Relaxation algorithms for matrix completion, with applications to seismic travel-time data interpolation.

Robert Baraldi[1], Carl Ulberg[2], Rajiv Kumar[3], Kenneth Creager[2] and Aleksandr Aravkin[1]

[1]Department of Applied Mathematics, University of Washington
[2]Department of Earth and Space Sciences, University of Washington
[3]School of Earth and Atmospheric Sciences, Georgia Institute of Technology

Abstract. Travel time tomography is used to infer the underlying three-dimensional wavespeed structure of the Earth by fitting seismic travel time data collected at surface stations. Data interpolation and denoising techniques are important pre-processing steps that use prior knowledge about the data, including parsimony in the frequency and wavelet domains, low-rank structure of matricizations, and local smoothness.

We show how local smoothness structure can be combined with low rank constraints using level-set optimization formulations, and develop a new relaxation algorithm that can efficiently solve these joint problems. In the seismology setting, we use the approach to interpolate missing stations and de-noise observed stations. The new approach is competitive with alternative algorithms, and offers new functionality to interpolate observed data using both smoothness and low rank structure in the presence of data fitting constraints.

Submitted to: *Inverse Problems*

1. Introduction

Travel-time tomography is used to determine the underlying structure of the earth and how seismic waves propagate through that structure. This weakly nonlinear problem is formulated as a data-fitting inverse problem and solved using iterative optimization techniques. Data quality and availability are key constraints and can drastically influence the merit of the results [21, 15, 16]. Hence, researchers often use prior information to denoise and interpolate the data prior to inversion. Parsimonious representations [6, 22] of the data in transform domains such as Fourier [23] and Curvelet [9] have been used in exploration seismology [8, 13], along with low-rank representations [1, 7].

We focus on regional seismology — in particular, we want to analyze data obtained by the Imaging Magma Under St. Helens (iMUSH) project, a multi-year effort to image and infer the architecture of the greater Mount St. Helens, WA, magmatic system

[27, 12]. The iMUSH project uses a variety of geophysical and petrological methods, including active source, local earthquake, and ambient noise seismic tomography to study the most active volcano in the Cascades arc. For the passive source seismic portion of the iMUSH project, 70 broadband seismometers were deployed from 2014 to 2016 within a 100km diameter circle around the mountain; these have an average station spacing of 10km, and are supplemented by permanent stations maintained by the Pacific Northwest Seismic Network (PNSN) and a temporary array of 20 broadband seismometers deployed by AltaRock Energy in June to November of 2016.

The data collected from this experiment comprises P-wave travel times from 23 active borehole explosions [11] and over 400 local earthquakes [27] recorded at the iMUSH broadband array. Collected data has a range of fidelities, and different subsets inform different parameters for 3D P-wave velocities and hence geometries of the structure underlying Mount St. Helens. Travel times are inverted to obtain 3D seismic velocity models and image complex subsurface structures, including low velocity zones. The efficacy of the approach depends on the quality of the data, which is affected by noise from roads, streams, ocean waves, and wind. The signal to noise ratio (SNR) of the data depends on the source size, distance from the seismometer, and attenuation structure of the earth. Data is scarce because many landscape features are impassable; typical station spacing is 10km. Arrival times are chosen by operators, who assign uncertainties for particular observations based on confidence in seismic readings. Low-magnitude events in particular are often difficult to distinguish from noise.

Our goal here is to increase the raypath coverage for use in earthquake tomography by simulating seismometer locations and observations by interpolating noisy data. This is particularly useful for areas within the study region that did not have seismometers installed or which had seismometers out of service for extended periods of time. Since we know the data is corrupted by uncertainty and noise, the problem is a good match for level-set optimization formulations [2] that minimize a regularizer subject to a prescribed level of data fit. We propose a relaxation formulation that allows misfit constraints while (1) penalizing rank of a tessellation that captures redundancy of features across sources, and (2) enforcing smooth features consistent with the underlying physical model. We develop an efficient block-coordinate algorithm for this formulation, and compare the approach with a variety of competing formulations and algorithms.

The paper proceeds as follows. In Section 2 we review relevant formulations and algorithms for interpolation and denoising. In Section 3, we develop the extended model formulation and the relaxation method to solve it. Section 4 describes the data used and how it fits into a low-rank interpolation scheme. In Section 5 we evaluate the approach and compare it against alternatives for denoising and interpolating Mount St. Helens data.

2. Preliminaries

In this section, we set up the notation, review prior information used in interpolation and denoising, and discuss different types of standard optimization formulations needed to implement such approaches.

2.1. Notation

We use the following notation conventions in the paper. Lowercase variables (x) denote vectors, while uppercase variables (X) represent matrices. Calligraphic uppercase letters ($\mathcal{A}$) are used for operators or functionals. The terms (R_x, R_y) represent a receiver coordinate grid. The variables $n_d, n_{R_x}, n_{R_y}, n_s$ represent the number of observed data points, the number of points in the x-direction of the receiver grid, the number of points in the y-direction of the receiver grid, and the number of sources. The variable Ω is used to represent the entire 4-dimensional source/receiver space and generically the interpolation space.

2.2. Formulations for Low-rank Interpolation

The goal of low-rank matrix completion is to accurately estimate the missing data entries of a matrix $X \in \mathbb{R}^{m\times n}$ from observed entries. Low-rank structure is often inferred if completed X has few non-zero singular values and experiences entry repetition. The observed entries are given by the vector $b \in \mathbb{R}^{n_d}$. We let $\mathcal{A} : \mathbb{R}^{n\times m} \to \mathbb{R}^{n_d}$ be the restriction/interpolation operator that maps elements from the regular grid X to the observed values b, which can be written as $b = \mathcal{A}(X) + \epsilon$ for $\epsilon \in \mathbb{R}^{n_d}$. Here, ϵ represents the data corruption of observed entries via some noise distribution. In formulating the problem, the nuclear norm

$$\|X\|_* = \sum_{j=1}^{\min(n,m)} \hat{\sigma}_j(X),$$

with $\hat{\sigma}_j$ the singular values of X, is used as a proxy for rank. The classic formulations that balance data fit with regularization are

$$\min_{X\in\mathbb{R}^{n\times m}} \|X\|_* + \frac{1}{\sigma}\|\mathcal{A}(X) - b\|_2 \tag{1}$$

$$\min_{X\in\mathbb{R}^{n\times m}} \quad \|\mathcal{A}(X) - b\|_2 \quad \text{s.t.} \quad \|X\|_* \le \tau \tag{2}$$

$$\boxed{\min_{X\in\mathbb{R}^{n\times m}} \|X\|_* \quad \text{s.t.} \quad \|\mathcal{A}(X) - b\|_2 \le \sigma}. \tag{3}$$

These formulations are known as Tichonoff, Ivanov, and Morozov regularization [20], respectively. The misfit-constraint Morozov variant (3) is best suited for situations where a good estimate of the uncertainty, σ, is available. To simplify exposition, we will focus on Morozov-type formulations.

2.3. Smoothness constraints

In travel time tomography, smoothness and continuity between gridpoints is a reasonable prior, since the geological structure of the crust exhibits the traits of approximately homogeneous media. Smoothness is enforced by introducing a penalty term $\|\mathcal{L}(X)\|_2^2$, with $\mathcal{L}$ the discretization of the Laplacian operator. In 1D, it is a tridiagonal difference matrix operating on the vectorized X; in 2D, it has four off-diagonal elements.

Enforcing smoothness and low-rank structure combines local and global information. A Morozov formulation, with γ balancing the regularizers, is given by

$$\min_X \|X\|_* + \frac{1}{2\gamma}\|\mathcal{L}(X)\|_2^2 \quad \text{s.t.} \quad \|\mathcal{A}(X) - b\|_2 \leq \sigma. \tag{4}$$

2.4. Factorized Formulations

Theoretical properties of matrix completion via nuclear-norm minimization have been extensively studied [4, 5]. The theoretical appeal of convex formulations is tempered by computational considerations — algorithms that optimize $\|X\|_*$ require a full matrix decision variable, and full or partial singular value decompositions (SVDs) at each iteration. An efficient alternative is to use matrix factorization formulations [1], writing $X = LR^T$, with $L \in \mathbb{R}^{n\times k}, R \in \mathbb{R}^{m\times k}$. From [22], we have the characterization

$$\|X\|_* = \inf_{L,R:X=LR^T} \frac{1}{2}(\|L\|_F^2 + \|R\|_F^2),$$

which allows us to replace $\|X\|_*$ in any formulation by $\frac{1}{2}(\|L\|_F^2 + \|R\|_F^2)$ for any factorization $X = LR^T$. For example, the three formulations (1)-(3) become

$$\min_{L\in\mathbb{R}^{n\times k},\ R\in\mathbb{R}^{m\times k}} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 + \frac{1}{\sigma}\|\mathcal{A}(LR^T) - b\|_2 \tag{5}$$

$$\min_{L\in\mathbb{R}^{n\times k},\ R\in\mathbb{R}^{m\times k}} \|\mathcal{A}(LR^T) - b\|^2 \quad \text{s.t.} \quad \|L\|_F^2 + \|R\|_F^2 \leq 2\tau \tag{6}$$

$$\boxed{\min_{L\in\mathbb{R}^{n\times k},\ R\in\mathbb{R}^{m\times k}} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 \quad \text{s.t.} \quad \|\mathcal{A}(LR^T) - b\|_2 \leq \sigma}, \tag{7}$$

where $k \ll \min(n, m)$, and the memory requirements are reduced from mn to $k(n+m)$. No SVDs are required; formulation (5) is smooth, formulation (6) requires simple projections onto the Frobenius-norm ball, and formulation (7) can be solved using (6) via root-finding as described by [1].

Our primary technical goal here is to solve the factorized Morozov formulation corresponding to (4):

$$\min_{L,R} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 + \frac{1}{2\gamma}\|\mathcal{L}(LR^T)\|_2^2 \quad \text{s.t.} \quad \|\mathcal{A}(LR^T) - b\|_2 \leq \sigma. \tag{8}$$

In particular, this formulation incorporates both local and global structure, and gives a misfit target σ. It requires a new algorithm, since (8) cannot be solved by the level-set approach of [1]. The only available alternative is to use Tichonoff-type formulations (see Section 3.1). Our main technical contribution is a nonconvex splitting algorithm for problem (8), developed in the next section.

3. Relaxed Joint Inversion

The main challenge of the factorized Morozov formulation (8) is the data-misfit constraint. To solve the problem, we propose a *relaxation* following the ideas of [29]. In particular, we introduce an auxiliary variable $W \approx LR^T$:

$$\min_{L,R,W} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 + \frac{1}{2\gamma}\|\mathcal{L}(W)\|_2^2 + \frac{1}{2\eta}\|W - LR^T\|_F^2 \quad \text{s.t.} \quad \|\mathcal{A}(W) - b\|_2 \leq \sigma. \tag{9}$$

Problem (9) is a relaxation for problem (8), since W approximates $X = LR^T$; in particular $\|W - LR^T\| = \mathcal{O}(\eta)$. The salient modeling features of (8) are still preserved. We can now design a simple block-coordinate descent algorithm by iteratively optimizing in each of (L, R, W), detailed in Algorithm 1.

Algorithm 1 Block-Coordinate Descent for (9).

1: **Input:** W_0, L_0, R_0
2: Initialize: $i = 0$.
3: **while** not converged **do**
4: $\quad L_{i+1} \leftarrow \left(I + \eta R_i^T R_i\right)^{-1} (\eta R_i^T W_i^T)$ ▷ Solves $\frac{1}{2}\|L\|_F^2 + \frac{1}{2\eta}\|W_i - LR_i^T\|_F^2$
5: $\quad R_{i+1} \leftarrow (\eta W_i^T L_{i+1}) \left(I + \eta L_{i+1}^T L_{i+1}\right)^{-1}$ ▷ Solves $\frac{1}{2}\|R\|_F^2 + \frac{1}{2\eta}\|W_i - L_{i+1}R^T\|_F^2$
6: $\quad W_{i+1} \leftarrow \arg\min_W \frac{1}{2\gamma}\|\mathcal{L}(W)\|_2^2 + \frac{1}{2\eta}\|W - L_{i+1}R_{i+1}^T\|_F^2 \quad \text{s.t.} \quad \|\mathcal{A}(W) - b\|_2 \leq \sigma$
7: $\quad i \leftarrow i + 1$
8: **Output:** W_i, L_i, R_i

Steps 4 and 5 of Algorithm 1 are simple least squares updates; each minimizes (9) in L and R respectively, with the remaining variables held fixed. Algorithm 1 converges to a stationary point of (9) by [26, Theorem 4.1]. In particular, $f(L, R, W)$ has a unique minimum in each coordinate block with the remaining blocks held fixed, which satisfies condition (c) of the theorem. The uniqueness of the minima are clear from the closed form solutions in steps 4 and 5, and from the strong convexity of the W subproblem in step 6. This step 6 is solved using an efficient root-finding method. First, we describe the equivalent penalized problem with penalty parameter λ, given by

$$w(\lambda) := \arg\min_w \frac{1}{2\gamma}\|L_\nabla w\|_2^2 + \frac{1}{2\eta}\|w - d_{i+1}\|_F^2 + \frac{\lambda}{2}\|Aw - b\|^2, \tag{10}$$

where $w = \mathrm{vec}(W)$, L_∇ is a sparse matrix that encodes the action of the Laplacian on w, A is a linear operator that sends w to b, and $d_{i+1} = \mathrm{vec}(L_{i+1}R_{i+1}^T)$. The root finding method obtains the smallest value of λ satisfying $\|\mathcal{A}w(\lambda) - b\|_2 \leq \sigma$.

Taking the gradient of the objective defining $w(\lambda)$ in (10) and setting it equal to 0, we find an explicit formula

$$w(\lambda) = \left(\frac{1}{\gamma}(L_\nabla^T L_\nabla) + \frac{1}{\eta}I + \lambda A^T A\right)^{-1} \left(\frac{1}{\eta}d_{i+1} + \lambda A^T b\right). \tag{11}$$

We need only find the smallest $\lambda \geq 0$ so that $\|Aw(\lambda) - b\|_2 = \sigma$. The special case $\lambda = 0$ occurs when the constraint is satisfied at the least squares solution $w(0)$ in (11):

$$\left\| A \left(\frac{1}{\gamma}(L_\nabla^T L_\nabla) + \frac{1}{\eta} I \right)^{-1} \left(\frac{1}{\eta} d_{i+1} \right) - b \right\| \leq \sigma,$$

In all other cases, we have

$$f(\lambda) := \sigma - \|Aw(\lambda) - b\|_2, \quad f'(\lambda) = -\frac{\langle A^T A w(\lambda) - b, \nabla_\lambda w(\lambda) \rangle.}{\|Aw(\lambda) - b\|_2}$$

We compute the quantity $\nabla_\lambda w(\lambda)$ required to evaluate $f'(\lambda)$ using the complex step method [18], instead of differentiating (11) directly. The root-finding update is given by

$$\lambda^+ := \lambda - \frac{f(\lambda)}{f'(\lambda).}$$

The expensive step (11) is implemented using Cholesky factors of the sparse matrix $\left(\frac{1}{\gamma}(L_\nabla^T L_\nabla) + \frac{1}{\eta} I + \lambda A^T A\right)$. This system only changes when η is updated. Updating W in Algorithm 1 has a relatively high computational cost. Table 1 gives a detailed arithmetic complexity analysis, with $L \in \mathbb{R}^{n \times k}$, $R \in \mathbb{R}^{m \times k}$, $W \in \mathbb{R}^{n \times m}$. For our data, $n = m$ and W is square.

Table 1. Numerical complexity for Algorithm 1. Key steps are Cholesky factorization (CF), back-substitution (BS), and root finding.

Line	$\mathcal{O}(\cdot)$	Explanation
Line 4	$\frac{k^3}{3} + 2k^2 m$	CF of $k \times k$ matrix with BS of $k \times m$ matrix
Line 5	$\frac{k^3}{3} + 2k^2 n$	CF of $k \times k$ matrix with BS of $k \times n$ matrix
Line 6 $\sigma = 0$	$\frac{(nm)^3}{3} + 2(nm)^2$	CF of $nm \times nm$ matrix and BS
Line 6 $\sigma > 0$	$(k_r(2k_p) + 1 + k_l)\,(nm)^2$	Rootfinding algorithm with $k_l, k_p, k_r \leq 100$.

Line 6 of Algorithm 1 requires a root-finding algorithm or another Cholesky decomposition, depending on the value selected for σ. If $\sigma = 0$, then we have the Cholesky decomposition of

$$\left(\frac{1}{\gamma}(L_\nabla^T L_\nabla) + \frac{1}{\eta} I + \lambda A^T A \right)$$

from Equation 11 which is order $\frac{(nm)^3}{3} + 2(nm)^2$; the update is required when η or λ change. If $\sigma > 0$, then we implement the root finding algorithm with inputs $L_\nabla^T L_\nabla \in \mathbb{R}^{nm \times nm}$, $L_\nabla A^T b = q \in \mathbb{R}^{nm}$, and $A \in \mathbb{R}^{n_d \times nm}$. This requires three steps. The first is matrix-vector multiply $L_\nabla^T q$. Next, we apply LSQR iterations of order $\mathcal{O}\,(nm)^2$ where $k_l \leq 100$ is the number of LSQR iterations. Finally, we use k_r iterations of PCG for $k_p \leq 100$ iterations. Here, the major cost is $\mathcal{O}\,(nm)^2$. Hence, in the total root finding algorithm with k_r iterations, we have one mat-vec, k_l LSQR, and k_p PCG iterations, with $k_l, k_p, k_r \leq 100$.

3.1. Alternative Approaches for Low-Rank & Smooth Inversion

There are alternative ways to model problem (9). We consider two formulations and algorithms to compare with Algorithm (1).

Nuclear-norm formulation using FISTA. A simple convex formulation that uses smoothness, data misfit, and a rank proxy (nuclear norm) is given by

$$\min_X \frac{\lambda}{2}\|\mathcal{A}(X) - b\|^2 + \frac{1}{2\gamma}\|\mathcal{L}(X)\|^2 + \|X\|_* \tag{12}$$

Formulation (12) is the sum of a smooth and a simple function, and can be solved using projected gradient or Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [3], detailed in Algorithm 2. This class of algorithms can be viewed as an extension of the classical gradient algorithm and is attractive due to its simplicity. The step size α is the reciprocal of the largest singular value of $(\lambda\mathcal{A}^*\mathcal{A} + \gamma^{-1}\mathcal{L}^*\mathcal{L})$, and the operator S_α is the soft-thresholding operator:

$$S_\alpha(\Sigma)_{jj} = \max(0, \Sigma_{jj} - \alpha).$$

Algorithm 2 FISTA for (12).

1: **Input:** $X^0 = X^{-1} \in \mathbb{R}^{m\times n},\ t^0 = t^{-1} = 1$
2: Initialize: $i = 0$
3: **while** not converged **do**
4: $\quad Y_i \leftarrow X_i + \frac{t_{i-1}-1}{t_i}(X_i - X_{i-1})$
5: $\quad G_i \leftarrow Y_i - \alpha\left((\lambda\mathcal{A}^*\mathcal{A} + \gamma^{-1}\mathcal{L}^*\mathcal{L})\mathrm{vec}(Y_i) - \lambda\mathcal{A}^*b\right)$
6: $\quad U, \Sigma, V^T \leftarrow \texttt{svd}(G_i)$
7: $\quad X_{i+1} \leftarrow US_\alpha(\Sigma)V^T$
8: $\quad t_{i+1} \leftarrow \frac{1+\sqrt{1+4(t^i)^2}}{2}$
9: $\quad i \leftarrow i + 1$
10: **Output:** X_i

Algorithm 2 uses gradients of the smooth terms, which requires appying $\mathcal{A}$, $\mathcal{L}$ and their adjoints, and the prox operator of $\|\cdot\|_*$, which requires thresholding on singular values computed via SVD (steps 6,7). The most expensive computational step here is the `svd`, which requires $\mathcal{O}(2mn^2 + 2n^3)$ arithmetic operations [25]. These steps become prohibitively expensive as the dimensions of X grow.

Smooth Factorized Formulation with L-BFGS. To avoid the SVD steps of Algorithm 2, we use the factorization strategy described in Section 2.4:

$$\min_{L,R} \frac{\lambda}{2}\|\mathcal{A}(LR^T) - b\|^2 + \frac{1}{2\gamma}\|\mathcal{L}(LR^T)\|^2 + \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2. \tag{13}$$

Formulation (13) is smooth with respect to the decision variables L and R, and at larger scales, the limited memory BFGS (L-BFGS) [19] algorithm is a reasonable choice. The number of operations per iteration is $\mathcal{O}(k_L nm)$, with k_L the L-BFGS history size.

Convergence Complexity An apples to apples complexity analysis is difficult for the algorithms presented. We have no complexity result for Algorithm 1; only a convergence guarantee (eventual stationarity of iterates). The accelerated proximal gradient method used for matrix completion in [24] has a rate of convergence $\mathcal{O}\left(1/\sqrt{\varepsilon}\right)$ or $\mathcal{O}\left(\sqrt{\frac{L_f}{k}}\right)$, where L_f is the Lipschitz constant of the least squares component of formulation 12. However, this paper addresses the *exact* problem, while FISTA and L-BFGS are solving relaxations. In the context of our application, Algorithm 1 converges in an order of magnitude fewer iterations to a higher root-mean-squared error (for both observed and unobserved). For example, this is 2000 iterations of FISTA relative to our 200 iterations (in Algorithm 1, both $\sigma > 0$ and $\sigma = 0$), see Section 5.

In the next section, we describe the travel-time interpolation problem for regional seismology, evaluate low rank and smooth regularization, and compare the performance of Algorithm 1 for (9) to those of FISTA (Algorithm 2) on (12) and L-BFGS on (13), in terms of computational efficiency and quality of reconstruction.

4. Interpolation of synthetic travel time iMUSH data

This experiment interpolates *synthetic* travel time residuals, which are calculated with respect to travel times predicted by a 1D velocity model. Travel times vary on the order of tens of seconds, while travel time residuals are generally between -1 and 1 seconds. We obtain synthetic travel times by using the forward modeling portion of the `struct3DP` code, which uses a finite difference 3D eikonal equation solver [28, 10]. These travel times are calculated for the best-fit 3D model from the iMUSH local earthquake tomography and compared to travel times through the PNSN S4 [17] 1D velocity model to obtain the synthetic residual. Similarly, for observed travel times, we subtract the travel time predicted through the 1D model from the observed travel time to obtain the observed residual.

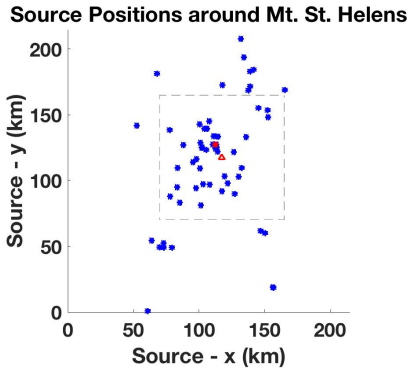
Here, we define the experimental data. While raw iMUSH/PNSN data exists at station locations around the mountain (see blue dots in Figure 1(b)), these iMUSH/PNSN stations are not gridded. To deal with this problem, we include an interpolation operator into the standard linear map $\mathcal{A}$, as discussed in detail in Section 6. In the following synthetic experiments, we solve a forward problem with the 3D eikonal equation solver and use the best-fit 3D model to generate synthetic results for uniformly gridded stations in a square of 70km to 165km (with origin being at latitute 45.2, longitude -123.7). For the rest of this paper, we refer to the travel-time residuals between these synthetic travel times and those predicted by the 1D PNSN S4 model

as the *true* data, or X_{true}. To generate observations for the interpolation schemes, we subsample this data down to 15% of total grid coverage over all sources by picking synthetic gridded stations near raw iMUSH/PNSN stations. This is a subset of synthetic stations meant to represent the distribution of the real-world stations that recorded the event. This subsampled data is then corrupted with noise. For each station, we generate a standard deviation parameter from the uniform distribution (0.03-0.15)(s); this captures each synthetic station's inherent uncertainty. The deviation range is based on raw iMUSH uncertainties recorded over the two year period. Then, for each station's data, we add zero mean random gaussian noise generated using that station's deviation parameter to create the *observed* data, or X_{obs}. Unobserved entries of X_{obs} are zero, while observed entries are given by $\mathcal{A}(X_{obs}) = b$. The operator $\mathcal{A}(\cdot) : \mathbb{R}^{n\times m} \to \mathbb{R}^{n_d}$ takes the tensor we wish to interpolate, X, and selects the observed entries to return a vector the same size as $b \in \mathbb{R}^{n_d}$. This is represented by a binary matrix $A \in \mathbb{R}^{n_d \times nm}$ that selects the observed entries out of a vectorized X.

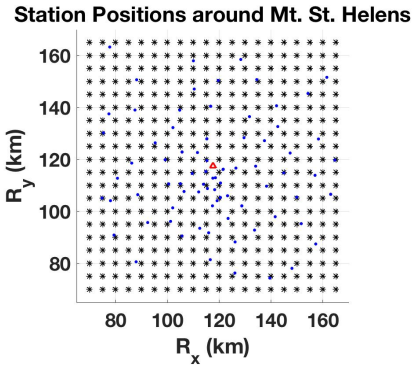
4.1. Description of source tensor construction

In order to apply the methods of Section 3, we have to specify a matricization of the data. The matricization we use is derived from the receiver grid, which is represented as a 95km-wide mesh with 5 kilometer spacing centered near Mount St. Helens. Again, the station grid is centered near the mountain and roughly coincides with the 70 stations deployed for 2 years. Each entry in the matrix represents a point on this uniform receiver grid. Missing entries are designated by zeros, while observed entries are represented with the travel time residuals relative to the 1D model. Our experiments in this paper focus mainly on synthetic residuals of the nonlinear 3D modeled data relative to the 1D model, which provide a 'ground truth' dataset we use to evaluate and compare interpolation techniques.

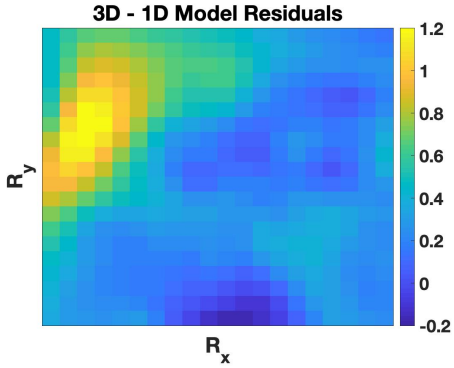
Each source represents a wave moving through the same media. The underlying physical model suggests enforcing smoothness between station gridpoints. We further improve interpolation with low-rank methods by finding a tessellation of source-receiver grid matrices in which full data exhibits fast decay of singular values, while subsampled data does not. Intuitively, such a tessellation reflects the redundancy of features across sources. Our combined goal is to interpolate X from a subset of observations by penalizing rank across sources, and nonsmooth local features. We parameterize the time picks into 4-tensors Ω_{ijkl} where (i, j) are the receiver index pairs and k, l are the source index pairs. The observed residual data is recorded in this 4D tensor format with dimension $(i, j, k, l) \in (1 \dots n_{R_x}, 1 \dots n_{R_y}, 1 \dots \sqrt{n_s}, 1 \dots \sqrt{n_s})$, where $n_{R_x} = 20, n_{R_y} = 20, n_s = 64$ for the experiments. Each source (k, l) has an associated receiver grid of observations $(R_x, R_y) \in (70, 165\text{km}) \times (70, 165\text{km})$ wherein each receiver coordinate pair (i, j) lies. Each receiver axis lies between 70km and 165km with 5km spacing. The observation grid was chosen to lie close to the mountain and to contain a relatively large number of sensors (see Figure 1(b)). Initially, we observed 64 sources, where each



(a) Spatial locations of the sources ('*') around Mount St. Helens (Δ). The box represents the edges of the uniform 20×20 station grid. The grid panel (c) is taken from the source marked with a red '*'.



(b) Receiver grid for a single source with the most datapoints. Black '*' represent synthetic stations on a grid, while blue '.' represent iMUSH/PNSN stations off-grid.

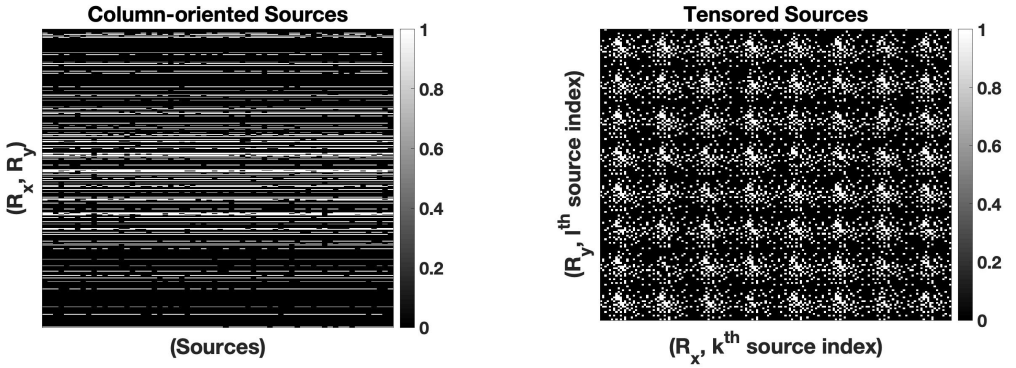


(c) *True* residual data (in seconds) for the receiver grid in panel (b).

Figure 1. Data information for synthetic test.

source recorded at most 80 receivers (out of the potential 400). Source locations for the 64 sources are shown in Figure 1(a), and a sample receiver grid for a particular source is given in Figure 1(b) and Figure 1(c) is the *true* data for that grid. Since observed residual data is a tensor, we have to choose a matricization to exploit the induced low-rank structure.

We consider two such matricizations: 1) $X_{(n_{R_x}-1)i+j,(\sqrt{n_s}-1)k+l} = \Omega_{ijkl}$, where we group receivers along columns and sources along rows, and 2) $X_{i+(n_{R_x}-1)k,j+(n_{R_y}-1)l} = \Omega_{ijkl}$, where each receiver grid is block-inserted into the underlying matrix. On a 20x20 receiver grid with 64 sources, the first matricization $X \in \mathbb{R}^{400\times 64}$, depicted in Figure 2(a), is obtained by letting $\Omega_{ij11} \in \mathbb{R}^{R_x,R_y} = \mathbb{R}^{400\times 1}$ be the vectorized receiver grid for single source $k = l = 1$; the sources are then arranged column-wise for $k, l = 1, \ldots, \sqrt{n_s}$. The second matricization $X \in \mathbb{R}^{160\times 160}$, depicted in Figure 2(b), is obtained by letting $\Omega_{ij11} \in \mathbb{R}^{R_x\times R_y} = \mathbb{R}^{20\times 20}$ be the nested receiver grid for single source $k = l = 1$. The sources are then nested together for $k, l = 1, \ldots, \sqrt{n_s}$.



(a) Subsampled matricization $X_{(n_{R_x}-1)i+j,(\sqrt{n_s}-1)k+l}$; the missing data (zeros) go across the rows.

(b) Subsampled matricization $X_{i+(n_{R_x}-1)k,j+(n_{R_y}-1)l}$; missing entries (zeros) are are interwoven throughout the matrix.

Figure 2. Different tensor formulations for low-rank interpolation.

The subsampling scheme is crucial for the approach. The ideal situation is for the subsampled data to have high rank (slow decay of singular values), while the full data has low rank (fast decay of singular values). Then, it is possible to recover the full volume by penalizing rank while matching observed data. **R2.10** For the low-rank penalty to be effective, the subsampled data must have high rank; otherwise, the low-rank penalty will have no effect on the problem. That is, the matricized X_{obs} (plotted in Figure 2(b)) should have a higher rank than the fully sampled matricized volume we want to recover. The difference between tensor respresentations in Figure 2 is that the columns of $X_{(n_{R_x}-1)i+j,(\sqrt{n_s}-1)k+l}$ (Figure 2(a)) have high mutual coherence, while the columns of $X_{i+(n_{R_x}-1)k,j+(n_{R_y}-1)l}$ (Figure 2(b)) have low mutual coherence. The coherence or mutual coherence of a matrix is defined as the maximum absolute value of the cross-correlations between the columns of that matrix. According to Theorem 1.3 in [5], low coherence implies that few entries of X are required for recovery of the full matrix. Hence, we require that X_{obs} have high rank, and the tensor representation plays an important role in the success of the algorthim. The first matricization does not satisfy this simple requirement: the subsampled matrix is itself low-rank and has rows and columns made up of zeros. Therefore we use the second matricization with tessellated sources, which indeed satisfies the requirement, see Figure 4.1.

In both matricizations, the sources are organized from largest in magnitude to smallest in magnitude. In Formulation 1, the sources are arranged such that the source with the greatest absolute residual values is in the first column and the least absolute residuals is in the last. In Formulation 2, the source with the greatest energy is in the top left, and the least in the bottom right; the energy then decays by source column (i.e. the next highest energy is the source immediately below in the row of sources). This was done to promote an even slower decrease in singular values for the observed matrix, and makes the low-rank approach far more effective.

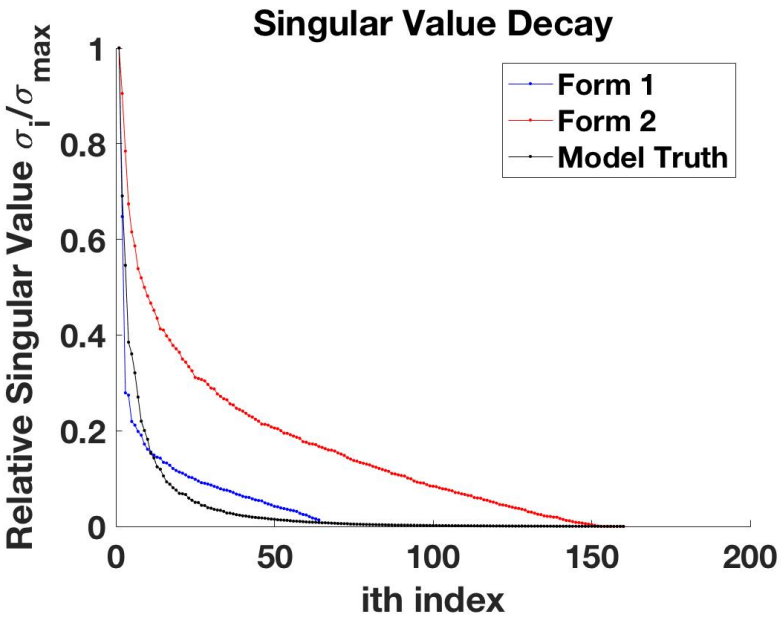


Figure 3. Singular value decay for matricization formulations 1 and 2 of the interpolation tensor.

5. Application to Synthetic Data

First, we evaluate the accuracy of using both smoothness and low-rank, compared to using either property alone. Then, we compare the speed and robustness of the new algorithm with competitors: FISTA (Algorithm 2, Equation 12) and L-BFGS (Equation 13). All tests use a tensored grid with the number of sources $n_s = 64$, and grid sizes $n_{R_x}, n_{R_y} = 20$ for $R_x, R_y \in (70, 165)$ evenly spaced at 5km. For all plots, north is up. The subsampling rate for every test is approximatestely 15%. We set the value k (for the LR^T formulation) to be 40 for all algorithms. The value of k was chosen to be significantly smaller than the total number of sources; it is 1/4th of the total number of columns in our experiment. A systemic method for chosing k is currently a driving question within the community; in practice, one can choose a larger k, if there is sufficient computational budget, since penalizing $\|L\|_F^2 + \|R\|_F^2$ controls the nuclear norm (and hence rank) through the relationship

$$\|LR\|_* \leq \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2.$$

These ideas are discussed at length in [1], where numerical tests confirm that driving k to be higher does not result in overfitting the data.

First, we compare the 'low-rank only' formulation (4) with the 'smoothing only' formulation

$$\min_W \|\mathcal{L}(W)\|_2^2 \quad \text{s.t.} \quad \|\mathcal{A}(W) - b\|_2 \leq \sigma. \tag{14}$$

For this test, we use the inequality constraint $\|\mathcal{A}(W) - b\|_2 \leq \sigma$; the equality constraint ($\mathcal{A}(W) = b$) is infeasible. This requires a root-finding algorithm nearly identical to

the W-update of Algorithm 1 in Section 3. For the low-rank formulation, we use the matricization in Figure 2(b). The convergence criteria (l_2-norm of minimized variable(s) iterate difference) for all algorithms is 10^{-10}. All model hyper-parameters are shown in Table 2.

Table 2. Model hyper-parameters. We set $\gamma = 6.45 \times 10^{-7}$ for all algorithms except low-rank and smoothing only (where it is not used). The smoothing only formulation requires a root-finding problem for the inequality constraint, and has no Max Iteration value corresponding to block-coordinate-descent portion. For FISTA, the step-size $\alpha = \|\mathcal{L}\|_2^{-2}$, the reciprocal of the largest singular value of $\mathcal{L}$. In the joint formulation, we update η every 30 iterations; in the low-rank only, we update every 100 iterations.

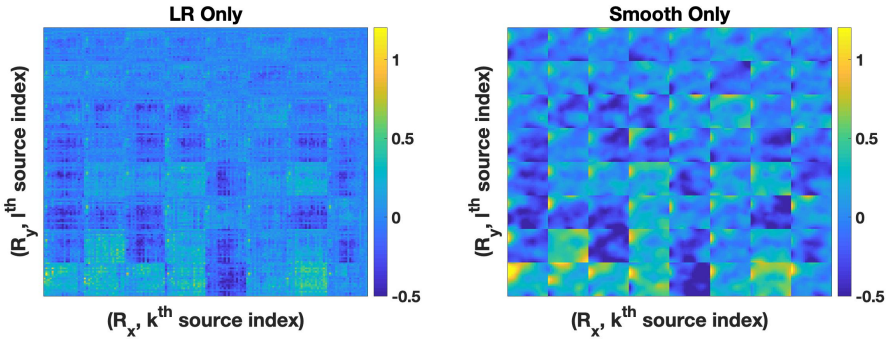
Alg	λ	η	η_f	Max Iterations	σ
Combined - VR Exact	0.0111	0.5	4.17	90	0
Combined - VR Noise	0.0111	0.5	4.17	90	3.719
FISTA	2.2222×10^{-4}	—	—	1500	—
L-BFGS	1.1111×10^{-4}	—	—	1500	—
Smooth only	—	—	—	—	3.719
Low-rank only	—	1.0	4.17	500	3.719

For the proposed relaxation algorithm, η is updated by $\eta_+ = \eta_f \eta$ for $\eta_f = \frac{\sum_j \hat{\sigma}_j}{k}$ at every 30th iteration. Here, $\hat{\sigma}_j$ are the singular values of the pre-interpolated matrix, which is the observed data on the grid and zeros where no data was observed. For low-rank only (with the l_2-norm), we update η ever 100 iterations. Increasing η can accelerate the algorithm [29]. We control η starting with a value that allows a large difference between W and LR^T, and then drive the value down such that $W \approx LR^T$. Recall that W captures smoothness and LR^T the low-rank properties of sources; the scheme drives them closer so that the final solution is both smooth and close to low rank. While Algorithm 1 is guaranteed to converge with any η value [29], the η path affects the behavior of the overall algorithm. In this work, η was updated when the feasibility measure and $\|W - LR^T\|_2$ stopped changing, i.e. when each problem was solved. A systematic approach for continuation is part of ongoing work.

The interpolated results for these two schemes and the *true* data are shown in Figure 4 with root-mean-squared (RMS) errors for both observed and interpolated data listed in Table 3. The RMS error is calculated with respect to *true* data, and is split into two categories: RMS for *true* data at locations that were observed (designated $\mathcal{A}(X_{true})$ or obs), and RMS for *true* data for locations that were not observed (given by the complement $\mathcal{A}^{\mathsf{c}}(X_{true})$ or int). Recall that in this context, *true* data refers to the residuals of the 3D model compared to the PNSN S4 1D velocity model uncorrupted by noise. The RMS is not weighted by uncertainties. While the low-rank only interpolation can accurately capture the observed data, it fails to reproduce the missing data.

In Figure 4(a), the low-rank interpolation produces mono-color bands across the tensor: when entries for a receiver coordinate are missing in every source, the low-rank mechanism places zeros in the entire row or column. Likewise, if there is a single observed receiver in an entire row or column of the matrix, that value is propagated through every other entry in that row or column. The sparsity of the data impairs low-rank only

interpolation, which gives good results at sampling ratio of 70% and above (these results not shown), but is less effective in our context. Smoothing alone, shown in Figure 4(b) can capture major model dynamics, yet overestimates observed data worse than other methods and does not overall match data as well as other methods (see Table 3). While the scale on Figure 4(b) breaches is capped at 1.2 (s), smoothing only actually has values larger than 1.5 (s) in the bright yellow section, and the observed differences are also larger (Figure 8(f)). It also makes sense that smoothing alone would out-perform low-rank alone, since the underlying model used to generate data is inherently smooth. Some of the overall residual patterns are matched (larger residual energies are correctly placed around larger observed residuals), but the magnitude of the residuals is incorrect across all sources. A blown-up version of Figure 4 for the source with the highest number of observed data ponts is shown in Figure 5. Both low-rank only and smoothing only formulations are inadequate for our application, especially with sparse data.



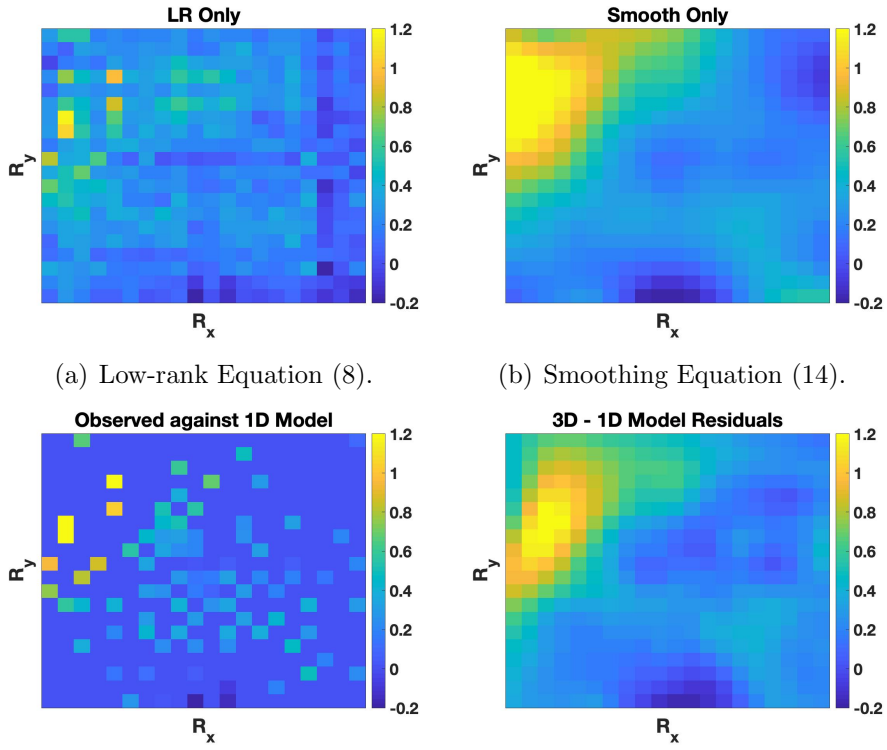
(a) Low-rank Equation (8).

(b) Smoothing Equation (14).

(c) Observed model residuals, X_{obs}.

(d) *True* model residuals, X_{true}.

Figure 4. Full tensor residuals (s) results for Low-rank and smoothing only.



(a) Low-rank Equation (8).

(b) Smoothing Equation (14).

(c) Observed model residuals.

(d) *True* model residuals.

Figure 5. Residual results (in seconds) for a single source with Low-rank and smoothing only.

In contrast, we show in Figure 6 that using both types local and global information can meaningfully interpolate the data. This can be done with the new joint formulation as well as unconstrained formulations that use FISTA and L-BFGS algorithms; so we also test the efficacy of the new approach against these two competitors. We consider two different choices of our variable relaxation algorithm: $\sigma = 0$ and $\sigma > 0$. In the latter case, we assume that we do not know the true data misfit, and use available uncertainties to set $\sigma = \sqrt{\sum_{i=1}^{n} 0.06^2} \approx 3.72(s)$ for n being the size of b. The true data misfit is actually $\|b - \mathcal{A}(X_{true})\|_2 = \sigma_{true} = 5.85(s)$ where $\mathcal{A}(X_{obs}) = b$ is our observed data. The results for tensored and single-source matrix-completion are shown in Figures 6 and 7, and are summarized in Table 3. The new approach (in bold) achieves better results than competing methods in similar amounts of compute time. Setting $\sigma > 0$ in the variable relaxation scheme produces smaller RMS values, and in particular recovers missing data with higher accuracy.

Table 3. Different formulations for model residuals with sampling rate of 15%. Terminal feasibility is $\|\mathcal{A}(X) - b\|_2 - \sigma$ at the algorithm's termination (which would mean l_2-norm data misfit where $\sigma = 0$). Note that the feasibility is calculated against observed data while RMS is calculated against *true* data. Recall that (obs) signifies $\mathcal{A}(X_{true})$, while (int) stands for $\mathcal{A}^{\mathsf{c}}(X_{true})$.

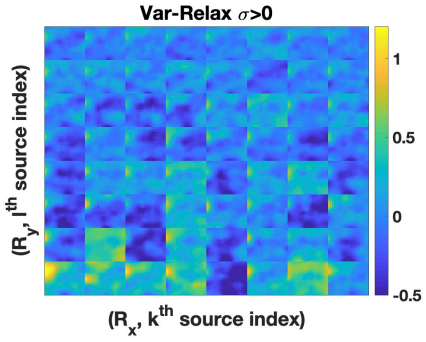
Alg	Terminal Feasibility	Time (s)	RMS (obs)	RMS (int)
Combined - VR Exact	**0.0031**	**10.81**	**0.09**	**0.110**
Combined - VR Noise	**1.18e-08**	**19.84**	**0.06**	**0.100**
FISTA	0.081	12.30	0.09	0.119
L-BFGS	0.074	57.22	0.09	0.128
Smooth only	0.0016	5.42	0.06	0.125
Low-rank only	0.058	1.27	0.08	0.216

Table 3 also shows the degree to which algorithms match the feasibility constraint. FISTA and L-BFGS use penalties rather than constraints, so the precise data-misfit level is hard to control. The terminal feasibility is 0.08(s) for FISTA and 0.075(s) for L-BFGS, which means for each individual point, the values are close to fitting the observed entries. FISTA's feasibility level settles at approximately 0.08(s) and does not change after about 1000 iterations; similar behavior is seen in L-BFGS. Variable relaxation schemes can match the feasibility constraint to a high accuracy, with the explicit inequality constraint matching close to numerical precision.

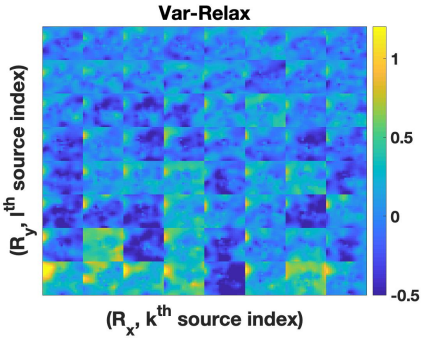
With both smoothness and low-rank regularization, fitting observed data inexactly yields a better interpolation for missing data. Figure 6 shows that variable relaxation inexact data fit (subfigure 6(a)) has fewer contrasts overall when compared to the fits obtained using exact fit and competing algorithms. Focusing on a single source in Figure 7, we can see that the inequality constraint produces a smoother image for each particular source. While each algorithm effectively captures high energy area in the northwest corner of the plot, most algorithms overestimate the amount of energy that is actually present; variable relaxation with $\sigma > 0$ gets the closest values. All algorithms tend to smooth out the observed entries, and tend to be less accurate the fewer entries there are in a designated space, notably around the corners of the receiver grid. Generally speaking, the combination of smoothness and low-rank works much better for interpolating the interior station grid rather than extrapolation near the edges. Figures 8 (full tensor) and 9 (single source) depict the absolute values of the results for each algorithm and the true value. Figure 8 shows that some sources are estimated very poorly, with the main error contributions coming further away from the mountain.

Zooming in to a particular source, Figure 9 shows how much each algorithm overestimates the high-energy in the northwest corner of the map, primarily because the observed data in that corner is also very large. Certain artifacts of the interpolation schemes are seen more vividly in the difference plots. For instance, slightly northeast of the center, there is a higher energy region where very few data points are collected. This is very pronounced in the L-BFGS case (Figure 9(d)) but less pronounced in the variable

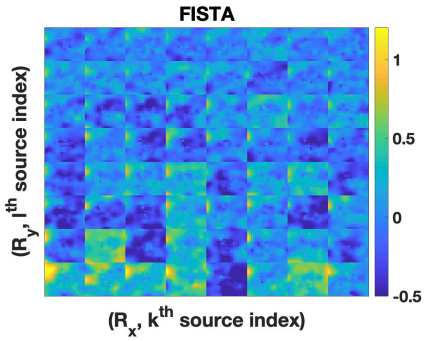
relaxation case (Figure 9(a)). Overall, the amount of error present in the $\sigma > 0$ variable relaxation case is lower than all the other schemes. This is particularly significant given the relative magnitude of the problem. We can also solve the problem **exactly** for a variety of general functions; the $\|\mathcal{L}(\cdot)\|_2^2$ smoothing-penalty is pertinent for the seismic iMUSH data, but one can select different functions for different datasets. Competing algorithms, FISTA and L-BFGS, solve approximate variants of the target problem. The simplest Formulation (9) solved via FISTA (12) is essentially the Tikhonov Formulation (1) with a Laplacian penalty. This solution performs worse overall, especially with respect to interpolating unobserved data. Hence, the approach implemented with Algorithm 1 is preferable, as it solves the target problem, achieving results with greater accuracy, while requiring similar computational resources as the alternatives.



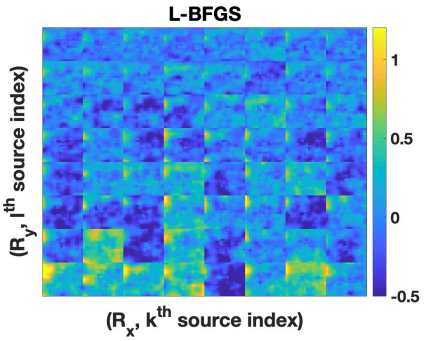
(a) Alg. 1 ($\sigma > 0$), Equation (9).



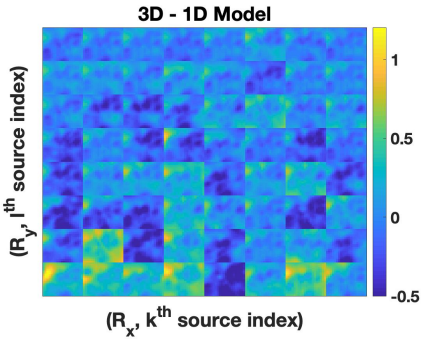
(b) Alg. 1 ($\sigma = 0$), Equation (9).



(c) FISTA (Alg. 2), Equation (12).

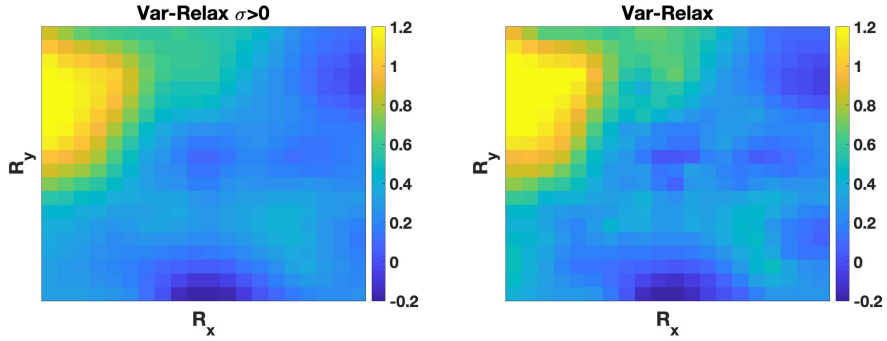


(d) L-BFGS, Equation (13).

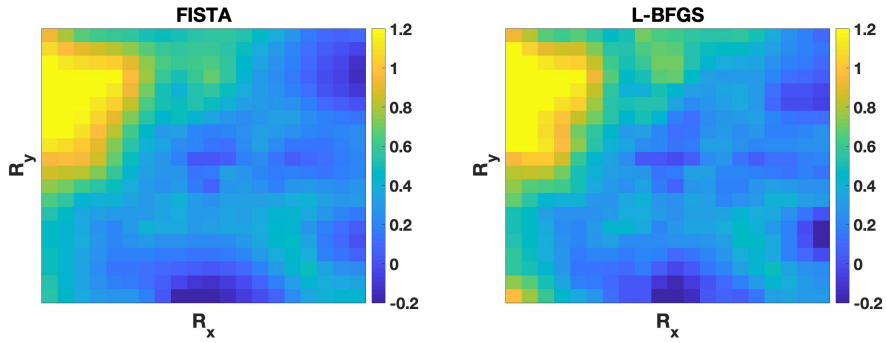


(e) *True* model residuals, X_{true}.

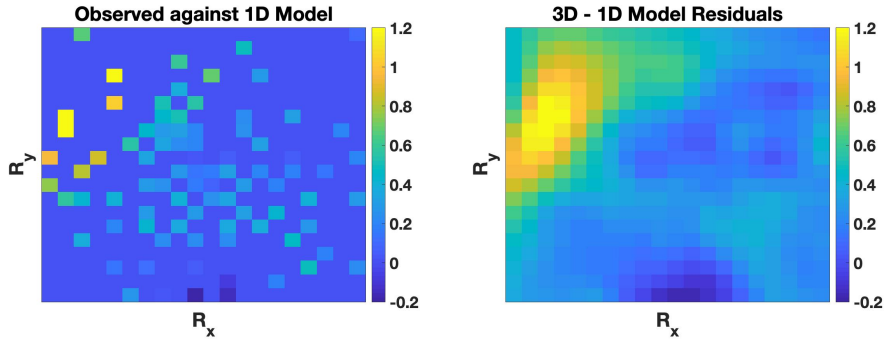
Figure 6. Full tensor residual results (s) for different algorithms and formulations.



(a) Alg. 1 ($\sigma > 0$), Equation (9). (b) Alg. 1 ($\sigma = 0$), Equation (9).

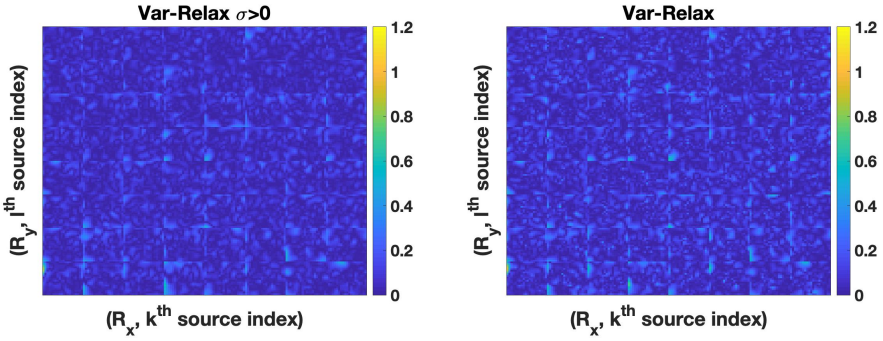


(c) FISTA Alg. 2, Equation (12) (d) L-BFGS, Equation (13).

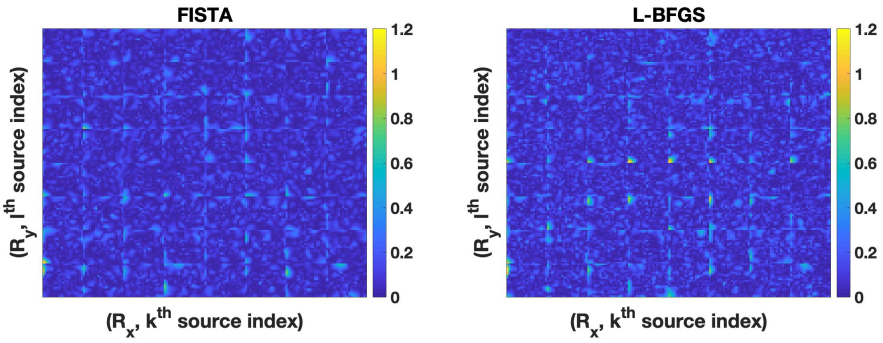


(e) Observed model residuals. (f) *True* model residuals.

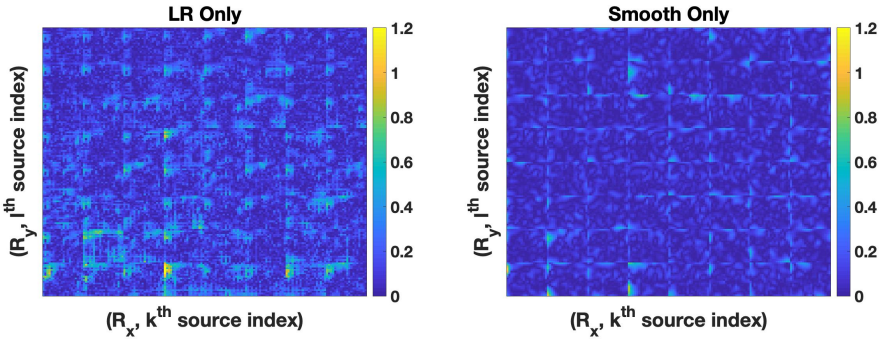
Figure 7. Single source residuals (s) for the different algorithms.



(a) Alg. 1 ($\sigma > 0$), Equation (9). (b) Alg. 1 ($\sigma = 0$), Equation (9).

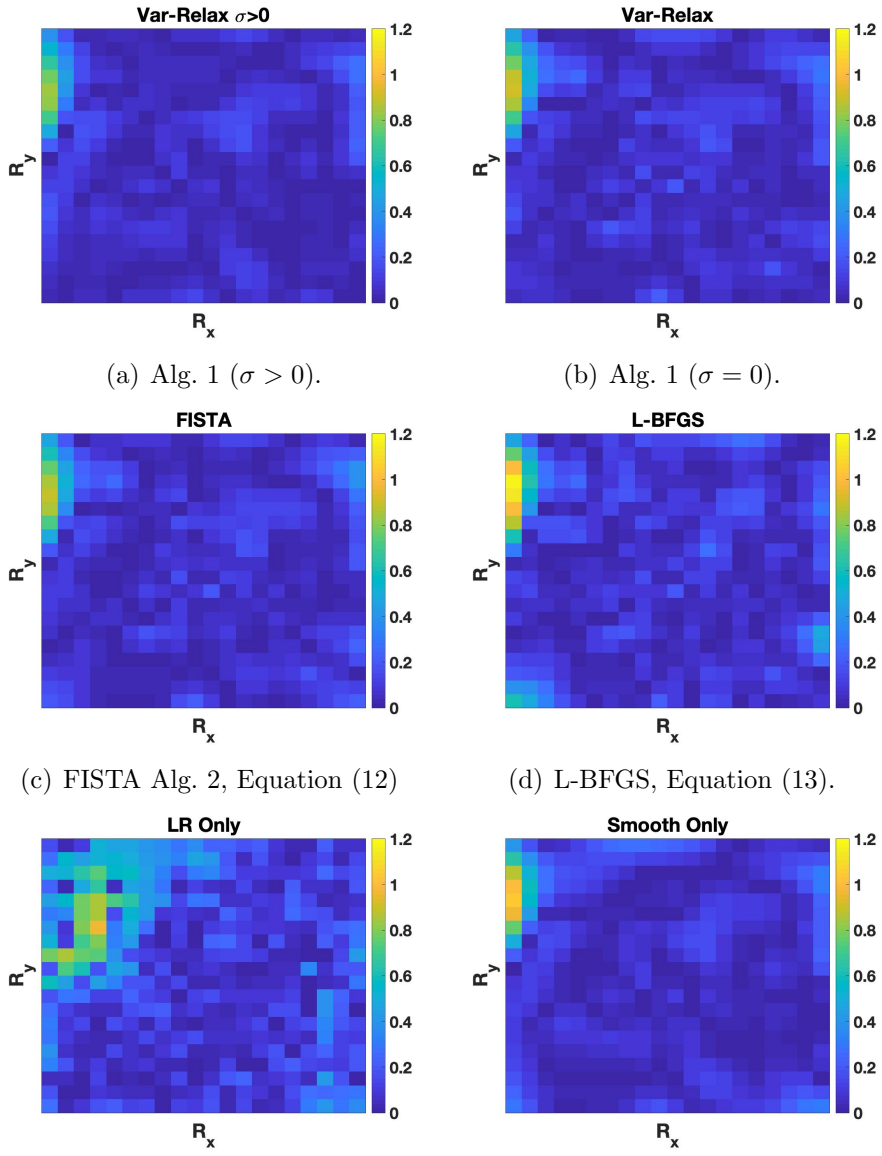


(c) FISTA Alg. 2, Equation (12) (d) L-BFGS, Equation (13).



(e) Low-Rank Equation (8). (f) Smoothing Equation (14).

Figure 8. Full tensor $|X - X_{true}|$ for all algorithms. Note the significant scaling difference in the FISTA result.



(a) Alg. 1 ($\sigma > 0$).

(b) Alg. 1 ($\sigma = 0$).

(c) FISTA Alg. 2, Equation (12)

(d) L-BFGS, Equation (13).

(e) Low-Rank Equation (8).

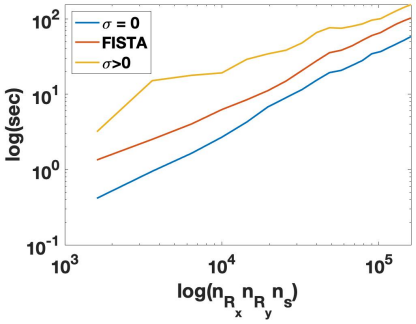
(f) Smoothing Equation (14).

Figure 9. $|X - X_{true}|$ for all algorithms in a single source.

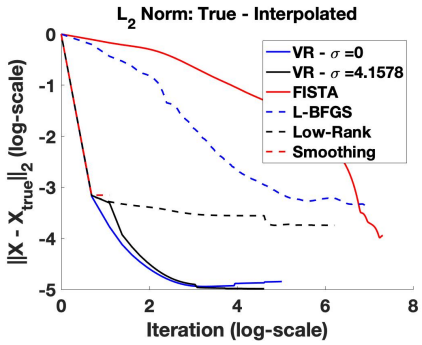
Next, we turn to singular value decay and convergence history. The singular value decay is shown in Figure 10(c), and a log-log convergence plot appears in Figure 10(b). With the exception of the smoothing-only implementation, all algorithms match the SVD decay well. In Figure 10(b), L-BFGS and FISTA have similar convergence histories but L-BFGS has relatively slow compute time compared to the FISTA (and indeed all other algorithms) when achieving a satisfactory answer. The proposed approach converges faster than competing methods, and also matches the SVD decay of the ground truth datasets.

Finally, we address the scalability of our algorithm and problem. Figure 5 shows our results in computational time and root mean-squared error as we increase the number

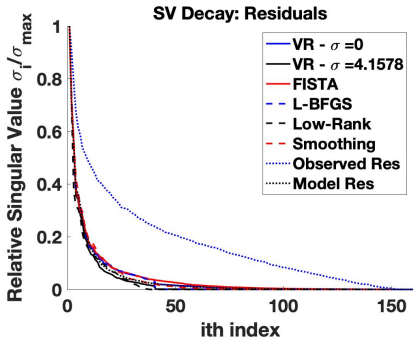
of sources from 2^2 to 20^2, the latter of which being the nearest perfect square to the total number of sources observed in the iMUSH experiments. The number of iterations for each algorithm remains the same, as do the hyperparameters for these algorithms. The only change made is that for $\sigma > 0$; here, we make it so the average root mean-squared error is 0.06. Much beyond 20^2, the matrix storage for the smoothing operator becomes infeasible for a laptop computer, and source-separation parallelization would have to occur [14]. From this, we can see that the algorithm, both with and without root-finding, scales in similar computational time to FISTA.



(a) Log-log plot of time (s) as number of sources increases.



(b) Data misfit decay.



(c) SVD decay for interpolated matrices.

Figure 10. Convergence information for different algorithms.

6. Application to Real Data

Algorithm 1 can be easily modified to fit real data by changing the constraint, $\mathcal{A}(\cdot)$, to another operator that manages the fit between interpolation data and the observed stations. The formulation (9) is represented then as

$$\min_{L,R,W} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 + \frac{1}{2\gamma}\|\mathcal{L}(W)\|_2^2 + \frac{1}{2\eta}\|W - LR^T\|_F^2 \quad \text{s.t.} \quad \|\mathcal{D}(W) - b\|_2 \leq \sigma \quad (15)$$

where the only difference between Formulation (9) and Formulation (15) is the substitution of the restriction operator $\mathcal{A}(\cdot)$ with an interpolation operator $\mathcal{D}(\cdot)$:

$\mathbb{R}^{n_{R_x}\sqrt{n_s}\times n_{R_y}\sqrt{n_s}} \to \mathbb{R}^{n_d}$. The only requirement on the interpolation operator is that it is linear; i.e. can be represented as a matrix. In this problem, the interpolation operator takes the tensor W, permutes it and then uses a 2D linear Lagrange interpolation function to map it to the observed field data for all sources, which is in $b \in \mathbb{R}^{n_d}$. Hence, step 6 of Algorithm 1 becomes

$$W_{k+1} = \arg\min_{W} \|\mathcal{L}(W)\|_2^2 + \frac{1}{2\eta}\|W - L_{k+1}R_{k+1}^T\|_F^2 \quad \text{s.t. } \|\mathcal{D}(W) - b\|_2 \leq \sigma \tag{16}$$

which collapses to

$$w(\lambda) := \arg\min_{w} \frac{1}{2\gamma}\|L_\nabla w\|_2^2 + \frac{1}{2\eta}\|w - d_{k+1}\|_F^2 + \frac{\lambda}{2}\|Dw - b\|^2, \tag{17}$$

where $D \in \mathbb{R}^{n_d \times n_{R_x} n_{R_y} n_s}$ is the matrix representing our interpolation operator. Hence our solution for w is

$$w(\lambda) = \left(\frac{1}{\gamma}(L_\nabla^T L_\nabla) + \frac{1}{\eta}I + \lambda D^T D\right)^{-1}\left(\lambda D^T b + \frac{1}{\eta}d_{k+1}\right).$$

Another way of formulating this problem is to take the analytic perspective; that is, we let $w \in \mathbb{R}^{n_d}$ and modify the problem as such

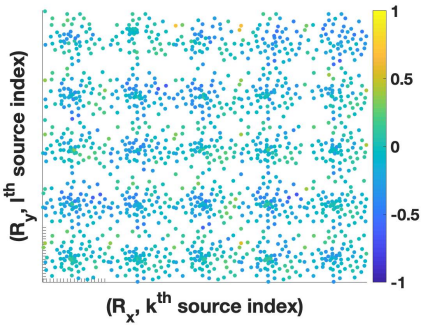
$$\min_{L,R,w} \frac{1}{2}\|L\|_F^2 + \frac{1}{2}\|R\|_F^2 + \frac{1}{2\gamma}\|\mathcal{L}\left(\mathcal{D}(w)\right)\|_2^2 + \frac{1}{2\eta}\|\mathcal{D}(w) - LR^T\|_F^2 \quad \text{s.t.} \quad \|w - b\|_2 \leq \sigma \tag{18}$$

where $\mathcal{D}(\cdot) : \mathbb{R}^{n_d} \to \mathbb{R}^{n_{R_x}\sqrt{n_s}\times n_{R_y}\sqrt{n_s}}$; this makes our root-finding process a bit easier to implement. However, both approaches yield the same results.

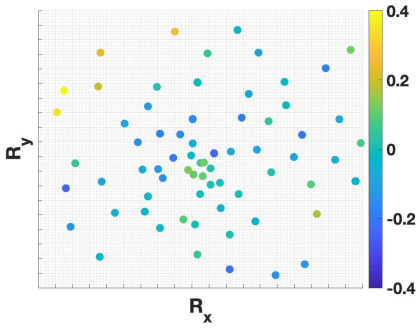
For real data, the number of sources is reduced to $n_s = 25$, while the grid remains the same. We withhold 10% of the observed data for cross-validation purposes, and perform Algorithm 1 with Formulation 15 for $\sigma = 0$. The results for this are shown in Figure 11 and 12 and are summarized in Table 4. These results show that our accuracy is greater the closer the points are to the center of the mountain. This makes sense, as there are more sensors closer to the mountain than in the surrounding area. However, we see an overall smoothness in the interpolated solution. This is a route for further exploration, and possibly the makings of another paper.

Table 4. Results for real data with a sub-sampling rate of 10%. We ran the modification of Algorithm 1 with Equation 15 with $\sigma = 0$. Since we don't have *true* data for comparison, this is simply compared to withheld data to measure accuracy, despite uncertainty in observed data.

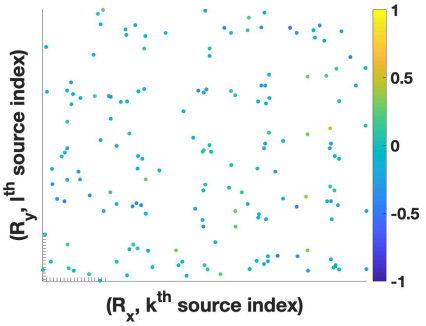
Data	Mean $\lvert X - X_{obs}\rvert$	RMS	Median $\lvert X - X_{obs}\rvert$
Observed	0.14	1.3e-3	0.115
Withheld	0.15	1.4e-3	0.133



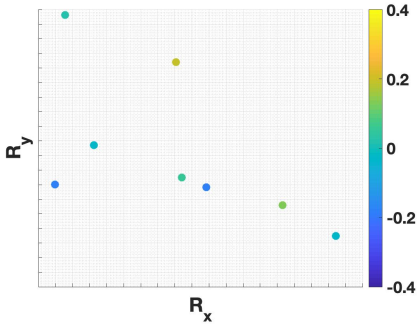
(a) Observed.



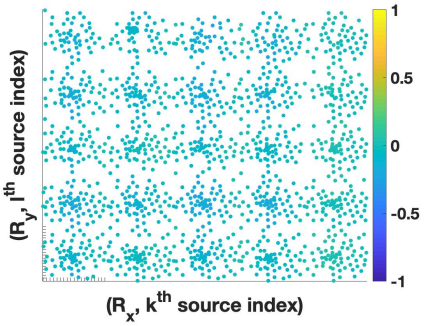
(b) Observed single source.



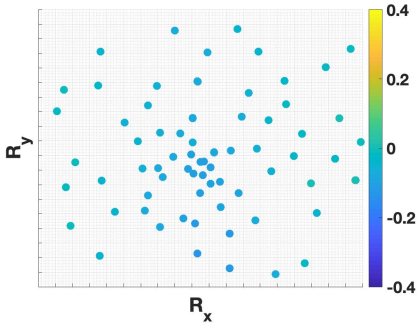
(c) Omitted.



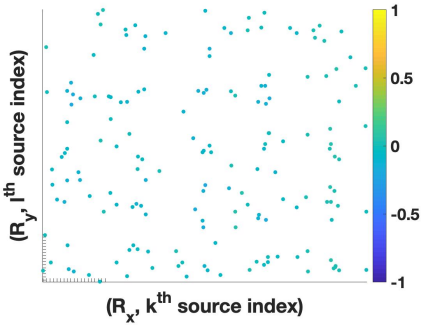
(d) Omitted single source.



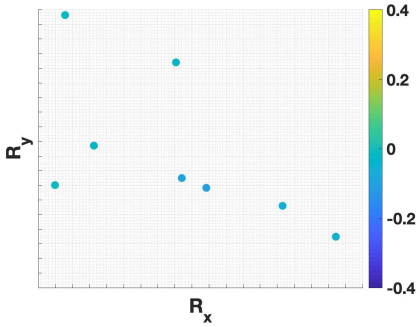
(e) Interpolated at observed receivers.



(f) Interpolated at observed receivers for a single source.

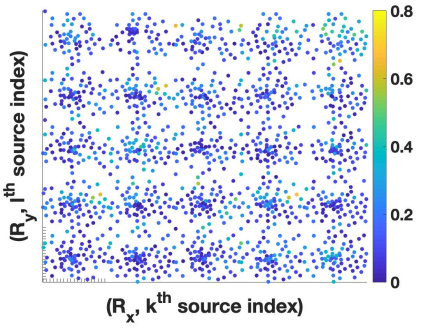


(g) Interpolated at hidden receivers.

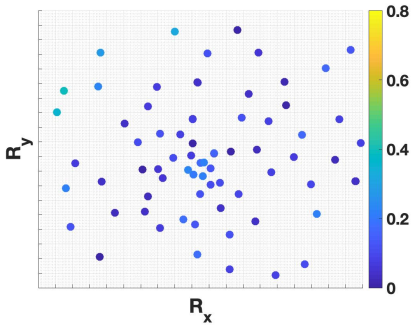


(h) Interpolated at hidden receivers for a single source.

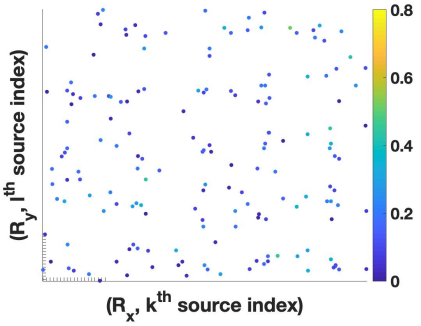
Figure 11. Residuals (s) for Algorithm 1 with real IMUSH data.



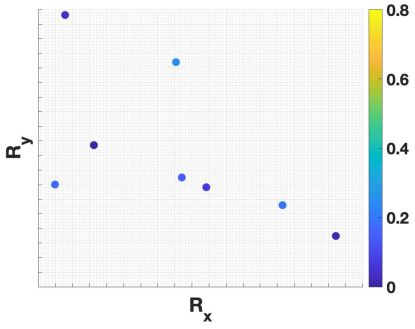
(a) Difference in observed



(b) Difference in observed for a single source.



(c) Difference at hidden receivers.



(d) Difference at hidden receivers for a single source.

Figure 12. Receiver Differences for real IMUSH data.

7. Conclusions and Future Directions

Travel time tomography suffers from data collection constraints, reducing model resolution. We proposed an interpolation scheme that combines both local smoothness and low-rank information from a carefully chosen tessellation of the data to estimate residual values at prospective stations. To implement the scheme, we developed a new relaxation approach that is flexible enough to allow multiple regularizers, and efficient in practice. We used it to estimate data from missing stations with a relatively high degree of accuracy (measured against a synthetic ground-truth dataset), in the presence of observation noise in available data. The algorithm is more flexible than available alternatives, and in particular can fit available data to a prescribed error level. The new approach is competitive with standard alternatives, and offers new functionality to interpolate with both local and global structure over data fitting constraints. It is important to note that none of the methods do well estimating where there are very few stations. This can be observed in Figure 9, where the top left corner of each graph has a high residual and no stations present towards border of the receiver space. The proximity of at least one station increases the accuracy of our scheme.

8. Acknowledgements

R. Baraldi acknowledges support from the Department of Energy Computational Science Graduate Fellowship, which is provided under grant number DE-FG02-97ER25308. C. Ulberg and K. Creager were supported by National Science Foundation grant number EAR-1358512. The work of A. Aravkin was supported by the Washington Research Foundation Data Science Professorship.

9. References

[1] A. Aravkin, R. Kumar, H. Mansour, B. Recht, and F. J. Herrmann. Fast methods for denoising matrix completion formulations, with applications to robust seismic data interpolation. *SIAM Journal on Scientific Computing*, 36(5):237–266, 2014.

[2] A. Y. Aravkin, J. V. Burke, D. Drusvyatskiy, M. P. Friedlander, and S. Roy. Level-set methods for convex optimization. *arXiv preprint arXiv:1602.01506*, 2016.

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[4] E. J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, June 2010.

[5] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–723, Apr 2009.

[6] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies. *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[7] D. Davis and W. Yin. Convergence rate analysis of several splitting schemes. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 115–163. Springer, 2016.

[8] G. Hennenfent and F. J. Herrmann. Simply denoise: wavefield reconstruction via jittered undersampling. *Geophysics*, 73(3):19–28, 2008.

[9] F. J. Herrmann and G. Hennenfent. Non-parametric seismic data recovery with curvelet frames. *Geophysical Journal International*, 173(1):233–248, 2008.

[10] J. A. Hole and B. C. Zelt. 3-D finite-difference reflection traveltimes. *Geophysical Journal International*, 121(2):427–434, 1995.

[11] E. Kiser, A. Levander, C. Zelt, B. Schmandt, and S. Hansen. Focusing of melt near the top of the Mount St. Helens (USA) magma reservoir and its relationship to major volcanic eruptions. *Geology*, 2018.

[12] E. Kiser, A. Levander, C. A. Zelt, I. Palomeras, K. Creager, C. W. Ulberg, B. Schmandt, S. M. Hansen, S. H. Harder, G. A. Abers, and K. Crosbie. Three-dimensional velocity models of the Mount St. Helens magmatic system using the iMUSH active-source data set. In *AGU Fall Meeting Abstracts*, Dec 2017.

[13] R. Kumar, C. Da Silva, O. Akalin, A. Y. Aravkin, H. Mansour, B. Recht, and F. J. Herrmann. Efficient matrix completion for seismic data reconstruction. *Geophysics*, 80(5):97–114, 2015.

[14] R. Kumar, H. Wason, and F. J. Herrmann. Source separation for simultaneous towed-streamer marine acquisition — a compressed sensing approach. *Geophysics*, 80(6):WD73–WD88, 11 2015.

[15] M. Landrø. Uncertainties in quantitative time-lapse seismic analysis. *Geophysical Prospecting*, 50(5):527–538, 2002.

[16] M. Landrø. The effect of noise generated by previous shots on seismic reflection data. *Geophysics*, 73(3):9–17, 2008.

[17] S. D. Malone and G. L. Pavlis. Velocity structure and relocation of earthquakes at Mount St. Helens. In *EosTrans AGU*, volume 64, page 895, 1983.

[18] J. Martins, P. Sturdza, and J. Alonso. The connection between the complex-step derivative

approximation and algorithmic differentiation. In *39th Aerospace Sciences Meeting and Exhibit*, page 921, 2001.

[19] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2000.

[20] L. Oneto, S. Ridella, and D. Anguita. Tikhonov, Ivanov and Morozov regularization for support vector machine learning. *Machine Learning*, 103(1):103–136, 2016.

[21] W. S. Phillips and M. C. Fehler. Traveltime tomography: A comparison of popular methods. *Geophysics*, 56(10):1639–1649, 1991.

[22] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, Aug 2010.

[23] M. D. Sacchi, T. J. Ulrych, and C. J. Walker. Interpolation and extrapolation using a high-resolution Discrete Fourier transform. *IEEE Transactions on Signal Processing*, 46(1):31–38, Jan 1998.

[24] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. 2009.

[25] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, Philadelphia, 1 edition, 1997.

[26] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of optimization theory and applications*, 109(3):475–494, 2001.

[27] C. W. Ulberg, K. Creager, S. C. Moran, G. A. Abers, K. Crosbie, R. S. Crosson, R. P. Denlinger, W. A. Thelen, E. Kiser, A. Levander, and O. Bachmann. Imaging seismic zones and magma beneath Mount St. Helens with the iMUSH Broadband Array. In *AGU Fall Meeting Abstracts*. Submitted, Dec 2017.

[28] J. E. Vidale. Finite-difference calculation of traveltimes in three dimensions. *Geophysics*, 55(5):521–526, 1990.

[29] P. Zheng and A. Aravkin. Fast methods for nonsmooth nonconvex minimization. *arXiv preprint arXiv:1802.02654*, 2018.