Applications of low-rank compressed seismic data to full waveform inversion and extended image volumes

Curt Da Silva[*,1,2], Yiming Zhang[*,1], Rajiv Kumar[1] and Felix J. Herrmann[1,3]

[1] Seismic Laboratory for Imaging and Modeling (SLIM), University of British Columbia

[2] Department of Mathematics, University of British Columbia

[3] Georgia Institute of Technology

[*] Equal contributors

(January 12, 2019)

Running head: **Applying low-rank methods**

ABSTRACT

Conventional oil and gas fields are increasingly difficult to explore and image, resulting in the call for more complex wave-equation-based inversion algorithms that require dense long-offset samplings. Consequently, there is an exponential growth in the size of data volumes and prohibitive demands on computational resources. We propose a method to compress and process seismic data directly in a low-rank tensor format, which drastically reduces the amount of storage required to represent the data. Seismic data exhibits low-rank structure in a particular transform domain, which can be exploited to compress the dense data in one extremely storage-efficient tensor format when the data is fully sampled, or interpolated when the data has missing entries. In either case, once our data is represented in its compressed tensor form, we propose an algorithm to extract source or receiver gathers directly from

the compressed parameters. This extraction process can be done on-the-fly directly on the compressed data and does not require scanning through the entire dataset in order to form shot gathers. We apply this shot-extraction technique in the context of stochastic full-waveform inversion as well as forming full subsurface image gathers through probing techniques and demonstrate the minor differences between using the full and compressed data, while drastically reducing the total memory costs.

INTRODUCTION

Seismic processing and inversion are challenging problems when a large amount of data is involved and the resulting high computational costs. State-of-art acquisition techniques, such as high-density and wide-azimuth samplings, are widely used to avoid aliasing and inaccuracy in subsequent processing procedures. Unfortunately, for realistically-sized 3D surveys, the multidimensional nature of the data results in the so-called "curse of dimensionality"–an exponential increase in the number of data points as the size of each individual dimension increases. Data sets can easily range from terabytes to petabytes in size and the resulting wave-equation-based algorithms must compute solutions to tens of thousands of partial differential equations (PDEs).

Low-rank techniques are one approach to mitigate the enormous costs of traditional algorithms working on full data volumes. The underlying idea in this case is that the data volume, when organized as a matrix or tensor in a particular fashion, should have quickly decaying singular values. In this case, the underlying volume can be well approximated by a low-rank matrix or tensor. The number of parameters needed to describe such a low-rank matrix is much smaller than the ambient space, i.e., it is a simpler object than an arbitrary matrix. Subsequently, operations acting directly on the compressed form of the matrix/tensor are significantly cheaper. Compressed sensing (Donoho, 2006), and likewise matrix completion (Candès and Recht, 2009), theory tells us that we can subsample our data, at a rate commensurate with the underlying dimensionality, i.e., the rank, rather than the ambient space. To recover our original signal, we solve an associated optimization problem (Candès and Recht, 2009; Candès and Tao, 2010; Recht, 2011).

There has been a surge of interest in recent years in applying low-rank techniques to

seismic data problems, including interpolation (Ma, 2013; Kumar et al., 2015; Aravkin et al., 2014; Trickett et al., 2010), noise attenuation (Freire and Ulrych, 1988; Bekara and Van der Baan, 2007; Nazari Siahsar et al., 2016), estimation of primaries by sparse inversion (Jumah and Herrmann, 2014), simultaneous source deblending (Cheng and Sacchi, 2015; Kumar et al., 2016), and travel-time tomography (Stork, 1992). Extensions of these low-rank ideas to multi-dimensional tensors in the seismic context can be found in, e.g., Kreimer and Sacchi (2012), Kreimer et al. (2013), Trickett et al. (2013), Da Silva and Herrmann (2015). The work of Abubakar et al. (2012), Li et al. (2011) uses a low-rank decomposition of the source fields, organized as a matrix with each wavefield in the columns, which is in turn based on the so-called CUR decomposition (Boutsidis and Woodruff, 2017). Applying the CUR decomposition to the data matrix could be used to provide a low rank matrix decomposition, but does not exploit the full redundancy of the data volume when considering its tensor structure. This method is also unable to handle missing data, as it requires algorithm-driven access to entries of the underlying matrix. The tensor completion methods considered in this work, on the other hand, are able to handle data volumes with missing entries, by design.

Low-rank methods are not the only technique to represent high-dimensional wavefields in a low-dimensional manner. Transform-based methods consider a representation of the data volume in a domain such as wavelets (Villasenor et al., 1996) or curvelets (Herrmann et al., 2007; Herrmann and Hennenfent, 2008) and store only a small subset of the total coefficients. While sufficient for the purposes of storing, retrieving, or interpolating data, these methods become particularly cumbersome if one is interested in extracting specific subsets from the compressed volume. Furthermore, for curvelets, since they are four times redundant in 2D (Candes et al., 2006) and five times redundant in 3D (Ying et al., 2005), the benefits of sparsity are somewhat dwarfed by the need to handle such large coefficient vectors in memory,

while other orthogonal transforms, such as wavelets, typically compress seismic waveforms much less efficiently than curvelets, relative to their ambient dimensionality. On a per-byte basis, however, curvelets do not offer the same compression rate as wavelets. Additionally, transform-based methods can also be much more complicated to implement compared to low-rank methods, the latter merely involving efficient matrix-matrix multiplications.

In practice, acquired data frequently contain missing entries, either due to budget, time, or environmental constraints. There is a variety of mathematical techniques used to estimate the fully sampled data volume in this case. By exploiting the sparsity or correlations among coefficients, transform-based approaches can be used for interpolation, in the case of representations such as Radon (Kabir and Verschuur, 1995; Wang et al., 2010), Fourier (Sacchi et al., 2009; Curry, 2010), wavelets (Villasenor et al., 1996), and curvelets (Hennenfent and Herrmann, 2006; Herrmann and Hennenfent, 2008). These methods have been successful for interpolation, but the computational costs of these approaches can be large. Recent development in matrix completion (Oropeza and Sacchi, 2011; Kumar et al., 2015) and tensor completion (Kreimer and Sacchi, 2012; Trickett et al., 2013; Da Silva and Herrmann, 2015) techniques can substantially lower these costs. Singular value decomposition (SVD)-based implementations of matrix and tensor completion are ill-advised to solve these problems as these methods require a one or more SVD computation of large matrices at each iteration, in addition to requiring a large number of iterations. As a result, practitioners using SVD-based methods often resort to working with small subsets of the data at a time, i.e., windowing. As demonstrated by Kumar et al. (2015), the act of windowing can even significantly degrade the quality of the recovered data.

CONTRIBUTIONS

Low-rank methods possess significant advantages over transform-based methods as they often have reduced memory and computational costs due to their lower inherent dimensionality. This redundancy arises from the fact that multiple source experiments ultimately image the same underlying earth. In this work, we demonstrate that possessing a representation of our data volume in a low-rank tensor format will enable us to extract relevant subsets of the data directly from its compressed form. Here, we use the Hierarchical Tucker (HT) format (Hackbusch and Kühn, 2009; Grasedyck, 2010) to represent our seismic data, as it requires significantly fewer parameters to represent the data relative to its uncompressed size and one can develop algorithms directly for the compressed parameters using tensor algebra. These benefits outweigh its seemingly complicated construction. We consider two instances of data sampling in this work. First, if our data volume is fully sampled, we use existing, computationally efficient methods to compress a tensor into HT form (Grasedyck, 2010). Second, if our data has missing entries, we use existing methods for interpolating tensors with missing entries in the HT format (Da Silva and Herrmann, 2015). Once our data is in compressed HT form, either through compression or interpolation, we develop an algorithm for extracting arbitrary source or receiver gathers in an on-the-fly manner from the compressed HT parameters, rather than having to form the full data volume explicitly. This approach gives us the ability to extract source/receiver gathers at arbitrary locations and reduces the memory costs of working with the full seismic data by two orders of magnitude at the low frequencies. We consider two case studies for integrating our data extraction technique: a 3D full-waveform inversion (FWI) example and an example forming 3D subsurface image gathers. Our results differ only marginally from using the fully sampled original volume compared to its compressed form but drastically reduces the memory requirements. The

outline of the methodology proposed in this work is shown in Figure 1.

[Figure 1 about here.]

NOTATION

We represent vectors (i.e., one-dimensional quantities) as boldfaced lowercase letters, e.g., **xyz** and matrices and tensors (i.e., multi-dimensional quantities) as boldfaced uppercase letters, e.g., $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$. We use $\mathbf{X}^H$ to denote the conjugate transpose of $\mathbf{X}$ and $\mathbf{X}^T$ to denote the real transpose of $\mathbf{X}$, and $\|\cdot\|$ stands for ℓ_2 norm. The vectorization operator vec stacks the columns of a matrix, or the multiple dimensions of a tensor, into a vector. Let I_n denote the identity operator of size n. The complex conjugate of a vector $\mathbf{z}$ is written as $\overline{\mathbf{z}}$.

We write $\mathbf{H}(\mathbf{m})$ to denote the discretization of the monochromatic constant-density Helmholtz equation $\nabla^2 + \omega^2\mathbf{m}(\mathbf{x})$, which is a matrix of size $N \times N$ where $N = n_z n_x$ in 2D and $N = n_x n_y n_z$ in 3D. Here, $\mathbf{m}$ is the slowness squared, ∇^2 is the Laplacian, and ω is the temporal angular frequency in radians.

The *matricization* of a $d-$dimensional tensor $\mathbf{X} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$, along the $i-$th coordinates, denoted $\mathbf{X}^{(i)}$, reshapes the tensor so that the coordinates corresponding to the indices i are along the rows and the remaining indices are along the columns. For example, for a 5D tensor $\mathbf{X} \in \mathbb{C}^{n_1 \times n_2 \times n_3 \times n_4 \times n_5}$, $\mathbf{X}^{(1)}$ is a matrix with the size $n_1 \times n_2 n_3 n_4 n_5$–the first dimension along the rows and dimensions 2,3,4,5 along the columns. Similarly, $\mathbf{X}^{(1,3)}$ is an $n_1 n_3 \times n_2 n_4 n_5$ matrix that has dimensions 1,3 along the rows and dimensions 2,4,5 along the columns. A Matlab code implementing these examples can be written as

```
X_1 = reshape(X,n1,n2*n3*n4*n5)
```

```
X_13 = reshape(permute(X,[1 3 2 4 5]),n1*n3,n2*n4*n5).
```

Given a $d-$dimensional tensor $\mathbf{X} \in \mathbb{C}^{n_1 \times n_2 \times \cdots \times n_d}$ and a linear operator $\mathbf{A} \in \mathbb{C}^{m_i \times n_i}$, the *multilinear product* of $\mathbf{X}$ and $\mathbf{A}$ in the $i-$th dimension, denoted $\mathbf{Y} = \mathbf{A} \times_i \mathbf{X}$, is defined in terms of matricizations as

$$\mathbf{Y}^{(i)} = \mathbf{A}\mathbf{X}^{(i)} \tag{1}$$

In words, we are matricizing the tensor along the $i-$th dimension, applying the operator $\mathbf{A}$, and then reshaping this matrix into a tensor. The resulting tensor $\mathbf{Y}$ is of size m_i in the i^{th} dimension and size n_j for each $j \neq i$. In seismic terms, this concept encapsulates many familiar operations. For instance, if one has a data volume $\mathbf{D}$ with coordinates (source x, source y, receiver x, receiver y, time), performing a Fourier transform along the time axis, with Fourier matrix $\mathbf{F}_t$, would be written as $\mathbf{F}_t \times_5 \mathbf{D}$. In Matlab code, this operation would be written as

```
D_5 = reshape(permute(D,[5 1 2 3 4]),n5,n1*n2*n3*n4)
Y_5 = F_t*X_5
Y = permute(reshape(Y_5,n5,n1,n2,n3,n4),[2 3 4 5 1]).
```

An important property of multilinear products that we will make use of is that they commute, e.g., for matrices $\mathbf{A}$ and $\mathbf{B}$ and a tensor $\mathbf{C}$ of appropriate size, the following holds:

$$\begin{aligned} \mathbf{A} \times_1 \mathbf{B} \times_2 \mathbf{C} &= \mathbf{A} \times_1 (\mathbf{B} \times_2 \mathbf{C}) \\ &= \mathbf{B} \times_2 (\mathbf{A} \times_1 \mathbf{C}) \\ &= \mathbf{B} \times_2 \mathbf{A} \times_1 \mathbf{C}. \end{aligned} \tag{2}$$

HIERARCHICAL TUCKER REPRESENTATION FOR SEISMIC DATA

Multilinear or tensor algebra is a branch of computational mathematics that has become increasingly ubiquitous in the era of "big data". As noted previously, the number of entries of a tensor grows exponentially with increasing number of dimensions, i.e., so-called "curse of dimensionality", which incurs substantial memory and computational cost to process such objects. As such, various low-rank tensor formats, such as the Canonical Polyadic (CP) format (Carroll and Chang, 1970) and the Tucker format (De Lathauwer et al., 2000), have been developed to exploit redundancies among the various dimensions and represent the full tensor in a more compact manner. The HT tensor format is a novel structured tensor format introduced in Hackbusch and Kühn (2009), which results in the number of HT parameters growing linearly with the number of dimensions rather than exponentially. This is extremely storage-efficient and computationally tractable for parametrizing high-dimensional problems.

Although it is slightly technical to define the HT format explicitly, we define some preliminary components below. Given a $d-$dimensional tensor, we associate a *dimension tree* $\mathcal{T}$ as a binary tree with the root being assigned the label $t_{\text{root}} = \{1, 2, \ldots, d\}$. Every node t is assigned a set of labels $t \subset \{1, \ldots, d\}$ and its left and right children, denoted t_l and t_r, respectively, form a disjoint partition of t, i.e., $t = t_l \cup t_r$, $t_l \cap t_r = \emptyset$. The dimension tree specifies how we *separate* groups of dimensions from each other, where "separate" is understood in the sense of the SVD. An example of a dimension tree for a $6-$dimensional tensor is shown in Figure 2. Here, we interpret this figure as dimensions $\{1, 2, 3, 4\}$ being separated from dimensions $\{5, 6\}$. Dimensions $\{1, 2\}$ of the tensor are further split apart from dimensions $\{3, 4\}$ on the left side of the tree, and so on. We shall see a specific instance

of a dimension tree for seismic data later in this work.

[Figure 2 about here.]

In lieu of an elaborate technical definition of the HT format, which can be found in Grasedyck (2010), we instead provide a pictorial representation of its construction in Figure 3. For a $4-$dimensional tensor $\mathbf{X}$, we consider *matricizing* it along its first two dimensions, resulting in $\mathbf{X}^{(1,2)}$. We can consider an "SVD-like" decomposition of this matrix, which splits it into a product of three smaller matrices. Noticing that the matrix $\mathbf{U}_{(1,2)}$ contains dimensions $(1,2)$ of the tensor along the rows, we can further reshape it into a $n_1 \times n_2 \times k_{1,2}$ cube that can be further decomposed in this multilinear fashion. We apply the same recursive splitting to the matrix $\mathbf{U}_{(3,4)}$, although it is not shown. As a result of this recursive construction, the intermediate matrices $\mathbf{U}_{(1,2)}, \mathbf{U}_{(3,4)}$, which contain multiple spatial dimensions and hence are onerous to construct, do not have to be formed explicitly. Instead, once we have knowledge of the small matrices $\mathbf{U}_i$, $i = 1, 2, 3, 4$, small $3-$dimensional tensors $\mathbf{B}_i$, $i = (1,2), (3,4)$, and matrix $\mathbf{B}_{(1,2,3,4)}$, running this multilinear construction in reverse will reconstruct the entire tensor $\mathbf{X}$. As a result, the total number of parameters needed to specify a $d-$dimensional HT tensor is bounded from above by $dNK + (d-2)K^3 + K^2$, where N is the maximum spatial sampling in all dimensions and K is the maximum rank. Tensors that can be represented in the HT format have significantly fewer parameters than the N^d parameters required to represent the full data. For instance, if $N = 100, d = 4, K = 20$, then the number of HT parameters needed to represent the tensor is 24400, compared to 10^8 for the pointwise array data.

[Figure 3 about here.]

In the seismic context, our data has five dimensions (source x, source y, receiver x, receiver y, time). We process individual temporal frequency slices, one at a time, resulting in data volumes under consideration with dimensions (source x, source y, receiver x, receiver y). For notational simplicity, we abbreviate these dimensions as (sx, sy, rx, ry), respectively. Each frequency slice has dimensions $n_{\text{sx}} \times n_{\text{sy}} \times n_{\text{rx}} \times n_{\text{ry}}$. For the purposes of compression, we need to ensure that the data volume has quickly decaying singular values in each of the relevant matricizations. Noting that the organization of the tensor has a major impact on its low-rank nature, we typically permute our data from canonical organization (source x, source y, receiver x, receiver y) into a non-canonical organization (source x, receiver x, source y, receiver y), which leads to faster decaying singular values for the associated matricizations (Aravkin et al., 2014; Da Silva and Herrmann, 2015). This permutation results in a data volume that is much more amenable to compression in the HT format compared to the standard (source x, source y, receiver x, receiver y) organization. The corresponding dimension tree is shown in Figure 4 and outlines the free parameters that are stored at each node of the tree. One potential reasoning is contained in Demanet (2006), wherein the author uses this organization in the context of compressing solution operators of the wave equation. Since the data itself is the Green's function of the wave equation restricted to the surface, we find that this non-canonical organization enables fast singular value decay of the data volume. If we are in the missing sources or receivers context, considering our data in the non-canonical ordering results in growth of the singular values in the corresponding matricizations of the tensor. This leads to more favourable reconstruction conditions, as noted in Kumar et al. (2015). Seismic data is particularly compressible in the HT format at the low to mid frequencies and we focus our efforts on this frequency range.

[Figure 4 about here.]

We visualize the mathematical decomposition depicted in Figure 3 using actual seismic data, shown in Figures 5, 6, and 7. As indicated previously, we first reshape our seismic data in the non-canonical matrix ordering, and decompose this matrix into a product of three matrices. From the matrix $\mathbf{U}_{\text{sx, rx}}$, we can consider each column as a vectorized matrix. Reshaping this vector into a matrix allows us to further decompose it in an SVD-like manner. The matrices $\mathbf{U}_{\text{sx}}$ and $\mathbf{U}_{\text{rx}}$, i.e., the quasi-left and right singular vectors, are kept constant across when decomposing each column of $\mathbf{U}_{\text{sx, rx}}$. As a result, the number of parameters needed in the HT tensor format is greatly reduced compared to the ambient tensor space. It is not straightforward to derive physical insight from these quasi-singular vectors, although they do tend to qualitatively behave as Green's functions. For a more thorough treatment of the HT format, as well as attendant software, we refer the interested reader to Da Silva and Herrmann (2015).

[Figure 5 about here.]

[Figure 6 about here.]

[Figure 7 about here.]

SEISMIC DATA COMPRESSION

When our data is fully sampled, we can use the method of Tobler (2012) to truncate the full volume to the HT tensor format. This method allows us to prescribe an error level and a maximum rank parameter to approximate the data. We outline the performance of this algorithm in Table 1 on a synthetically generated data set on the 3D Overthrust

model (Aminzadeh et al., 1997) with 50 x 50 sources and 396 x 396 receivers. As there is no unique notion of (minimal) rank for tensors, the truncation algorithm merely upper bounds the error by choosing appropriate intermediate ranks automatically. As the temporal frequency increases, so do the internal ranks of the tensor format, and thus lower-frequency data benefits from compression more than higher-frequency data. As noted previously, the canonical organization of the data performs much more poorly than the non-canonical permutation and the difference becomes more apparent at higher frequencies. We focus on using the non-canonical organization of the data for the remainder of this work.

[Table 1 about here.]

SEISMIC DATA INTERPOLATION

As discussed above, terrain restrictions or cost limitations almost always limit fully sampled data in realistic scenarios. In order to compensate for this missing data, we apply the algorithm described by Da Silva and Herrmann (2015) to reconstruct the full data volume by solving the optimization problem

$$\min_{\mathbf{x}} \|\mathcal{A}\phi(\mathbf{x}) - \mathbf{b}\|^2, \tag{3}$$

where $\mathbf{x}$ is the vectorized set of HT parameters $(\mathbf{U}_t, \mathbf{B}_t)_{t\in\mathcal{T}}$ defined previously, t indexes the nodes of the dimension tree $\mathcal{T}$, ϕ maps $\mathbf{x}$ to the fully-expanded tensor $\phi(\mathbf{x})$, as in the reverse process of Figure 3, $\mathcal{A}$ is the subsampling operator, and $\mathbf{b}$ is our subsampled data. This algorithm can interpolate each 4D monochromatic frequency slice quickly, as it does not compute SVDs on large matrices, and it can successfully recover seismic data volumes with a high level of randomly missing data. When our data has randomly missing entries, we use

these efficient algorithms to recover an estimate of the fully sampled data in compressed HT form. We refer the reader who is interested in the details of these algorithms to Da Silva and Herrmann (2015).

ON-THE-FLY EXTRACTION OF SHOT/RECEIVER GATHERS

Irrespective of our sampling regime, once we have a HT representation of our data volume, we can greatly reduce the computational and memory costs of working with our data. To make full use of the data directly in its compressed form, we present an approach to extract a shot (or receiver) gather at a given source location (i_x, i_y) directly from the compressed parameters. Here, we use Matlab colon notation $\mathbf{A}(i,:)$ to denote the extraction of the i^{th} row of the matrix $\mathbf{A}$, and similarly for column extraction. The common shot gather can be extracted by computing

$$\begin{aligned}
\mathbf{u}_{\text{sx,rx}} &= \mathbf{U}_{\text{sx}}(i_x,:) \times_1 \mathbf{U}_{\text{rx}} \times_2 \mathbf{B}_{\text{sx,rx}} \\
\mathbf{u}_{\text{sy,ry}} &= \mathbf{U}_{\text{sy}}(i_y,:) \times_1 \mathbf{U}_{\text{ry}} \times_2 \mathbf{B}_{\text{sy,ry}} \\
\mathbf{D}_{i_x,i_y} &= \mathbf{u}_{\text{sx,rx}} \times_1 \mathbf{u}_{\text{sy,ry}} \times_2 \mathbf{B}_{\text{sx,rx,sy,ry}}.
\end{aligned} \tag{4}$$

This algorithm follows the main construction of the parameters to full tensor mapping outlined in Grasedyck (2010), although specified to a single shot location. A pictorial representation of this algorithm is given in Figure 8. Most importantly, at no point do we need to form any intermediate quantities of the size of the full tensor and all the computations in Equation 4 can be implemented via multilinear products, outlined in Algorithm 1. This allows us to efficiently have query-based access to the data volume in its compressed form, which will be useful for the stochastic FWI approach detailed below. Note that common-receiver gather extraction can be implemented in an analogous way. The results of applying Algorithm 1 to a sample dataset generated from the BG Compass model at 6 Hz is shown in Figure 9.

Algorithm 1 Extracting a common shot gather from compressed Hierarchical Tucker parameters

Input: Indices of the source coordinates i_x and i_y, and dimension tree

1. Extract the row vector $\mathbf{u}_{\text{sx}}$ from the matrix $\mathbf{U}_{\text{sx}}(i_x, :)$

2. Multiply $\mathbf{B}_{\text{sx, rx}}$ along the k_{sx} dimension with the row vector $\mathbf{u}_{\text{sx}}$ (in the sense of multilinear product)

3. Multiply the matrix obtained from step 2 along the k_{rx} dimension (second dimension) with $\mathbf{U}_{\text{rx}}$, resulting in a matrix $\mathbf{U}_{\text{sx, rx}}$ of size $n_{\text{rx}} \times k_{(\text{sx, rx})}$

4. Repeat steps $1, 2, 3$ along the y coordinate to obtain the matrix $\mathbf{U}_{\text{sy,ry}}$ of size $n_{\text{ry}} \times k_{(\text{sy, ry})}$

5. The product $\mathbf{U}_{\text{sx, rx}}\mathbf{B}_{\text{sx,rx,sy,ry}}\mathbf{U}^T_{\text{sy, ry}}$ results in the final shot gather

[Figure 8 about here.]

[Figure 9 about here.]

CASE STUDY 1: STOCHASTIC FULL-WAVEFORM INVERSION

FWI is a non-linear data-fitting procedure that estimates a model of the subsurface given measurements made on the Earth's surface. Mathematically, we find a model $\mathbf{m}$ that generates predicted data that best agrees with our observed data in a least-squares sense, i.e.,

$$\min_{\mathbf{m}} \Phi(\mathbf{m}) = \frac{1}{2N_s} \sum_{i=1}^{N_s} \|\mathbf{P}_r \mathbf{H}(\mathbf{m})^{-1} \mathbf{q}_i - \mathbf{d}_i\|^2, \tag{5}$$

where $\mathbf{P}_r$ maps the computed wavefield from the subsurface to the receiver locations, $\mathbf{u}_i = \mathbf{H}(\mathbf{m})^{-1}\mathbf{q}_i$ is the wavefield corresponding to the $i-$th source $\mathbf{q}_i$, and $\mathbf{d}_i$ is the observed data, and we have N_s sources. Our model of the wave equation in this case is the constant-

density acoustic Helmholtz equation, although other models are possible. We consider inverting a single frequency at a time, without loss of generality. This optimization problem in Equation 5 is typically solved by using iterative algorithms with the model update

$$\mathbf{m}_{k+1} = \mathbf{m}_k + \alpha_k \mathbf{s}_k, \tag{6}$$

where $\mathbf{s}_k$ denotes the search direction and α_k the step size at each iteration. The simplest case is the steepest-descent method in which the search direction is chosen as the negative gradient of the objective function, i.e., $\mathbf{s}_k = -\nabla\Phi(\mathbf{m})$. Computing the full gradient $\nabla\Phi(\mathbf{m})$ is computationally daunting when N_s is large as we have to solve N_s PDEs at each iteration.

We follow the algorithmic developments of Da Silva and Herrmann (2016) and use a parallel stochastic optimization approach to improve the convergence with limited passes through the data. If our computational environment has p independent parallel processes with $p \ll N_s$, we partition the data into p disjoint subsets of size $N_s/p > 1$. Each node j has access to its own subset of data indexed by $I_j \subset \{1, \ldots, N_s\}$. At every outer iteration of our FWI algorithm, we choose a random subset $\tilde{I}_j \subset I_j$ at each node, where $|\tilde{I}_j| < |I_j|$ (in our experiments, we choose $|\tilde{I}_j| = 1$, so we choose a single shot per node). We use the Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method, described in Schmidt et al. (2009), to approximately solve the following subproblem

$$\begin{aligned} \min_{\mathbf{m}} \ \tilde{\Phi}(\mathbf{m}) :=& \frac{1}{2p} \sum_{j=1}^{p} \frac{1}{|\tilde{I}_j|} \sum_{k \in \tilde{I}_j} \|\mathbf{P}_r \mathbf{H}(\mathbf{m})^{-1} \mathbf{q}_k - \mathbf{d}_k\|^2 \\ & \text{such that } m_{LB} \leq \mathbf{m} \leq m_{UB}, \end{aligned} \tag{7}$$

where m_{LB} and m_{UB} are constant lower and upper bounds for the velocity (2000 m/s and 5000 m/s, in this example).

We repeat this procedure for T outer iterations, redrawing the shots we use in the subproblem at each iteration with replacement. Applying the L-BFGS method directly in a

stochastic context results in computing differences of gradients that correspond to different subsets of shots. Computing this difference is not representative of the full, non-stochastic Hessian, upon which the L-BFGS method relies, and therefore results in poor convergence. This approach, on the other hand, allows us to avoid this difficulty, as we use the L-BFGS method directly to solve the subproblem and postpone our redrawing of shots until the algorithm moves on to another subproblem. Other approaches to stochastic L-BFGS (Moritz et al., 2016; Gower et al., 2016) have nice theoretical properties but may be computationally expensive for FWI. We limit the number of L-BFGS iterations for the subproblem so that the number of PDEs used to solve the subproblem is equivalent to computing the objective and gradient with the entire dataset (i.e., computing an approximate solution to each subproblem is equivalent in cost to one full pass through the data). In this manner, we can drastically reduce the per-iteration cost of FWI while still respecting the utilization of our parallel resources effectively. A standard stochastic gradient algorithm, by comparison, would select a random subset of all shots to process at a given time, which would not respect the distribution of data among the workers and therefore require either costly data redistribution between nodes or result in some nodes remaining idle while others are working.

NUMERICAL EXAMPLES

We integrate our data-extraction approach with the aforementioned stochastic FWI method and perform inversion on a truncated portion of the BG Compass model. We select a 10 km x 10 km x 1.8 km central portion of the model which has 50 m x 50 m x 12 m grid spacing. We place a 49 x 49 grid of sources along the top of the model and a 196 x 196 grid of receivers in an ocean-bottom node setup, using a Ricker wavelet with peak frequency at 10 Hz. We use frequency-domain inversion code from Da Silva and Herrmann (2016), inverting a single

frequency at a time, from 3 Hz to 6 Hz in 0.25 Hz increments. As our maximum inverted frequency is quite low, we do not expect to recover a highly detailed model, but instead focus on the differences between the inversion with the original data and its compressed version. We limit the number of passes through the full data to $T = 3$, i.e., we compute three outer iterations. We run our FWI algorithm on a computational cluster with 100 computational nodes with 4 local workers each. This configuration results in a number of inner subproblem iterations equal to $\lceil \frac{50^2}{400} \rceil = 7$. Despite these limited number of passes through the data, our algorithm makes significant progress towards the true solution. Various 2D slices through the true and initial models are shown in Figure 10.

[Figure 10 about here.]

In the "full data" scenario, where our data has full source and receiver coverage, we compare the inversion results using the full data as well as the HT compressed data, which are shown in Figure 11. Despite our data being compressed by over 90%, the inversion of the compressed data is nearly identical to that of the fully sampled data, as one would expect from the high SNRs shown in Table 1. The relative model error is reduced from the initial 8.64% to approximately 5.9%, for both the fully sampled data and HT compressed data. The relative difference between the inverted models produced by these two datasets is 0.31%.

[Figure 11 about here.]

In the "missing data" scenario for our data, we remove data from our volume by randomly decimating 75% of the receiver coordinates. We consider two further inversion scenarios, one where we use the subsampled data directly in our FWI method and one where we interpolate the data via the method in Da Silva and Herrmann (2015) prior to inversion. In both

cases, the parameters and methods of the optimization problem are identical and described previously. The amount of total computation between these two cases is identical, i.e., the number of PDEs solved at each frequency and the frequencies chosen are the same, and the random seed for the shot subsampling is identical, so that each problem instance operates on identical subsets of sources. The workflows for all of these examples are the same and the only differing factor between these two scenarios is whether the data is used directly in its subsampled form or is interpolated prior to inversion.

There is still an open question in the literature as to whether one should use the subsampled data directly in inversion or use interpolation. Although we do not aim to provide a comprehensive answer to this question in this work, the results in Figure 12 would seem to indicate that there is a substantial benefit in interpolating the data volume prior to inversion. Compared to the results in Figure 11, the inverted models arising from interpolated data are visually similar to those generated from the fully sampled data. Using the subsampled data directly, on the other hand, results in an updated model that has only been marginally updated from the initial model.

[Figure 12 about here.]

CASE STUDY 2: EXTENDED-IMAGE VOLUME WITH COMPRESSED DATA VIA PROBING

Image gathers are a useful method to map reflection events present in prestack seismic data to their associated reflectors in the subsurface. The end result is a function of the original spatial coordinates and a set of redundant auxiliary coordinates, the so-called *extended image*. This approach is helpful when studying angle-dependent reflection coefficients and additionally

the failure of such gathers to focus can indicate errors in the large-scale background velocity model (Claerbout, 1970; Doherty and Claerbout, 1974; De Bruin et al., 1990; Biondi and Sava, 1999; Biondi and Symes, 2004; Sava and Vasconcelos, 2011; Koren and Ravve, 2011). The main computational costs associated with forming these image gathers result from having to compute the solution of a forward and adjoint wave equation for each shot and the subsequent cost of crosscorrelations. The computational costs scale linearly with the number of sources which, as the extended image itself is multi-dimensional, renders forming the full image volume computationally intractable even for 2D examples.

Traditional approaches to forming image gathers restrict the auxiliary coordinates in some fashion, such as using only surface or horizontal offsets, in order to mitigate some of the computational complexity. If computational costs were not an issue, our goal would be to have access to the information stored in the entire extended-image volume as a function of subsurface-offsets in all spatial coordinate directions. In this scenario, one could use this full-volume information to compute, for example, amplitude-versus-angle (AVA) or amplitude-versus-offset (AVO) methods derived from the linearized Zoeppritz equations (Aki and Richards, 2002).

In order to overcome these computational bottlenecks, van Leeuwen et al. (2016) proposed to *probe* the extended-image volume with particular test vector $\mathbf{w}$, rather than restrict it in some ad-hoc fashion such as to a horizontal or surface offset. This approach has the advantage of not artificially limiting the information present in the extended image for computational purposes. Moreover, the computational costs of forming explicit images of the subsurface grow linearly with the number of images formed, rather than with the number of sources and receivers in the traditional case. We outline this approach below and demonstrate how our data-extraction approach can be integrated to further reduce the memory costs of this

method.

In its discretized form, the extended image $\mathbf{E}$ (at a single frequency) is an $N \times N$ complex-valued matrix, where $N = n_z n_x$ in 2D or $N = n_x n_y n_z$ in 3D. Given a background model $\mathbf{m}_0$, $\mathbf{E}$ is formed as an outer product of the $N \times n_{\text{src}}$ matrices

$$\mathbf{E} = \mathbf{U}\mathbf{V}^H. \tag{8}$$

Here, $\mathbf{U}$ and $\mathbf{V}$ are the source and receiver wavefields, respectively, which obey the equations

$$\begin{aligned} \mathbf{H}(\mathbf{m}_0)\mathbf{U} &= \mathbf{P}_s^*\mathbf{Q} \\ \mathbf{H}(\mathbf{m}_0)^*\mathbf{V} &= \mathbf{P}_r^*\mathbf{D}. \end{aligned} \tag{9}$$

The operators $\mathbf{P}_s, \mathbf{P}_r$ map the computational domain to the source and receiver locations, respectively. The adjoint of these operators inject their inputs at their respective locations into the computational domain. The data volume $\mathbf{D}$ at the current frequency contains reflection data (i.e., each column is a shot gather after removal of the direct arrival) organized as an $n_{\text{rec}} \times n_{\text{src}}$ matrix. $\mathbf{Q}$ is an $n_{\text{src}} \times n_{\text{src}}$ matrix containing the source weights associated to each shot. The full extended image is merely a sum over the extended images computed at each frequency, which we omit for notational simplicity.

Rather than forming this volume explicitly, which would require the solution of $n_{\text{src}} n_{\text{rec}} n_{\text{freq}}$ PDEs, the work of van Leeuwen et al. (2016) introduced the notion of *matrix probing* in this context, which is to say selecting a column of $\mathbf{E}$ implicitly by multiplying it with a vector $\mathbf{w}$. The product $\mathbf{y} = \mathbf{E}\mathbf{w}$ can be computed as

$$\mathbf{y} = \mathbf{H}^{-1}\mathbf{P}_s^*\mathbf{Q}\mathbf{D}^H\mathbf{P}_r\mathbf{H}^{-1}\mathbf{w} \tag{10}$$

at the cost of two PDE solves. When $\mathbf{w} = [0, \ldots, 0, 1, 0, \ldots, 0]$ is an identity basis vector, $\mathbf{E}\mathbf{w}$ is a common-image-point at the location (x_k, y_k, z_k) associated to the index of the non-zero

entry. Other choices of probing vectors can be used for purposes other than imaging, such as using Gaussian vectors in the context of Wave Equation Migration Velocity Analysis (WEMVA).

From van Leeuwen et al. (2016), forming $\mathbf{Ew}$ involves computing the quantities in Equation 10, read from right to left. We can take advantage of keeping our data volume in compressed form throughout this computation by Algorithm (2), which involves the computation of the term $\mathbf{D}^H\mathbf{P}_r\mathbf{H}^{-1}\mathbf{w}$, which we write as $\mathbf{D}^H\mathbf{v}$. This algorithm is simply a matrix-vector multiplication with the explicit matrix row extraction replaced with Algorithm (1). The overhead introduced by this method is minimal but the memory savings are significant.

Algorithm 2 Forming the CIP gathers with compressed Hierarchical Tucker parameters

Input: $\mathbf{v} = \mathbf{P}_r\mathbf{H}^{-1}\mathbf{w}$

Output: $\mathbf{z} = \mathbf{D}^H\mathbf{v}$

For each source index $\mathbf{i} = (i_{\text{src}_x}, i_{\text{src}_y})$

1. Extract the common source gather from the data using Algorithm (1), resulting in $\mathbf{D}_i$.
2. Correlate $\overline{\mathbf{D}_i}$ and $\mathbf{v}$ to produce $\mathbf{z}_\text{i}$

NUMERICAL EXAMPLES

We showcase our data compression and extraction method integrated into the context of forming common-image-point (CIP) gathers. Our model is a 2.5 km $\times$ 2.5 km $\times$ 0.6 km subset of the BG Compass model with 25 m x 25 m x 6 m spacing, resulting in a model grid of size 101^3. We generate data using 15 frequencies ranging from 5 to 12 Hz with 0.5 Hz spacing and a Ricker wavelet with central frequency of 15 Hz. The sources are placed at the surface of the ocean with 75 m spacing and the receivers are located on the sea floor every

50 m, resulting in 1156 sources and 2601 receivers. We compare forming the CIP with the true data as well as the volume in which each monochromatic frequency slice is truncated in the HT format with 1% relative error. Figure 13 shows the 3D CIP using both data volumes and Figure 14 plots various 2D slices through the computed volume, extracted at (x, y, z) coordinates $(1250\text{ m}, 1250\text{ m}, 390\text{ m})$. Given the compressibility of seismic data in HT format, the plots using each data set are virtually indistinguishable and, using compressed data, the CIP gather has an SNR of 49.8 dB.

[Figure 13 about here.]

[Figure 14 about here.]

DISCUSSION

One of the key issues in low-rank compression and interpolation is that the original signal must be sufficiently *compressible*, i.e., well-approximated by a low-rank tensor. In practical terms, this requires that the singular values of the data, when matricized along the appropriate dimensions, must decay quickly in order for this low-rank approach to work. As shown in Da Silva and Herrmann (2013), Aravkin et al. (2014), Kumar et al. (2015), a seismic data tensor must be mapped to an appropriate *transform domain* as well as having *sufficient spatial sampling* in order for its singular values to decay sufficiently quickly. One of the challenges for low-rank methods in seismic data is that the effective ranks of the data matricizations increase with the temporal frequency of the data, rendering them less favourable to low-rank approximation. The examples considered in this work have therefore been restricted to a

relatively low frequency range. Potentially multidimensional windowing can alleviate this shortcoming (Kumar et al., 2013), but this remains to be seen in the 3D seismic case.

Given that the error level for the truncation approach is user-defined, we are able to produce inversion and image-gather results that differ only slightly from using the uncompressed data volume. In this framework, one also has the ability to trade off increased memory for increased accuracy, when such considerations are important. When the input data is noisy, the HT format can also reduce the imprint of the noise when the noise is high-amplitude yet spatially-sparse (Da Silva and Herrmann, 2014).

Working with the data directly in a compressed form has the potential to decouple the synchronous computational requirements of FWI in a large-scale parallel environment. Our approach offers the possibility of moving towards an asynchronous full-waveform paradigm, where each node has a local copy of the model and only synchronizes its copy periodically with its neighbours, while still having access to a full copy of the compressed data locally, as in Mokhtari et al. (2015).

CONCLUSION

Low-rank methods are an efficient approach to dealing with the curse of dimensionality. In addition to being simple to implement and efficient from a memory perspective, they offer an opportunity to compute quantities directly in their compressed form. Using the HT format allows us to drastically reduce the storage costs of seismic data while offering us the flexibility to extract shot gathers from the corresponding HT parameters. When we have fully sampled the data, we can use existing truncation techniques to compress our data into HT form. The subsequent inversion results using both fully sampled and compressed data

differ only slightly from each other, while the memory savings for the data are substantial. When the data has missing entries, we see a significant improvement in the inversion results from the HT interpolated data compared to merely using the subsampled data. In the image volume context, we were able to seamlessly integrate our approach into forming image gathers through matrix-probing and the results are again only marginally affected by the use of compressed data. This method has the potential to drastically reduce communication costs in large-scale distributed environments, which will be particularly relevant as seismic data grows from the petabyte to exabyte scales.

REFERENCES

Abubakar, A., M. Li, Y. Lin, and T. M. Habashy, 2012, Compressed implicit jacobian scheme for elastic full-waveform inversion: Geophysical Journal International, **189**, 1626–1634.

Aki, K., and P. G. Richards, 2002, Quantitative seismology: Freeman and Co. New York.

Aminzadeh, F., B. Jean, and T. Kunz, 1997, 3-d salt and overthrust models: Society of Exploration Geophysicists.

Aravkin, A., R. Kumar, H. Mansour, B. Recht, and F. J. Herrmann, 2014, Fast methods for denoising matrix completion formulations, with applications to robust seismic data interpolation: SIAM Journal on Scientific Computing, **36**, S237–S266.

Bekara, M., and M. Van der Baan, 2007, Local singular value decomposition for signal enhancement of seismic data: Geophysics, **72**, V59–V65.

Biondi, B., and P. Sava, 1999, Wave-equation migration velocity analysis, *in* Expanded Abstracts: SEG, 1723–1726.

Biondi, B., and W. W. Symes, 2004, Angle-domain common-image gathers for migration velocity analysis by wavefield-continuation imaging: Geophysics, **69**, 1283–1298.

Boutsidis, C., and D. Woodruff, 2017, Optimal cur matrix decompositions: SIAM Journal on Computing, **46**, 543–589.

Candes, E., L. Demanet, D. Donoho, and L. Ying, 2006, Fast discrete curvelet transforms: Multiscale Modeling & Simulation, **5**, 861–899.

Candès, E. J., and B. Recht, 2009, Exact matrix completion via convex optimization: Foundations of Computational Mathematics, **9**, 717–772.

Candès, E. J., and T. Tao, 2010, The power of convex relaxation: Near-optimal matrix completion: IEEE Transactions on Information Theory, **56**, 2053–2080.

Carroll, J. D., and J.-J. Chang, 1970, Analysis of individual differences in multidimensional

scaling via an n-way generalization of "eckart-young" decomposition: Psychometrika, **35**, 283–319.

Cheng, J., and M. D. Sacchi, 2015, Separation and reconstruction of simultaneous source data via iterative rank reduction: Geophysics, **80**, V57–V66.

Claerbout, J. F., 1970, Coarse grid calculations of waves in inhomogeneous media with application to delineation of complicated seismic structure: Geophysics, **35**, 407–418.

Curry, W., 2010, Interpolation with fourier-radial adaptive thresholding: Geophysics, **75**, WB95–WB102.

Da Silva, C., and F. J. Herrmann, 2013, Hierarchical Tucker tensor optimization - applications to 4D seismic data interpolation: Presented at the EAGE Annual Conference Proceedings.

———, 2014, Low-rank promoting transformations and tensor interpolation - applications to seismic data denoising: Presented at the EAGE Annual Conference Proceedings.

———, 2015, Optimization on the hierarchical tucker manifold – applications to tensor completion: Linear Algebra and its Applications, **481**, 131 – 173.

———, 2016, A unified 2d/3d software environment for large-scale time-harmonic full-waveform inversion, *in* Technical Program Expanded Abstracts: SEG, 1169–1173.

De Bruin, C., C. Wapenaar, and A. Berkhout, 1990, Angle-dependent reflectivity by means of prestack migration: Geophysics, **55**, 1223–1234.

De Lathauwer, L., B. De Moor, and J. Vandewalle, 2000, A multilinear singular value decomposition: SIAM Journal on Matrix Analysis and Applications, **21**, 1253–1278.

Demanet, L., 2006, Curvelets, wave atoms, and wave equations: PhD thesis, California Institute of Technology.

Doherty, S., and J. Claerbout, 1974, Velocity analysis based on the wave equation: Geophysics, **37**, 741–768.

Donoho, D. L., 2006, Compressed sensing: IEEE Transactions on information theory, **52**, 1289–1306.

Freire, S. L., and T. J. Ulrych, 1988, Application of singular value decomposition to vertical seismic profiling: Geophysics, **53**, 778–785.

Gower, R., D. Goldfarb, and P. Richtárik, 2016, Stochastic block bfgs: squeezing more curvature out of data: International Conference on Machine Learning, 1869–1878.

Grasedyck, L., 2010, Hierarchical singular value decomposition of tensors: SIAM Journal on Matrix Analysis and Applications, **31**, 2029–2054.

Hackbusch, W., and S. Kühn, 2009, A new scheme for the tensor representation: Journal of Fourier Analysis and Applications, **15**, 706–722.

Hennenfent, G., and F. J. Herrmann, 2006, Application of stable signal recovery to seismic data interpolation, *in* Technical Program Expanded Abstracts: SEG, 2797–2801.

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

Herrmann, F. J., D. Wang, G. Hennenfent, and P. P. Moghaddam, 2007, Curvelet-based seismic data processing: A multiscale and nonlinear approach: Geophysics, **73**, A1–A5.

Jumah, B., and F. J. Herrmann, 2014, Dimensionality-reduced estimation of primaries by sparse inversion: Geophysical Prospecting, **62**, 972–993.

Kabir, M. N., and D. Verschuur, 1995, Restoration of missing offsets by parabolic radon transform: Geophysical Prospecting, **43**, 347–368.

Koren, Z., and I. Ravve, 2011, Full-azimuth subsurface angle domain wavefield decomposition and imaging part i: Directional and reflection image gathers: Geophysics, **76**, S1–S13.

Kreimer, N., and M. D. Sacchi, 2012, A tensor higher-order singular value decomposition for prestack seismic data noise reduction and interpolation: Geophysics, **77**, V113–V122.

Kreimer, N., A. Stanton, and M. D. Sacchi, 2013, Tensor completion based on nuclear norm minimization for 5d seismic data reconstruction: Geophysics, **78**, V273–V284.

Kumar, R., H. Mansour, A. Y. Aravkin, and F. J. Herrmann, 2013, Reconstruction of seismic wavefields via low-rank matrix factorization in the hierarchical-separable matrix representation: SEG Technical Program Expanded Abstracts, 3628–3633.

Kumar, R., S. Sharan, H. Wason, and F. J. Herrmann, 2016, Time-jittered marine acquisition–-a rank-minimization approach for 5D source separation: SEG Technical Program Expanded Abstracts, 119–123.

Kumar, R., C. D. Silva, O. Akalin, A. Y. Aravkin, H. Mansour, B. Recht, and F. J. Herrmann, 2015, Efficient matrix completion for seismic data reconstruction: Geophysics, **80**, V97–V114.

Li, M., A. Abubakar, J. Liu, G. Pan, and T. M. Habashy, 2011, A compressed implicit jacobian scheme for 3d electromagnetic data inversion: GEOPHYSICS, **76**, F173–F183.

Ma, J., 2013, Three-dimensional irregular seismic data reconstruction via low-rank matrix completion: Geophysics, **78**, V181–V192.

Mokhtari, A., Q. Ling, and A. Ribeiro, 2015, An approximate newton method for distributed optimization: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2959–2963.

Moritz, P., R. Nishihara, and M. Jordan, 2016, A linearly-convergent stochastic l-bfgs algorithm: Artificial Intelligence and Statistics, 249–258.

Nazari Siahsar, M. A., S. Gholtashi, A. R. Kahoo, H. Marvi, and A. Ahmadifard, 2016, Sparse time-frequency representation for seismic noise reduction using low-rank and sparse decomposition: Geophysics, **81**, V117–V124.

Oropeza, V., and M. Sacchi, 2011, Simultaneous seismic data denoising and reconstruction

via multichannel singular spectrum analysis: Geophysics, **76**, V25–V32.

Recht, B., 2011, A simpler approach to matrix completion: Journal of Machine Learning Research, **12**, 3413–3430.

Sacchi, M., S. Kaplan, and M. Naghizadeh, 2009, FX gabor seismic data reconstruction: Presented at the EAGE Annual Conference Proceedings.

Sava, P., and I. Vasconcelos, 2011, Extended imaging conditions for wave-equation migration: Geophysical Prospecting, **59**, 35–55.

Schmidt, M. W., E. Van Den Berg, M. P. Friedlander, and K. P. Murphy, 2009, Optimizing costly functions with simple constraints: A limited-memory projected quasi-Newton algorithm: Presented at the 12th International Conference on Artificial Intelligence and Statistics.

Stork, C., 1992, Singular value decomposition of the velocity-reflector depth tradeoff, part 2: High-resolution analysis of a generic model: Geophysics, **57**, 933–943.

Tobler, C., 2012, Low-rank tensor methods for linear systems and eigenvalue problems: PhD thesis, ETH Zürich.

Trickett, S., L. Burroughs, and A. Milton, 2013, Interpolation using hankel tensor completion: Ratio, **1**, 16.

Trickett, S., L. Burroughs, A. Milton, L. Walton, R. Dack, et al., 2010, Rank-reduction-based trace interpolation: Presented at the 2010 SEG Annual Meeting, Society of Exploration Geophysicists.

van Leeuwen, T., R. Kumar, and F. J. Herrmann, 2016, Enabling affordable omnidirectional subsurface extended image volumes via probing: Geophysical Prospecting.

Villasenor, J. D., R. Ergas, and P. Donoho, 1996, Seismic data compression using high-dimensional wavelet transforms: Data Compression Conference, IEEE, 396–405.

Wang, J., M. Ng, and M. Perz, 2010, Seismic data interpolation by greedy local radon transform: Geophysics, **75**, WB225–WB234.

Ying, L., L. Demanet, and E. Candes, 2005, 3D discrete curvelet transform: Optics & Photonics 2005, International Society for Optics and Photonics, 591413–591413.

LIST OF FIGURES

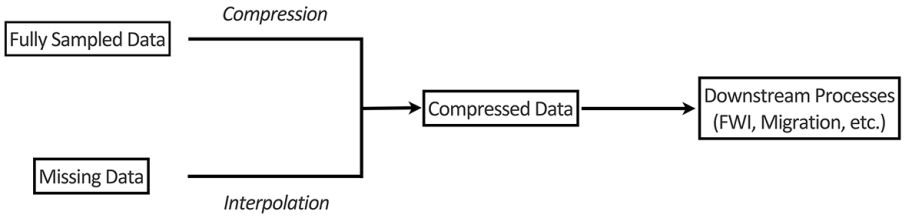


Figure 1: Workflow proposed in this article

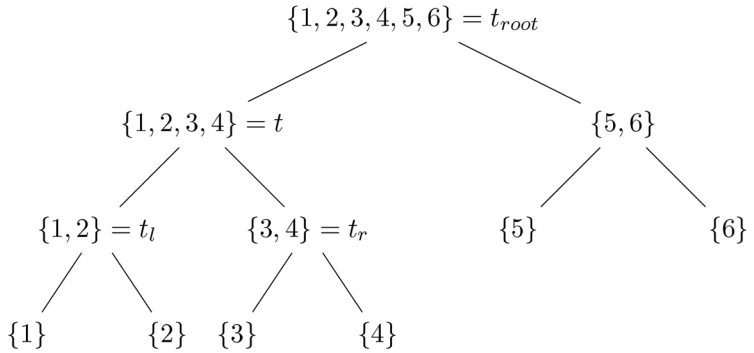


Figure 2: A dimension tree for $\{1, 2, 3, 4, 5, 6\}$, from Da Silva and Herrmann (2015).

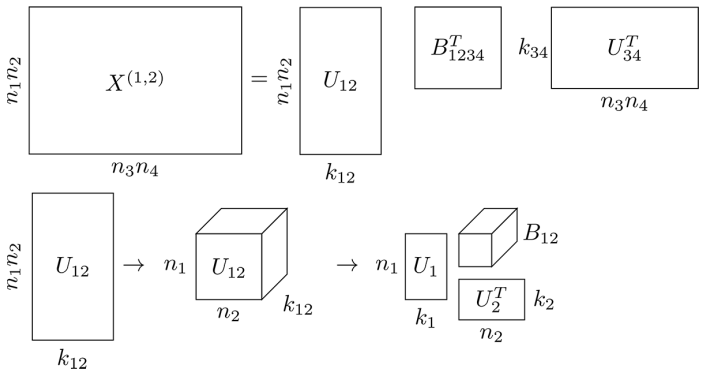


Figure 3: Hierarchical Tucker format for a 4D tensor $\mathbf{X} \in \mathbb{C}^{n_1 \times n_2 \times n_3 \times n_4}$. Image from Da Silva and Herrmann (2013) .

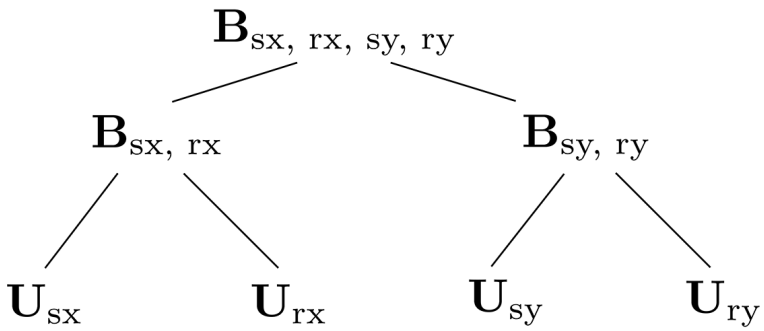


Figure 4: Non-canonical dimension tree for the HT format applied to each seismic 4D monochromatic slice. Shown at each node are the quantities (matrices at the leaves + root, 3-dimensional tensors at interior nodes) necessary for constructing the full tensor.

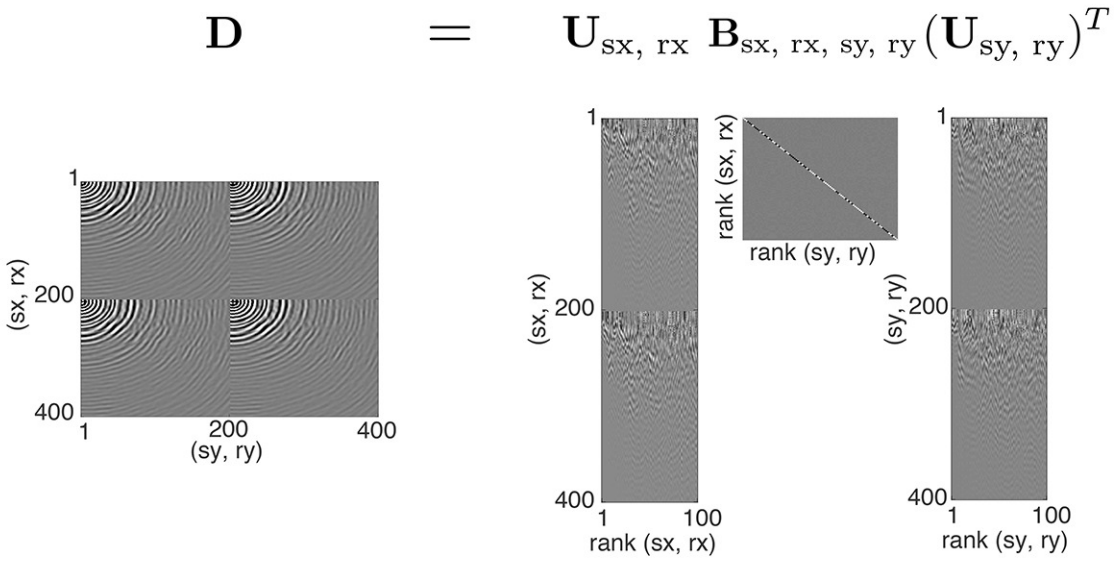


Figure 5: Visualizing the HT format on seismic data. Only a subset of the full matricized tensor is shown, for visibility.

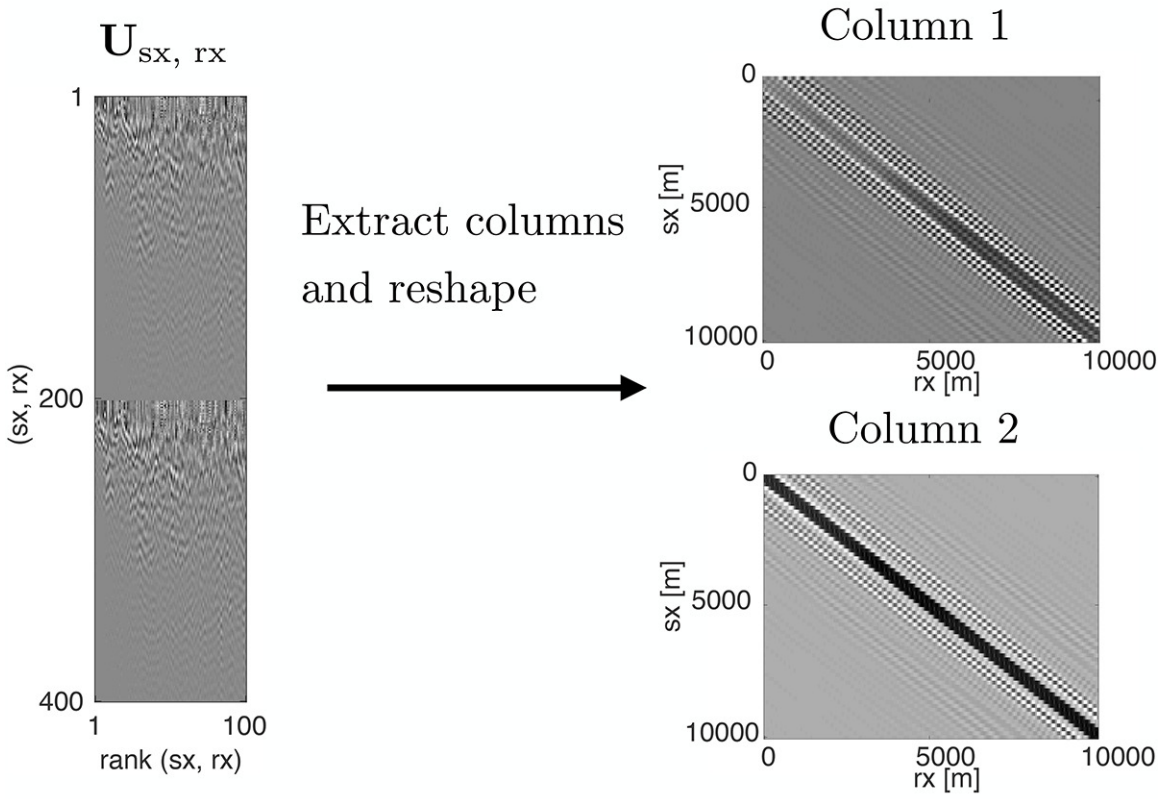


Figure 6: Visualizing columns of the intermediate matrices of the HT format. Each column can be reshaped into a matrix with dimensions $n_{\text{rx}} \times n_{\text{sx}}$. Note that the matrix $\mathbf{U}_{\text{sx, rx}}$ is shown with the joint (sx,rx) coordinate, which has no physical units as it is a linearized index, and the extracted columns are displayed in units of metres. Only a subset of the full matricized tensor is shown, for visibility.

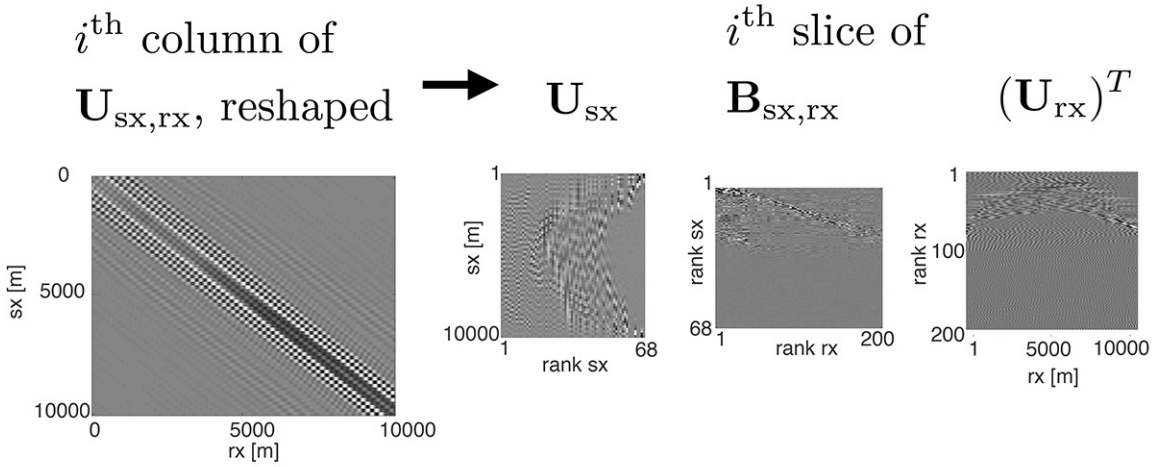
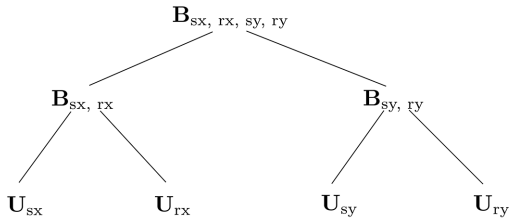
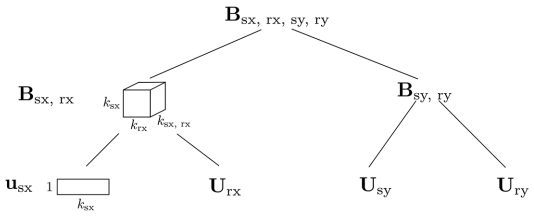


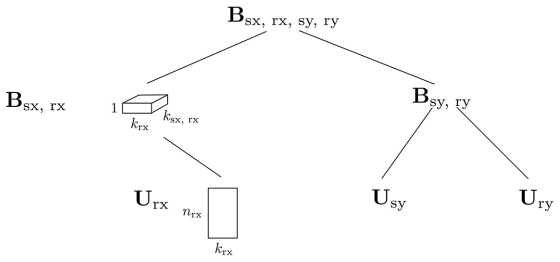
Figure 7: Visualizing the decomposition of the columns from Figure 6. The leaf matrices $\mathbf{U}_{\text{sx}}, \mathbf{U}_{\text{rx}}$ remain constant across the columns of $\mathbf{U}_{\text{sx,rx}}$ while the intermediate matrix $\mathbf{B}_{\text{sx,rx}}$ is allowed to vary for each column. Only a subset of the full matricized tensor is shown, for visibility.



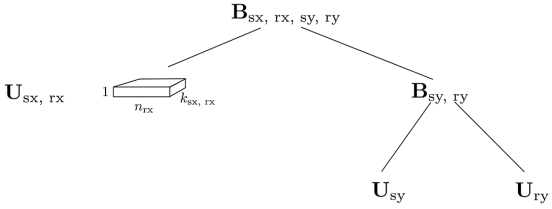
Initial parameters



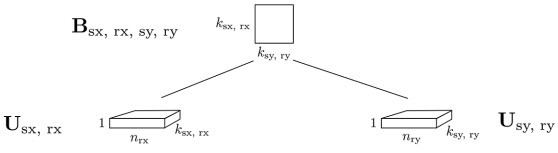
Step 1



Step 2

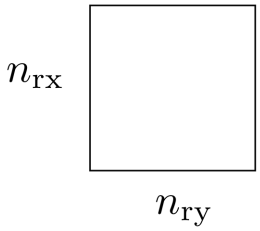


Step 3



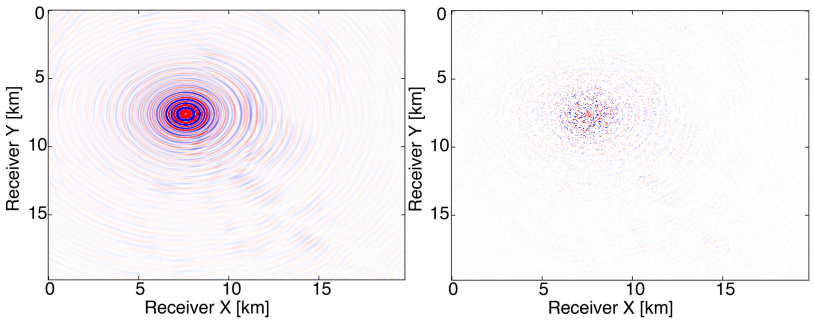
Step 4

$$\mathbf{U}_{\text{sx, rx}}\mathbf{B}_{\text{sx, rx, sy, ry}}(\mathbf{U}_{\text{sy, ry}})^T$$



Step 5

Figure 8: A pictorial depiction shot extraction of Algorithm (1)



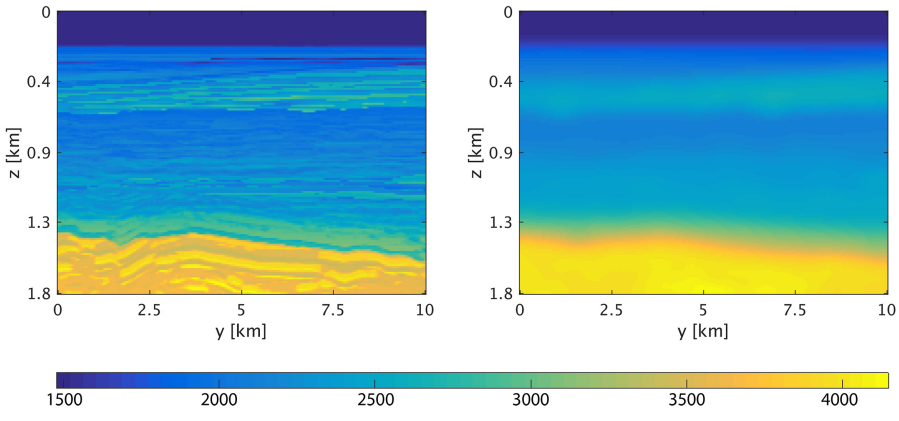
(a) True data

(b) Missing 80% data

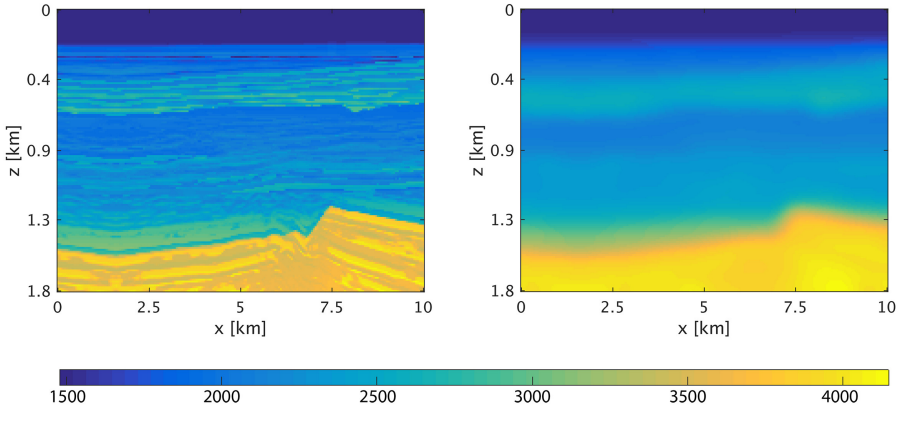
(c) Extracted from compressed, recovered data

(d) Difference

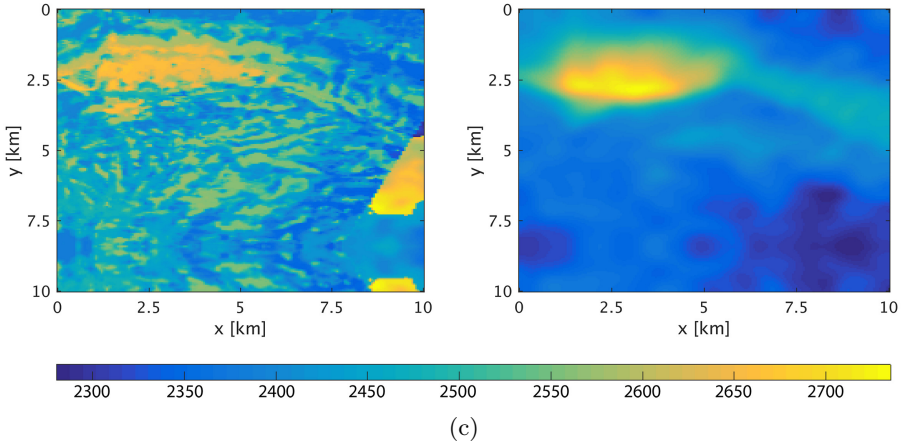
Figure 9: Extracted shot from the Overthrust data at 6Hz, full data vs compressed data with Algorithm (1) after HT interpolation



(a)

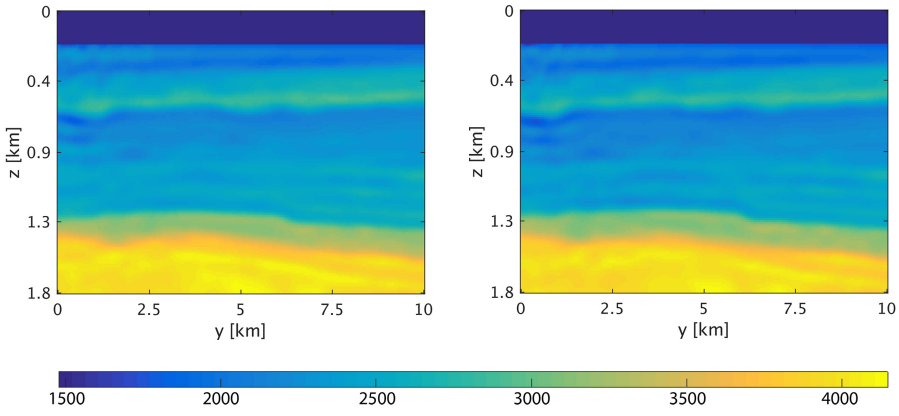


(b)

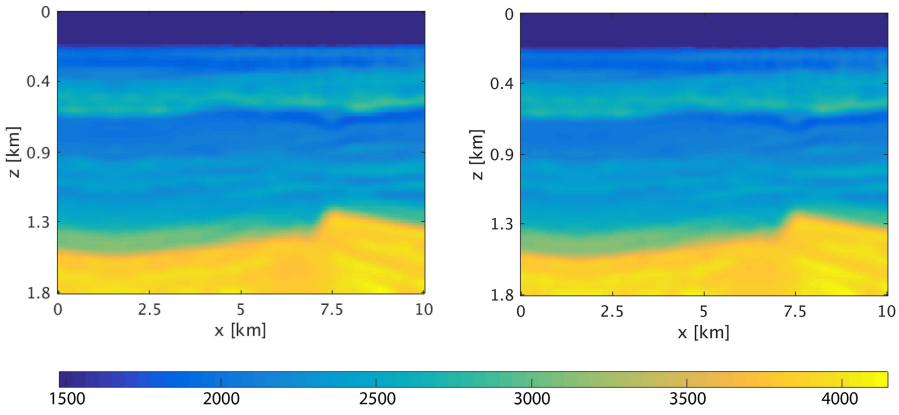


(c)

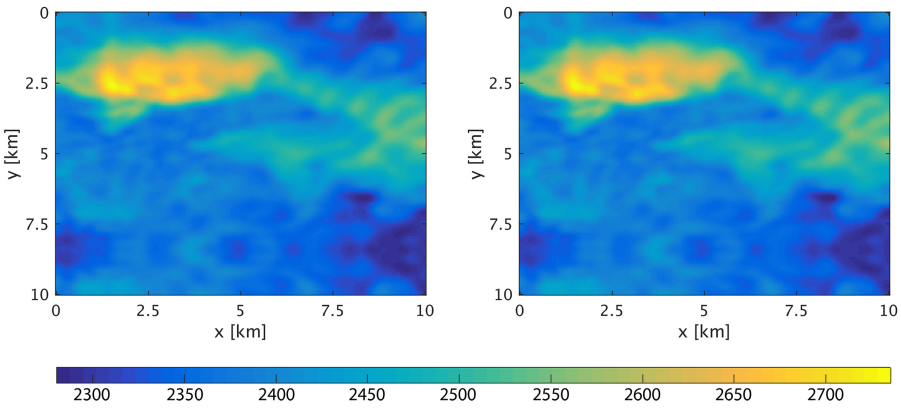
Figure 10: 2D slices of the true model (left column) and the initial model (right column), along the $x = 4900m$ (top row), $y = 5650m$ (middle row) and $z = 1200m$ (bottom row).



(a)



(b)



(c)

Figure 11: 2D slices of the inverted model with full data (left column) and compressed data (right column), along the $x = 4900m$ (top row), $y = 5650m$ (middle row) and $z = 1200m$ (bottom row).

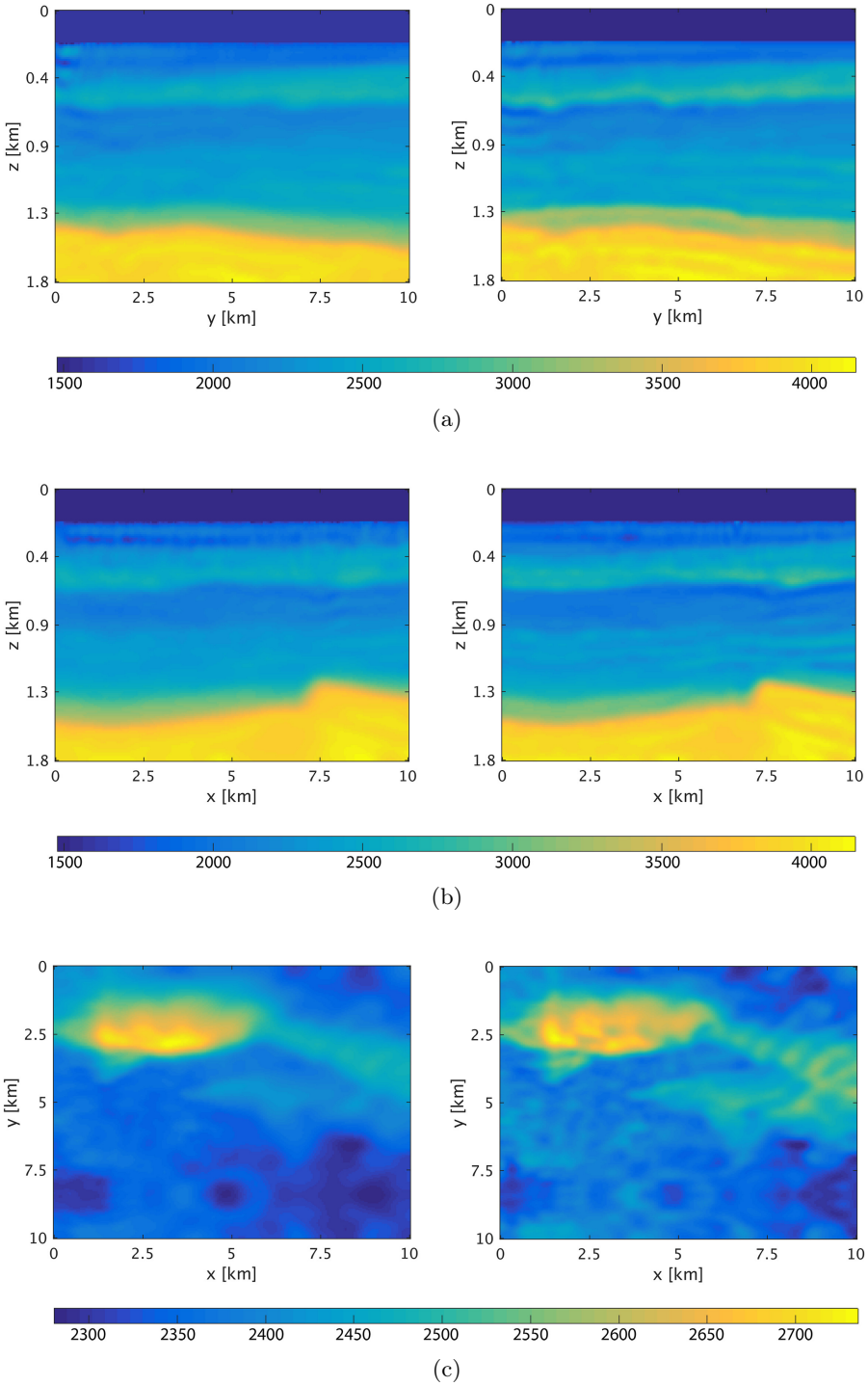
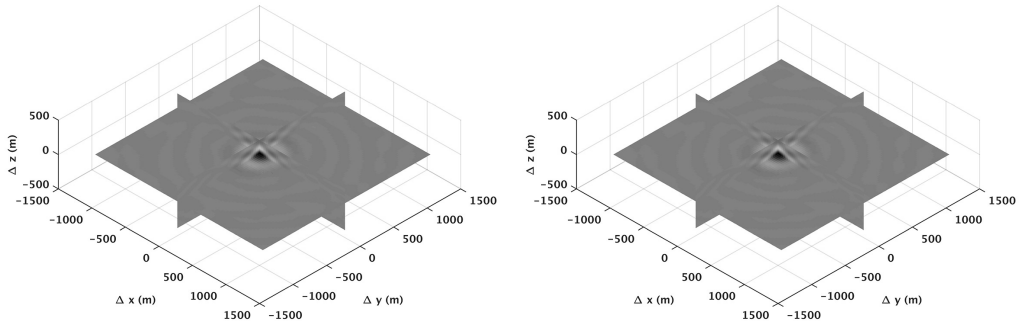


Figure 12: 2D slices of the inverted model with subsampled data (left column) and interpolated data (right column), along the $x = 4900m$ (top row), $y = 5650m$ (middle row) and $z = 1200m$ (bottom row).



(a) with fully sampled data

(b) with compressed HT parameters

Figure 13: A full CIP gather extracted at location $(x, y, z) = (1250m, 1250m, 390m)$.

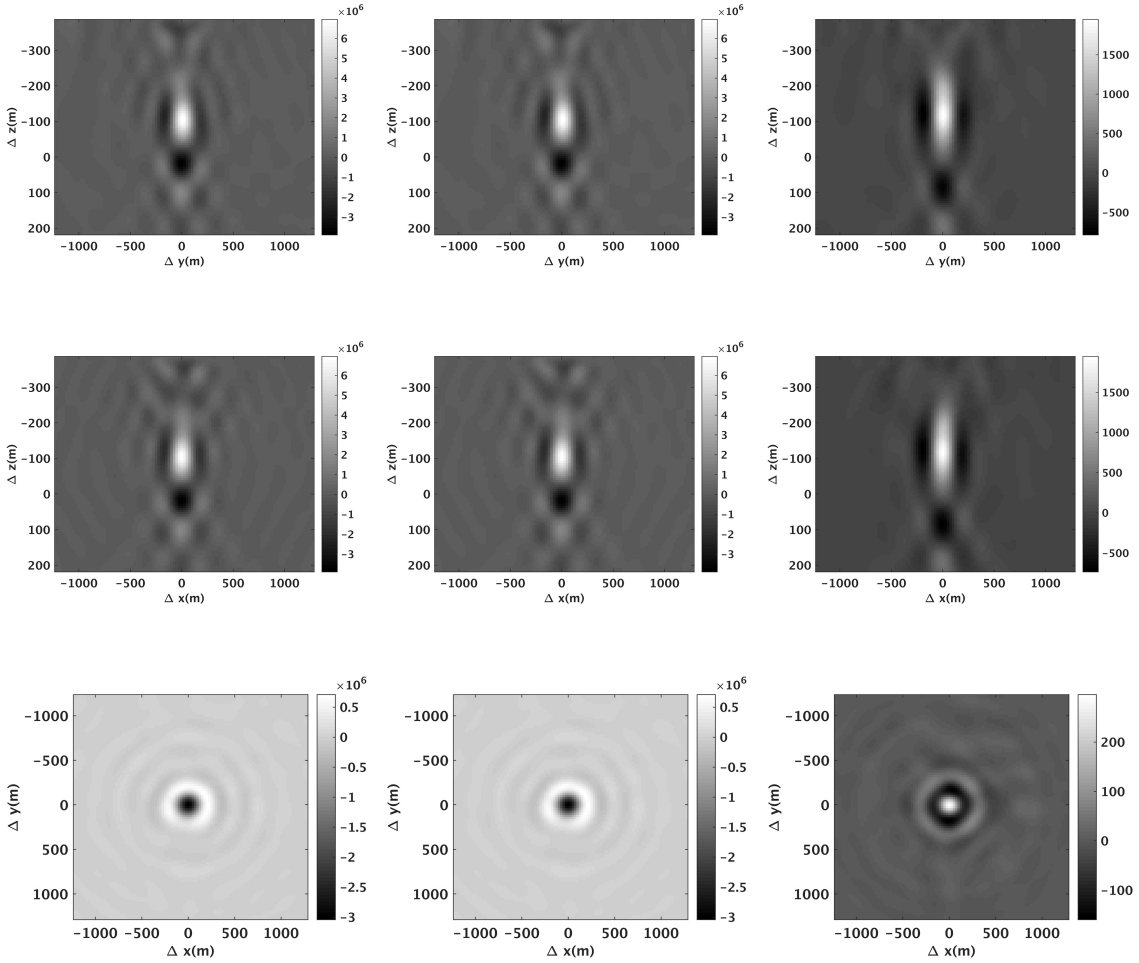


Figure 14: 2D slices extracted from the CIP gathers in Figure 13. Top and middle rows are along the horizontal offset directions, and bottom row is along vertical offset direction. First column is computed using the original data and the second column uses HT compressed data. Differences in the third column are displayed on a colorbar 100x smaller than the results.

LIST OF TABLES

	Frequency (Hz)	Parameter size	Compression Ratio	SNR
Non-canonical	3	71 MB	98.8%	42.8 dB
Canonical	3	501 MB	91.6%	42.9 dB
Non-canonical	6	421 MB	92.9%	43.0 dB
Canonical	6	1194 MB	79.9%	43.1 dB

Table 1: Compression rates of the HT truncation method at different frequencies, in different data organizations. The original frequency slice is approximately 5.8GB in size.