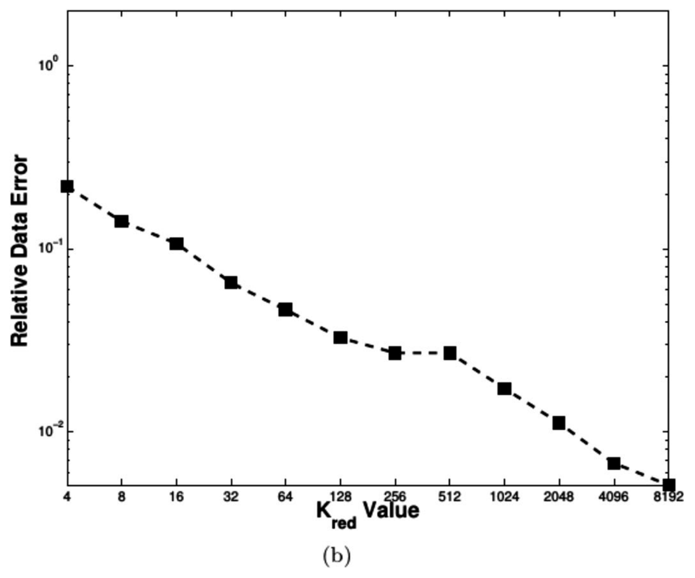


Figure 5. (a) RME and (b) RDE curves as defined in equations (34) and (35), respectively. For k_{red} increasing from 4 to 256, there is a significant decrease in RME. For $k_{red} \geq 256$, the RME curve starts to level off while RDE curve still reduces.

$$\mathrm{RDE}(\mathbf{d}) = \frac{||\mathbf{d}-\mathbf{d}_{rec}||_2}{\sqrt{n}\times \mathrm{std}_{noise}}, \qquad (35)$$

where $\mathbf{m}$ is the inverted transmissivity field and $\mathbf{m}_{ref}$ is the reference transmissivity field, m is the size of the model, and std_{field} is the prior standard deviation of the field, n is the size of the data, std_{noise} is the standard deviation of the additive noise, $\mathbf{d}$ are the simulated data and $\mathbf{d}_{rec}$ are the observations used for inversion.

We provide a plot of the rate of convergence of the PCGA method and our RGA method in Figure 4. We observe that both our method and the PCGA yield a very similar rate of convergence as a function of the number of iterations steps. At each iteration, the methods yield similar relative data error and model error values. After convergence, the RME values of our RGA method and PCGA method are 0.33 and 0.28, respectively. Therefore, together with the inversion result in Figure 3, this demonstrates that our RGA method yields a comparable accuracy to the PCGA method in a situation where both methods can be applied. We note, however, that one of the main benefits of the RGA method is that it can be applied in situations with a very large number of observations and yield accurate results and efficient performance. In this example, it took RGA only about 1300 s to converge, spending 1210 s on forward modeling and only 0.03 s on inversion.

5.2. Test on the Rank of the Sketching Matrix

The rank of the random sketching matrix k_{red} is critical to the accuracy and efficiency of our RGA method. In this section, we test our algorithm using sketching matrices with different rank values. The values of k_{red} used in the problem are 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, and 8192.

In Figure 5, we provide the RME value and the RDE value as a function of k_{red}. We notice that, the larger k_{red} becomes, the smaller the error of the inversion. In addition, there is a significant decrease in the RME values with increasing k_{red} for low values of k_{red}, which means that the inversion results are improving. In particular, the inversion results are completely off when k_{red} is 4. When $k_{red}=256$, the RME starts to level off while the RDE continues to decrease. Even though the data misfit of the inversion becomes smaller with increasing k_{red}, hardly any useful information is introduced into the results.

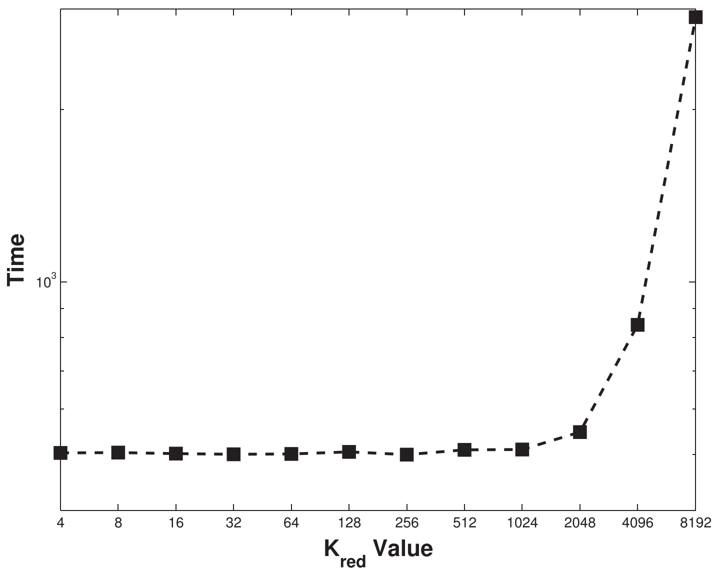


Figure 6. CPU time cost as a function of k_{red}. The CPU time is quite stable around 500 s for $k_{red} \leq 1024$. The time cost dependency on k_{red} can be explained by the fact that when k_{red} is relatively small, the CPU time is mostly dominated by the forward modeling operation, while as k_{red} increases, the linear solver for the solution of the system in equation (19) starts to dominate.

Figure 6 shows the corresponding wall time cost for different values of k_{red}. It can be observed that the time is quite stable around 500 s until $k_{red}=2048$, where the CPU time increases to about 550 s. When $k_{red}=8192$, the CPU time cost is 2902 s. This can be explained by the fact that when k_{red} is relatively small, the CPU time is mostly dominated by the forward modeling operations; while as k_{red} increases, the linear solver for the system in equation (19) starts to dominate.

From this test, we conclude that the optimal selection of the k_{red} value ranges from 256 to 1024 considering factors including model error, data misfit, as well as the corresponding time cost. In general, when choosing the value of k_{red}, one would want to choose a value that is large enough to produce accurate results (i.e., large enough to be in the flat portion of Figure 5a) and small enough so that the method is computationally efficient (i.e., small enough to be in the flat portion of Figure 6).

5.3. Test on the Randomness of the Sketching Matrix

Because of the random nature of our method, the resulting inversion can fluctuate among different realizations of the sketching matrix. In this test, we provide the inversion results and corresponding analysis using various sketching matrices. We use the same model set up as in Test 5.1. We generate 20 different realizations of the sketching matrix with $k_{red}=256$. Each of them is drawn from the Gaussian distribution with mean 0.0 and variance 1.0 according to equation (25). After convergence of each inversions, we calculated the relative data errors and relative model errors according to equations (34) and (35). The results in Figure 7 show that 19 out of 20 inversion runs converged. This shows that the random nature of the sketching matrix may lead to convergence failure in certain cases. Therefore, a safeguard will be necessary to prevent unconverged inversion results. One option is to use cluster analysis on the scatter plot in Figure 7 in order to detection inversion results that did not properly converge. However, this option can be computationally more expensive in that we need to postpone our decision until after all the computation is completed. An alternative is to access convergence directly from the inversion results

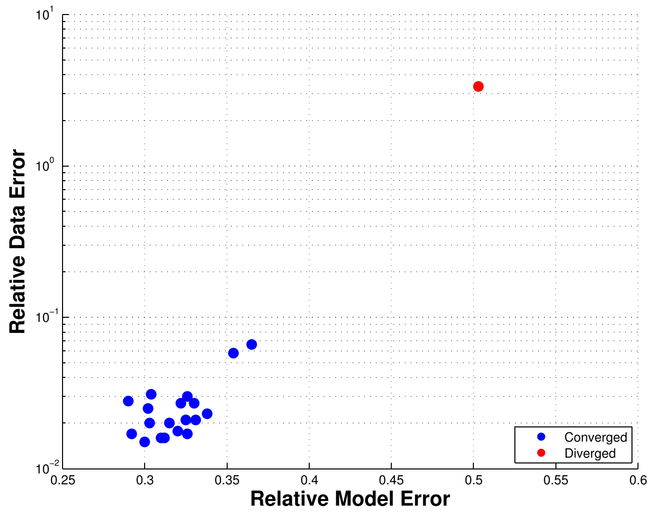


Figure 7. Plot of relative data errors versus relative model errors according to equations (34) and (35) using 20 different realizations of sketching matrices. The sketching matrices leading to converged inversion results are shown in blue. In contrast, those leading to diverged results are shown in blue.

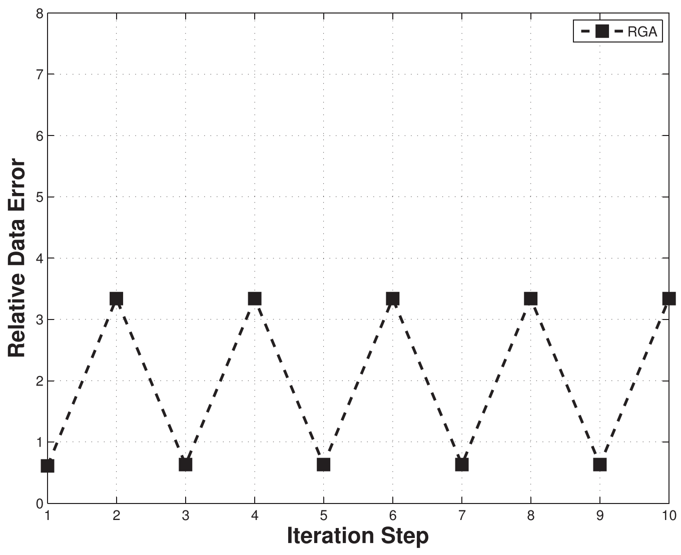


Figure 8. Plot of convergence corresponding to the failure scenario indicated by the red scatter shown in Figure 7. The convergence plot can be utilized as a safeguard during the computation to discard the realization of a sketching matrix that will lead to a diverged result.

using the convergence of the iteration. Specially, we provide a plot of RDE as a function of iteration number for the nonconverged inversion run in Figure 8. We observe that the RDE fluctuates between two values, and does not decrease as expected. Hence, this type of convergence plot can be utilized as a safeguard during the computation to avoid sketching matrices that lead to nonconverged inversion results. From this test, we conclude that even though a small number of realizations of the sketching matrix may lead to inappropriate inversion results, in most cases our method yields accurate results.

5.4. Test on the Number of Observations

To better understand the scalability of our method, we test RGA on a set of inverse problems that have an increasing number of observations. Specifically, we test our algorithm on inverse problems where the number of observations is equal to 2.56×10^5, 6.25×10^5, 1.296×10^6, 2.401×10^6, 4.096×10^6, 6.561×10^6, and 1.0×10^7. As before, the observations come from simulating a series of pumping tests and recording "observations" at a number of monitoring wells. For each observation well, there are 1000 observations for each pumping test. The increasing number of observations comes from increasing the number of pumping tests and the number of observation wells. For example, the case with 2.56×10^5 observations involves 16 pumping tests and 16 observations wells while the case with 1.0×10^7 involves 100 pumping tests and 100 observation wells. The reference transmissivity field is same as the one as in Figure 3a. The value of k_{red} is again set to 256.

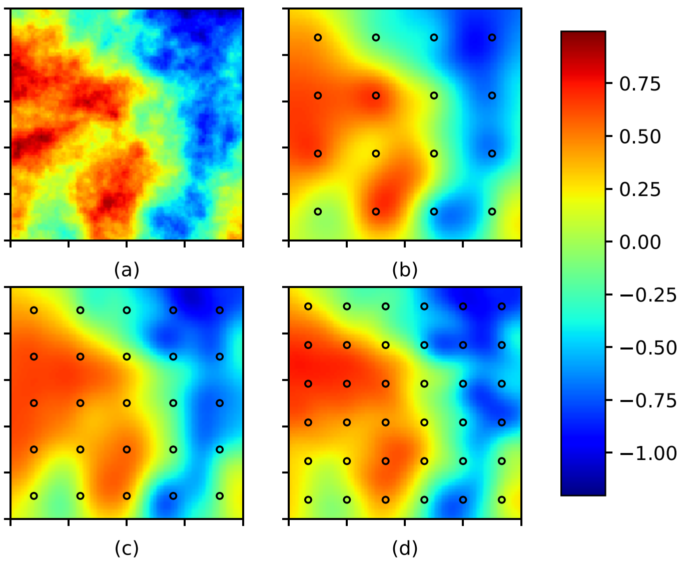


Figure 9. (a) The "true" field and (b) inversion results of our RGA method with different numbers of observations including 2.56×10^5, (c) 4.096×10^6, and (d) 1.0×10^7. Our RGA method yields reasonable inversion results when the size of the data sets becomes massive. As a comparison, the PCGA method fails in all three cases of (b–d) because of the insufficient memory.

Through our analysis on memory cost in section 4.2, we observe that both our RGA method and the PCGA method can be comparable if we construct the sketching matrix explicitly. However, our RGA method can be more memory efficient than the PCGA method when we generate the sketching matrix "on-the-fly." Using RGA, we are able to perform the inverse analysis with 10 million observations. We tested our RGA method on all the problem sizes mentioned above and provide the corresponding results where the number of observations is 2.56×10^5, 4.096×10^6, and 1.0×10^7 in Figure 9. We notice that our RGA method yields reasonable inversion results even when the size of the data sets becomes massive. The RME values of the inversion results in Figures 9b–9d are

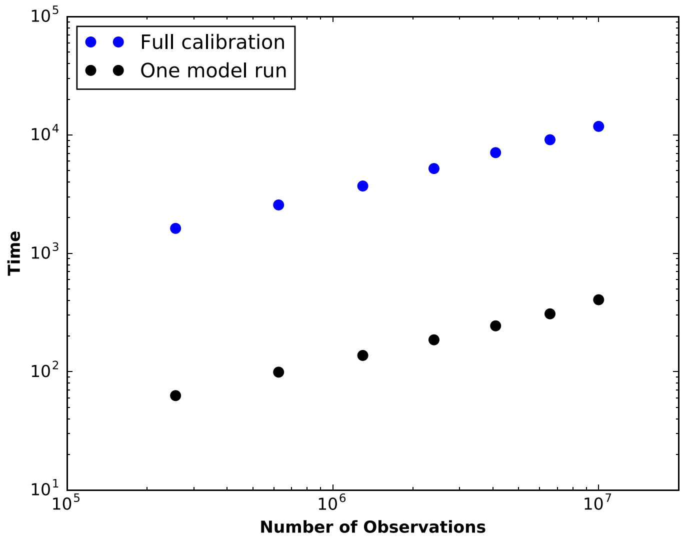


Figure 10. Wall-clock times to perform the model calibration with our RGA method and to perform a single model run. These times are shown for inverse analyses where the number of observations is 2.56×10^5, 6.25×10^5, 1.296×10^6, 2.401×10^6, 4.096×10^6, 6.561×10^6, and 1.0×10^7. For all these analyses, which vary over two orders of magnitude, the time to perform the full model calibration takes 28 times as long as performing a single model run and this could be reduced further with more CPU cores. The cost of an individual run increases because more the time that is simulated increases to account for a larger number of pumping tests.

0.26, 0.23, and 0.25, respectively. The availability of more measurements, in general, may lead to a better inversion. However, in our tests, data can be significantly redundant. Adding more data without increasing k_{red} may not result in an improved inverse model. This explains what we observe in Figure 9, where the quality of the inverse models is similar. Finally, as a comparison, the PCGA method fails in all three cases of Figures 9b–9d because of the insufficient memory. This is due in part to our use of off-the-shelf matrix data structures and matrix-vector multiplication operators for the PCGA implementation and could be alleviated with the use of custom matrix data structures and matrix-vector multiplication operators.

We also provide the wall time costs of our method with different numbers of observations in Figure 10. Figure 10 shows both the wall time to perform the model calibration with RGA and the wall time to perform a single model run. Independent of problem size, the time to perform the full model calibration takes $\sim$28 times as long as performing a single model run and this could be reduced further with more CPU cores. Also we notice that the computational cost of RGA scales well with the number of observations. Through this test, we conclude that our method can more readily calibrate models with a large number of observations compared to the PCGA method when off-the-shelf matrices and matrix-vector operations are used.

6. Conclusion

We have developed a computationally efficient, scalable, and implementation-friendly randomized geostatistical inversion method, which is especially suitable for inverse modeling with a large number of observations. Our method, which we call the randomized geostatistical approach (RGA), is built upon the principal component geostatistical approach (PCGA). To overcome the issues of excessive memory and computational cost that arise when dealing with a large number of observations, we incorporated a randomized sketching matrix technique into PCGA. The randomization method can be seen as a data-reduction technique, because it generates a surrogate system that has a much lower dimension than the original problem.

Through our computational cost analysis, we show that this matrix sketching technique reduces both the memory and computational costs significantly. Compared to the PCGA method, our RGA method yields a much smaller problem to solve when computing the next step in the iterative optimization process, therefore reducing both the memory and computational costs. We demonstrate through our numerical example that our RGA method yields rather efficient computational and memory costs, which can be scaled with the information content of the applied observation data in the inverse process (the computational and memory costs of the RGA inverse analyses do not scale directly with the size of the observation data). It is reasonable to conclude that the efficiency improvement can be significant when the size of the data set increases.

In summary, with an ever-increasing amount of data being assimilated into hydrogeologic models, there is a need to develop an inverse method that is able to handle a large number of observations. Our RGA method addresses this need. The contribution of our work is to incorporate a randomized numerical linear algebra technique into the PCGA method. Through both a computational cost analysis and numerical tests, we show theoretically and numerically that our RGA method is computationally efficient and capable of solving inverse problems with $\mathcal{O}(10^7)$ observations using modest computational resources (approximately 10 US dollars if state-of-the-art cloud services are employed). Therefore, it shows great potential for characterizing subsurface heterogeneity for problems with a large number of observations.

The RGA method is coded in Julia and implemented in the MADS open-source high-performance computational framework (http://mads.lanl.gov). However, the implementation of RGA is relatively simple, and can be easily added to any existing code. Finally, the randomization method is not limited to hydrogeologic problems and applications. Being a successful data/dimensionality reduction technique, randomization can be applied to a broad set of applications in many science and engineering domains.

Appendix A: Definitions of Column Space and Subspace Embedding

A.1. Definition: Column Space

Consider a matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ $(n > d)$. Notice that as one ranges over all vectors $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{Ax}$ ranges over all linear combinations of the columns of $\mathbf{A}$ and therefore defines a d-dimensional subspace of $\mathbb{R}^n$, which we refer to as the column space of $\mathbf{A}$ and denote it by $C(\mathbf{A})$.

With the column space defined in definition A.1, the definition of subspace embedding can be provided as:

A.2. Definition: Subspace Embedding

A matrix $\mathbf{S} \in \mathbb{R}^{r \times n}$ provides a subspace embedding for $C(\mathbf{A})$ if $||\mathbf{SAx}||_2^2 = (1 \pm \epsilon)||\mathbf{Ax}||_2^2$, $\forall \mathbf{x} \in \mathbb{R}^d$, such a $\mathbf{S}$ provides a low distortion embedding, and is called subspace embedding.

Acknowledgments
Youzuo Lin, Daniel O'Malley, Ellen B. Le, and Velimir V. Vesselinov were supported by Los Alamos National Laboratory Environmental Programs Projects. In addition, Daniel O'Malley was supported by a Los Alamos National Laboratory (LANL) Director's Postdoctoral Fellowship, and Velimir V. Vesselinov was supported by the DiaMonD project (An Integrated Multifaceted Approach to Mathematics at the Interfaces of Data, Models, and Decisions, U.S. Department of Energy Office of Science, grant 11145687). We thank Jonghyun Lee, Wolfgang Nowak, and the Associate Editor (Sander Huisman) for their valuable comments that helped improve our manuscript. The code that generates all of our synthetic data is available at https://github.com/madsjulia/GeostatInversion.jl.

References

Ailon, N., and B. Chazelle (2010), Faster dimension reduction, *Commun. ACM*, *53*, 97–104.
Ambikasaran, S., J. Y. Li, P. K. Kitanidis, and E. Darve (2013), Large-scale stochastic linear inversion using hierarchical matrices, *Comput. Geosci.*, *17*(6), 913–927.
Avron, H., P. Maymounkov, and S. Toledo (2010), Blendenpik: Supercharging LAPACK's least-squares solver, *SIAM J. Sci. Comput.*, *32*(3), 1217–1236.
Bebendorf, M. (2008), *Hierarchical Matrices: A Means to Efficiently Solve Elliptic Boundary Value Problems*, vol. 63, Springer, New York.
Bezanson, J., A. Edelman, S. Karpinski, and V. B. Shah (2014), Julia: A fresh approach to numerical computing, Julia Computing Inc, Allston, Md. [Available at http://http://julialang.org/.]
Boian, S., and V. V. Vesselinov (2014), Blind source separation for groundwater pressure analysis based on nonnegative matrix factorization, *Water Resour. Res.*, *50*, 7332–7347, doi:10.1002/2013WR015037.
Borm, S., L. Grasedyck, and W. Hackbusch (2003), Introduction to hierarchical matrices with applications, *Eng. Anal. Boundary Elem.*, *5*(27), 405–422.
Carrera, J., and S. P. Neuman (1986), Estimation of aquifer parameters under transient and steady state conditions: 1. Maximum likelihood method incorporating prior information, *Water Resour. Res.*, *22*, 199–210.
Carrera, J., A. Alcolea, A. Medina, J. Hidalgo, and L. J. Slooten (2005), Inverse problem in hydrogeology, *Hydrogeol. J.*, *13*(1), 206–222.
Cirpka, O. A., and P. K. Kitanidis (2000), Sensitivity of temporal moments calculated by the adjoint-state method and joint inversing of head and tracer data, *Adv. Water Resour.*, *24*(1), 89–103, doi:10.1016/S0309-1708(00)00007-5.
Clarkson, K. L., and D. P. Woodruff (2009), Numerical linear algebra in the streaming model, in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, edited by M. Mitzenmacher, pp. 205–214, Assoc. for Comput. Mach., New York, N. Y.
Constantine, P. G., E. Dow, and Q. Wang (2014), Active subspace methods in theory and practice: Applications to kriging surfaces, *SIAM J. Sci. Comput.*, *36*(4), A1500–A1524.
Dasgupta, A., R. Kumar, and T. Sarlós (2010), A sparse Johnson-Lindenstrauss transform, in *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, pp. 341–350, Assoc. For Comput. Mach., New York, N. Y.
Drineas, P., and M. W. Mahoney (2016), RandNLA: Randomized numerical linear algebra, *Commun. ACM*, *6*(6), 80–90.
Drineas, P., M. W. Mahoney, S. Muthukrishnan, and T. Sarlos (2011), Faster least squares approximation, *Numer. Math.*, *117*, 219–249.
Engl, H. W., M. Hanke, and A. Neubauer (1996), *Regularization of Inverse Problems*, Kluwer Acad., Dordrecht, Netherlands.
Golub, G. H., and C. F. Van Loan (1996), *Matrix Computations*, 3rd ed., Johns Hopkins Univ. Press, Baltimore, Md.
Hansen, P. C. (1998), *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*, Soc. of Ind. and Appl. Math., Philadelphia, Pa.
Huang, L., J. Shin, T. Chen, Y. Lin, K. Gao, M. Intrator, and K. Hanson (2016), Breast ultrasound tomography with two parallel transducer arrays, in *Proceedings of the SPIE 9783, Medical Imaging 2016: Ultrasonic Imaging, Tomography, and Therapy*, edited by N. Duric, pp. 97,830C-1–97,830C-12, SPIE, Bellingham, Wash.
Hunt, R. J., J. Doherty, and M. J. Tonkin (2007), Are models too simple? Arguments for increased parameterization, *Ground Water*, *45*, 254–262.
Illman, W. A., S. J. Berg, and Z. Zhao (2015), Should hydraulic tomography data be interpreted using geostatistical inverse modeling? A laboratory sandbox investigation, *Water Resour. Res.*, *51*, 3219–3237, doi:10.1002/2014WR016552.
Iyer, C., C. Carothers, and P. Drineas (2016), Randomized sketching for large-scale sparse ridge regression problems, in *7th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems (ScalA)*, edited by V. Alexandrov, A. Geist, and J. Dongarra, pp. 65–72, IEEE, Washington, D. C.

Johnson, W. B., and J. Lindenstrauss (1984), Extensions of Lipschitz mappings into a Hilbert space, *Contemporary Math.*, *26*, 189–206.
Kane, D. M., and J. Nelson (2014), Sparser Johnson-Lindenstrauss transforms, *J. ACM*, *61*(1), 4.
Kitanidis, P. K. (1995), Quasi-linear geostatistical theory for inversing, *Water Resour. Res.*, *31*, 2411–2419.
Kitanidis, P. K. (1997a), *Introduction to Geostatistics: Applications to Hydrogeology*, Stanford-Cambridge Program, Cambridge Univ. Press, Cambridge, U. K.
Kitanidis, P. K. (1997b), The minimum structure solution to the inverse problem, *Water Resour. Res.*, *33*, 2263–2272.
Kitanidis, P. K., and J. Lee (2014), Principal component geostatistical approach for large-dimensional inverse problem, *Water Resour. Res.*, *50*, 5428–5443, doi:10.1002/2013WR014630.
Krebs, J. R., J. E. Anderson, D. Hinkley, R. Neelamani, S. Lee, A. Baumstein, and M. D. Lacasse (2009), Fast full-wavefield seismic inversion using encoded sources, *Geophysics*, *74*, WCC177–WCC188.
Le, E. B., A. Myers, T. Bui-Thanh, and Q. P. Nguyen (2015), A data-scalable randomized misfit approach for solving large-scale PDE-constrained inverse problems, *Inverse Probl.*, *33*(6), 065003–065009.
Lee, J., and P. K. Kitanidis (2014), Large-scale hydraulic tomography and joint inversion of head and tracer data using the principal component geostatistical approach (PCGA), *Water Resour. Res.*, *50*, 5410–5427, doi:10.1002/2014WR015483.
Lee, J., H. Yoon, P. K. Kitanidis, C. J. Werth, and A. J. Valocchi (2016), Scalable subsurface inverse modeling of huge data sets with an application to tracer concentration breakthrough data from magnetic resonance imaging, *Water Resour. Res.*, *52*, 5213–5231, doi:10.1002/2015WR018483.
Lin, Y., B. Wohlberg, and H. Guo (2010), UPRE method for total variation parameter selection, *Signal Process.*, *90*(8), 2546–2551, doi:10.1016/j.sigpro.2010.02.025.
Lin, Y., D. O'Malley, and V. V. Vesselinov (2016), A computationally efficient parallel Levenberg-Marquardt algorithm for highly parameterized inverse model analyses, *Water Resour. Res.*, *52*, 6948–6977, doi:10.1002/2016WR019028.
Liu, X., Q. Zhou, J. T. Birkholzer, and W. A. Illman (2013), Geostatistical reduced-order models in underdetermined inverse problems, *Water Resour. Res.*, *59*, 6587–6600, doi:10.1002/wrcr.20489.
Liu, X., Q. Zhou, P. K. Kitanidis, and J. T. Birkholzer (2014), Fast iterative implementation of large-scale nonlinear geostatistical inverse modeling, *Water Resour. Res.*, *50*, 198–207, doi:10.1002/2012WR013241.
Mahoney, M. W. (2011), Randomized algorithms for matrices and data, *Found. Trends Mach. Learn.*, *3*(2), 123–224.
Meng, M. A., X. Saunders, and M. W. Mahoney (2014), LSRN: A parallel iterative solver for strongly over- or underdetermined systems, *SIAM J. Sci. Comput.*, *36*, 95–118.
Moghaddam, P. P., H. Keers, F. J. Herrmann, and W. A. Mulder (2013), A new optimization approach for source-encoding full-waveform inversion, *Geophysics*, *78*(3), R125–R132.
Neuman, S. P., and S. Yakowitz (1979), A statistical approach to the inverse problem of aquifer hydrology: 1. Theory, *Water Resour. Res.*, *15*, 845–860, doi:10.1029/WR015i004p00845.
Neuman, S. P., G. E. Fogg, and E. A. Jacobson (1980), A statistical approach to the inverse problem of aquifer hydrology: 2. Case study, *Water Resour. Res.*, *16*, 33–58.
Nowak, W., and O. A. Cirpka (2004), A modified Levenberg-Marquardt algorithm for quasi-linear geostatistical inversing, *Adv. Water Resour.*, *27*, 737–750.
Nowak, W., and O. A. Cirpka (2006), Geostatistical inference of hydraulic conductivity and dispersivities from hydraulic heads and tracer data, *Water Resour. Res.*, *42*, W08416, doi:10.1029/2005WR004832.
Nowak, W., S. Tenkleve, and O. A. Cirpka (2003), Efficient computation of linearized cross-covariance and auto-covariance matrices of interdependent quantities, *Math. Geol.*, *35*(1), 53–66, doi:10.1023/A:1022365112368.
Paige, C. C., and M. A. Saunders (1975), Solution of sparse indefinite systems of linear equations, *SIAM J. Numer. Anal.*, *12*, 617–629.
Peitgen, H. O., and D. Saupe (1988), *The Science of Fractal Images*, Springer, New York.
Rokhlin, V., and M. Tygert (2008), A fast randomized algorithm for overdetermined linear least-squares regression, *Proc. Natl. Acad. Sci. U. S. A.*, *105*(36), 13,212–13,217.
Saad, Y., and M. H. Schultz (1986), GMRES: A generalized minimal residual method for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.*, *3*(7), 856–869.
Saibaba, A. K., and P. K. Kitanidis (2012), Efficient methods for large-scale linear inversion using a geostatistical approach, *Water Resour. Res.*, *48*, W05522, doi:10.1029/2011WR011778.
Sarlos, T. (2006), Improved approximation algorithms for large matrices via random projections, in *FOCS'06. 47th Annual IEEE Symposium on Foundations of Computer Science*, edited by A. Z. Broder, pp. 143–152, IEEE, Washington, D. C.
Schoniger, A., W. Nowak, and H.-J. Hendricks Franssen (2012), Parameter estimation by ensemble Kalman filters with transformed data: Approach and application to hydraulic tomography, *Water Resour. Res.*, *48*, W04502, doi:10.1029/2011WR010462.
Sun, N. (1994), *Inverse Problems in Groundwater Modeling*, Kluwer Acad., Dordrecht, Netherlands.
Tarantola, A. (2005), *Inverse Problem Theory*, Soc. of Ind. and Appl. Math., Philadelphia, Penn.
Tonkin, M. J., and J. Doherty (2005), A hybrid regularized inversion methodology for highly parameterized environmental models, *Water Resour. Res.*, *41*, W10412, doi:10.1029/2005WR003995.
Vesselinov, V., S. Neuman, and W. Illman (2001a), Three-dimensional numerical inversion of pneumatic cross-hole tests in unsaturated fractured tuff 1. Methodology and borehole effects, *Water Resour. Res.*, *37*, 3001–3017, doi:10.1029/2000WR000133.
Vesselinov, V., S. Neuman, and W. Illman (2001b), Three-dimensional numerical inversion of pneumatic cross-hole tests in unsaturated fractured tuff 2. Equivalent parameters, high-resolution stochastic imaging and scale effects, *Water Resour. Res.*, *37*, 3019–3041, doi:10.1029/2000WR000135.
Vesselinov, V. V., D. O'Malley, Y. Lin, S. Hansen, and B. Alexandrov (2015), MADS.jl: (Model Analyses and Decision Support) in Julia, Los Alamos National Laboratory, Los Alamos, N. M. [Available at http://mads.lanl.gov/.]
Vogel, C. (2002), Computational Methods for Inverse Problems, Soc. of Ind. and Appl. Math., Philadelphia, Penn.
Wang, K., T. Matthews, F. Anis, C. Li, N. Duric, and M. A. Anastasio (2015), Waveform inversion with source encoding for breast sound speed reconstruction in ultrasound computed tomography, *IEEE Trans. Ultrason. Ferroelectr. Freq. Control*, *62*(3), 475–493, doi:10.1109/TUFFC.2014.006788.
Woodruff, D. P. (2014), Sketching as a tool for numerical linear algebra, arXiv preprint arXiv:1411.4357.
Yang, J., X. Meng, and M. Mahoney (2016), Implementing randomized matrix algorithms in parallel and distributed environments, *Proc. IEEE*, *104*, 58–92.
Yeh, T.-C. J., and J. Simunek (2002), Stochastic fusion of information for characterizing and monitoring the vadose zone, *Vadose Zone*, *1*, 2095–2105.

Yin, D., and W. A. Illman (2009), Hydraulic tomography using temporal moments of drawdown recovery data: A laboratory sandbox study, *Water Resour. Res.*, *45*, W01502, doi:10.1029/2007WR006623.

Zhang, J., and T.-C. J. Yeh (1997), An iterative geostatistical inverse method for steady flow in the vadose zone, *Water Resour. Res.*, *33*, 63–71.

Zhang, Z., L. Huang, and Y. Lin (2012), Efficient implementation of ultrasound waveform tomography using source encoding, in *Proc. SPIE 8320, Medical Imaging 2012: Ultrasonic Imaging, Tomography, and Therapy*, edited by J. Bosch and M. Doyley, pp. 832,003–1–832,003–10 SPIE, Bellingham, Wash.

Zhu, J., and J. Yeh (2006), Analysis of hydraulic tomography using temporal moments of drawdown recovery data, *Water Resour. Res.*, *42*, W02403, doi:10.1029/2005WR004309.