Compressive least-squares migration with on-the-fly Fourier transforms

Philipp A. Witte*, **Mathias Louboutin***, **Fabio Luporini**†, **Gerard J. Gorman**†
and Felix J. Herrmann*

**Georgia Institute of Technology*
School of Computational Science and Engineering
Coda building, 756 West Peachtree Street NW,
Atlanta, GA, 30308, U.S.A.

†*Imperial College London,*
Department of Earth Science & Engineering
Royal School of Mines, Prince Consort Rd, Kensington
London, SW7 2BP, U.K.

(September 24, 2019)

GEO-2018-0490.R2

Running head: **Compressive LSRTM with on-the-fly DFTs**

ABSTRACT

Least-squares reverse-time migration is a powerful approach for true amplitude seismic imaging of complex geological structures, but the successful application of this method is currently hindered by its enormous computational cost, as well as high memory requirements for computing the gradient of the objective function. We tackle these problems by introducing an algorithm for low-cost sparsity-promoting least-squares migration using on-the-fly Fourier transforms. We formulate the least-squares migration objective function in the frequency domain and compute gradients for randomized subsets of shot records and frequencies, thus significantly reducing data movement and the number of overall wave equations solves. By using on-the-fly Fourier transforms, we can compute an arbitrary number of monochromatic frequency-domain wavefields with a time-domain modeling code, instead of having to solve individual Helmholtz equations for each frequency, which becomes computationally infeasible when moving to high frequencies. Our numerical examples demonstrate that compressive imaging with on-the-fly Fourier transforms provides a fast and memory-efficient alternative to time-domain imaging with optimal checkpointing, whose memory requirements for a fixed background model and source wavelet is independent of the number of time steps. Instead, memory and additional computational cost grow with the number of frequencies and determine the amount of subsampling artifacts and crosstalk. In contrast to optimal checkpointing, this offers the possibility to trade both memory and computational cost for image quality or a larger number of iterations and is advantageous in new computing environments such as the cloud, where compute is often cheaper than memory and data movement.

INTRODUCTION

Reverse-time migration (RTM) is an increasingly popular wave-equation based seismic imaging algorithm that corresponds to applying the adjoint of the Born scattering operator to observed reflection data (Baysal et al., 1983; Whitmore, 1983). Without extensive preconditioning, applying the adjoint operator leads to an image with incorrect amplitudes, imprints of the source wavelet, finite apertures and therefore blurred reflectors. To overcome these issues and invert the Born scattering operator, imaging can be formulated as a linear least-squares optimization problem, in which the mismatch between observed and modeled data is minimized in a least-squares sense. Least-squares migration was introduced for ray-based imaging methods first (Lambaré et al., 1992; Schuster, 1993; Nemeth et al., 1999) and later extended to wave-equation based imaging (LS-RTM) (Valenciano, 2008; Tang and Biondi, 2009; Dong et al., 2012; Zeng et al., 2014).

The successful deployment of LS-RTM in practice is currently hampered by two distinct computational challenges. First of all, conventional LS-RTM requires the migration/demigration of all shot records in each iteration of gradient-based optimization algorithms, making this approach prohibitively expensive for large-scale data sets with thousands of individual shot records. To save computational resources, shots can therefore be subsampled or combined into supergathers/simultaneous shots, which avoids having to treat every shot separately in each iteration (Tang and Biondi, 2009; Herrmann, 2010; van Leeuwen et al., 2011; Dai et al., 2011, 2012, 2013; Liu, 2013). The resulting LS-RTM formulations can then be solved using stochastic optimization methods, such as stochastic gradient descent or variants of the stochastic conjugate gradient method (e.g. Huang and Zhou, 2014; Li et al., 2018). Since the migration of simultaneous shot records leads to crosstalk in the seismic image, the resulting artifacts need be addressed by additional constraints such as smoothing constraints (Chen et al., 2015) or transform-domain sparsity (Herrmann and Li, 2012; van Leeuwen et al., 2011; Lu et al., 2015; Tu and Herrmann, 2015). Common transforms that lead to sparsity of seismic images include the wavelet, seislet or curvelet transforms (Candès et al., 2006a; Herrmann et al., 2008; Fomel and Liu, 2010).

The second challenge of LS-RTM are the large requirements of data movement and fast memory access for computing the gradient of the objective function with the adjoint-state method (Tarantola, 1984; Plessix, 2006). The gradient for one shot record is computed by solving an adjoint wave equation, with the data residual between the predicted and observed data as the adjoint source, and requires access to the forward wavefields in reverse order. The forward wavefields are obtained by forward propagating the seismic source for the respective shot record, but they are typically too large to store in memory. To overcome this issue, a common strategy is to either write compressed wavefields to disk, or to store only a small subset of uncompressed wavefields, while the in-between wavefields are recomputed from checkpoints (Griewank and Walther, 2000; Symes, 2007) or the model boundary (McMechan, 1983). These approaches therefore offer a possible trade-off between memory usage and computational cost, as more wavefields need to be recomputed for a smaller number of checkpoints. Further alternatives for circumventing the storage of time-domain wavefields are discussed in Nguyen and McMechan (2015). Storing or recomputing wavefields is especially expensive for seismic imaging, since it is typically carried out at higher frequencies than full-waveform inversion (FWI) and thus involves substantially larger wavefields and models due to small grid spacings.

An alternative to recomputing time-domain wavefields from checkpoints, is to use time-to-frequency conversions to extract monochromatic frequency-domain wavefields from a time-stepping loop and to compute gradients for a small subset of frequencies. This approach circumvents the problem of having to store or recompute wavefields for a large number of time steps, as frequency-domain gradients are computed individually for one frequency at a time. Modeling frequency-domain wavefields with a time-domain modeling code is a well established approach in scientific computing and several algorithms exist to perform time-to-frequency conversions, including discrete on-the-fly Fourier transforms (DFTs) or linear equation methods (refer to Furse, 1998, for an overview). A conversion method in the context of seismic modeling using phase-sensitive detections (PSDs) is presented in Nihei and Li (2007). Extracting single frequency-domain wavefields from a time-stepping loop is even possible without any conversion at all and can be obtained by simply propagating a monochromatic source function to a steady state (Watanabe, 2015).

Time-to-frequency conversion methods enjoy great popularity in the context of full-waveform inversion (Sirgue et al., 2010; Etienne et al., 2010; Kim et al., 2013; Xu and McMechan, 2014; Ha et al., 2015), as it is generally desirable to carry out the inversion for single or few frequencies at a time (Bunks et al., 1995; Sirgue and Pratt, 2004). For seismic imaging on the other hand, the goal is to obtain a high-definition image with a broad frequency band, making it typically necessary to compute gradients for a large-number of evenly-spaced frequencies. Hence, we present a workflow for least-squares RTM using small subsets of randomly selected frequencies and shot records, with sparsity promotion to address the subsampling related imaging artifacts. In contrast to earlier works by Herrmann and Li (2012) and Herrmann et al. (2015), we use on-the-fly Fourier tranforms to compute gradients in the frequency domain with a highly optimized time-domain modeling code (Lange et al., 2016; Louboutin et al., 2017). On-the-fly DFTs not only allow us to compute an arbitrary number of frequencies in a single time-stepping loop, but also to scale the inversion to high frequencies, without solving large-scale 2D, and in particular 3D, Helmholtz equations. In our numerical examples, we demonstrate that compressive imaging with sparsity promotion (SPLS-RTM) and on-the-fly DFTs yields images of similar quality as time-domain LS-RTM, but without having to store or recompute time-domain wavefields and with a significantly reduced number of wave equation solves, using as few as two passes through the data. In the discussion, we analyze the asymptotic behaviour of memory requirements and computational cost for imaging with on-the-fly DFTs and compare it to optimal checkpointing. Thus, the contribution of this work is the formulation of frequency domain LS-RTM with a time-domain modeling operator and on-the-fly Fourier transforms using shot and frequency subsampling to overcome the prohibitively high cost of least squares RTM. The shot and frequency-subsampling effectively turn LS-RTM into an underdetermined compressed sensing problem, with computational flexibility regarding batch sizes and number of iterations, which can be customized according to the available computational resources. Furthermore, this paper introduces a forward-adjoint pair for imaging the impedance in both the time and frequency domain (with or without on-the-fly DFTs) and presents a quantitative and qualitative comparison of time-domain LS-RTM and frequency-domain LS-RTM with on-the-fly DFTs.

FREQUENCY-DOMAIN LEAST-SQUARES MIGRATION WITH TIME-DOMAIN MODELING

To circumvent the problem of having to store time-domain wavefields for computing the adjoint-state gradient, we formulate the least-squares reverse-time migration objective function in the frequency domain, using the frequency-domain linearized Born scattering operator $\mathbf{J}$:

$$\underset{\delta\mathbf{m}}{\text{minimize}} \quad \sum_{j=1}^{n_s}\sum_{k=1}^{n_f} \frac{1}{2}\left\| \mathbf{J}(\mathbf{m}_0, \bar{q}_{jk})\, \delta\mathbf{m} - \bar{\mathbf{d}}_{jk}^{\text{obs}} \right\|_2^2. \tag{1}$$

The vector $\mathbf{m}_0$ denotes the vectorized migration velocity model in squared slowness and $\delta\mathbf{m}$ is the unknown model perturbation (i.e. the seismic image). The vector $\bar{\mathbf{d}}_{jk}^{\text{obs}}$ is the observed reflection data in the frequency-domain (denoted by bars) of the j^{th} source location and the k^{th} frequency and $\bar{q}_{jk}$ is the complex-valued monochromatic source. The objective function is computed as the sum over all n_s source positions and n_f temporal frequencies. To model the predicted linearized data in the Fourier domain, we have to compute the action of the linearized Born modeling operator on the current image, $\bar{\mathbf{d}}_{jk}^{\text{pred}} = \mathbf{J}(\mathbf{m}_0, \bar{q}_{jk})\, \delta\mathbf{m}$, which corresponds to solving:

$$\bar{\mathbf{d}}_{jk}^{\text{pred}} = -\mathbf{P}_r \mathbf{H}(\mathbf{m}_0)^{-1} \text{diag}\left[\frac{\partial \mathbf{H}(\mathbf{m}_0)}{\partial \mathbf{m}} \mathbf{H}(\mathbf{m}_0)^{-1} \mathbf{p}_s^* \bar{q}_{jk} \right] \delta\mathbf{m}, \tag{2}$$

where $\mathbf{H}(\mathbf{m}_0)$ is the frequency-domain modeling operator and $\mathbf{P}_r$ is a projection operator that restricts the wavefield to the receiver locations. Accordingly, the vector $\mathbf{p}_s^*$ is the source injection operator, which is a column vector of zeros with value one at the source location. The asterisk denotes the adjoint of complex vectors and matrices, also known as the Hermitian adjoint.

While formulating the LS-RTM objective in the frequency domain avoids storing the history of the time-domain wavefields, it in principle requires inverting the Helmholtz matrix for modeling observed data and computing gradients. To avoid inverting large-scale Helmholtz systems, which are known to be ill-conditioned for large-scale systems with high frequencies, we instead compute the frequency-domain data with a time-domain modeling code (Furse, 1998; Nihei and Li, 2007; Sirgue et al., 2010). This combines the best of both worlds, as we can use a highly optimized time-domain modeling code for the PDE solves, while computing gradients memory efficiently in the frequency domain, where gradients are separable over frequencies. This means it is possible to compute the gradient for single frequencies at a time, as an element-wise product of the corresponding forward and adjoint monochromatic wavefields. In principal, modeling the frequency response of the linearized modeling operator involves computing the inverse DFT of the source wavelet $\bar{q}_{jk}$, solving the linearized wave equation in the time domain and then performing a time-to-frequency conversion of the single scattered wavefields for all frequencies (i.e. using a DFT). This is followed by extracting the k^{th} frequency through a frequency restriction operator $\mathbf{R}_k$ and restricting the wavefield to the receiver locations (through a receiver projection operator $\mathbf{P}_r$). Overall, we have:

$$\bar{\mathbf{d}}_{jk}^{\text{pred}} = -\mathbf{P}_r \mathbf{R}_k \mathbf{F} \mathbf{A}(\mathbf{m}_0)^{-1} \text{diag}\left[\frac{\partial \mathbf{A}(\mathbf{m}_0)}{\partial \mathbf{m}} \mathbf{A}(\mathbf{m}_0)^{-1} \mathbf{F}^* \mathbf{R}_k^* \mathbf{p}_s^* \bar{q}_{jk} \right] \delta\mathbf{m}, \tag{3}$$

where $\mathbf{A}(\mathbf{m})$ is the discretized time-domain wave equation and $\mathbf{F}$ is the discrete Fourier matrix. As mentioned, $\mathbf{R}_k$ is a restriction operator that extracts the k^{th} frequency of the wavefield, while its adjoint zero-padds a frequency wavefield along the frequency axis, so that we can compute its inverse temporal Fourier transform. However, in the actual implementation of equation 3, we combine the time-to-frequency conversion and the extraction of one or multiple frequencies into a single step, by performing on-the-fly DFTs for the respective frequencies, instead of an explicit DFT of all frequencies. Accordingly, we never explicitly zero-padd the source wavelet to perform an inverse DFT, but simply inject the time-domain wavelet.

To obtain the gradient of the frequency-domain LS-RTM objective function, we have to compute the action of the complex conjugate linearized modelig operator on the data residual, i.e.; $\bar{\mathbf{g}}_{jk} = \mathbf{J}(\mathbf{m}_0, \bar{q}_{jk})^* (\bar{\mathbf{d}}_{jk}^{\text{pred}} - \bar{\mathbf{d}}_{jk}^{\text{obs}})$. The expression for the gradient can be derived by taking the conjugate transpose of equation 2 and boils down to calculating the pointwise product of the (complex) forward and adjoint wavefields $\bar{\mathbf{u}}_{jk}$ and $\bar{\mathbf{v}}_{jk}$ (Pratt, 1999):

$$\bar{\mathbf{g}}_{jk} = -\mathrm{Re}\Big[\mathrm{diag}\big(\omega_k^2 \bar{\mathbf{u}}_{jk}\big)^* \bar{\mathbf{v}}_{jk}\Big]. \tag{4}$$

The scalar ω_k^2 is the squared angular frequency $\omega_k = 2\pi f_k$ and Re denotes the real part of the gradient. Once again, we do not compute the forward and adjoint frequency-domain wavefields by inverting the Helmholtz equation, but by solving time-domain wave equations, followed by time-to-frequency conversions for the respective frequencies. In an analogous manner to equation 3, we obtain the forward wavefield $\bar{\mathbf{u}}_{jk}$ for the j^{th} source location and k^{th} frequency by solving:

$$\bar{\mathbf{u}}_{jk} = \mathbf{R}_k \mathbf{F} \mathbf{A}(\mathbf{m}_0)^{-1} \mathbf{F}^* \mathbf{R}_k^* \mathbf{p}_s^* \bar{q}_{jk}. \tag{5}$$

The adjoint wavefield is obtained accordingly, by solving an adjoint time-domain wave equation with the data residual as the adjoint source. In general, this is different from time reversing the data residual and using the forward modeling operator, as it involves inverting the correct adjoint of the forward modeling operator:

$$\bar{\mathbf{v}}_{jk} = \mathbf{R}_k \mathbf{F} \mathbf{A}(\mathbf{m}_0)^{-*} \mathbf{F}^* \mathbf{R}_k^* \mathbf{P}_r^* (\bar{\mathbf{d}}_{jk}^{\mathrm{pred}} - \bar{\mathbf{d}}_{jk}^{\mathrm{obs}}), \tag{6}$$

with $\mathbf{A}(\mathbf{m}_0)^{-*}$ being the solution of the adjoint time-domain wave equation. In summary, we have derived an expression for computing the predicted linearized data in the frequency-domain using a time-domain modeling code, as well as corresponding expressions for the gradient of the LS-RTM objective function. Thus, these quantities can be computed with highly-optimized time-domain modeling codes instead of Helmholtz solvers, whose success heavily rely on the underlying preconditioners and linear solvers. However, the analytical expressions in this section contain explicit Fourier transforms of the time-domain wavefields that require access to their full time history in memory, which is what we wanted to avoid in the first place. Luckily, we can avoid having to explicitly compute discrete Fourier transforms of the time-domain wavefields, by computing the DFTs *on the fly*.

COMPUTING ON-THE-FLY FOURIER TRANSFORMS

To avoid saving the full time-dependent wavefield in memory and taking its Fourier transform after modeling, we compute Fourier domain wavefields for a given frequency f_k during the forward or reverse time loops on the fly. In the context of full-waveform inversion, this approach has been introduced by Sirgue et al. (2010) and has since then appeared in a number of publications related to FWI (e.g. Etienne et al., 2010; Kim et al., 2013; Xu and McMechan, 2014; Ha et al., 2015). The on-the-fly Fourier transforms correspond to computing the frequency-domain wavefields as a running sum over the current time-domain wavefield $\mathbf{u}_i$ of the i^{th} time step within a time modeling loop, multiplied with a complex exponential (Furse, 1998). To simplify the implementation and avoid complex arrays, we individually compute the real and imaginary part of the frequency domain wavefields. The on-the-fly Fourier transform of the forward wavefields is then given by:

$$\begin{aligned}\bar{\mathbf{u}}_{jk}^{\text{real}} &= \sum_{i=1}^{n_t} \cos(2\pi f_k i \Delta t)\mathbf{u}_i,\\ \bar{\mathbf{u}}_{jk}^{\text{imag}} &= -\sum_{i=1}^{n_t} \sin(2\pi f_k i \Delta t)\mathbf{u}_i.\end{aligned} \tag{7}$$

This expression can be fairly easily incorporated into an existing time-domain modeling code and only involves initializing the two frequency-domain wavefields with zeros and adding the current time-domain wavefield multiplied with the sine and cosine terms during each time step. To obtain the linearized data in the frequency-domain (equation 3), we technically have to perform the on-the-fly DFT on the linearized (single-scattered) wavefields, rather than on the source wavefields. Alternatively, it is possible to model time-domain shot records and perform the frequency conversion after modeling, since time-domain shot records themselves are not as big as time-domain wavefields and can generally be stored in memory. In the numerical examples section, we demonstrate, that this step is actually not necessary for LS-RTM, as we can simply use full time-domain sources and shot records (data residuals) as forward and adjoint sources. This means, we only use on-the-fly DFTs to compute wavefields for the gradient in the frequency-domain, but we work with time-domain data and sources.

The adjoint frequency-domain wavefields (equation 6) that are necessary for computing the LS-RTM gradient, are computed in the same manner as the forward wavefields, namely by performing an on-the-fly DFT on the adjoint time-domain wavefields $\mathbf{v}_i$. These wavefields are computed by solving an adjoint (i.e. time-reversed) wave equation $\mathbf{A}^{-\star}$, just as in conventional RTM. The LS-RTM gradient itself can be computed in the same time loop as the adjoint frequency-domain wavefields. We replace the complex Fourier domain wavefield $\bar{\mathbf{u}}_{jk}$ in equation 4 with the real and imaginary parts as given by equation 7 and compute the real part of the gradient through an on-the-fly DFT of the adjoint time-domain wavefield $\mathbf{v}_i$:

$$\bar{\mathbf{g}}_{jk} = -\sum_{i=1}^{n_t} (2\pi f_k)^2 \mathrm{diag}\Big[\bar{\mathbf{u}}_{jk}^{\mathrm{real}} \cos(2\pi f_k i \Delta t) - \bar{\mathbf{u}}_{jk}^{\mathrm{imag}} \sin(2\pi f_k i \Delta)\Big]\mathbf{v}_i. \tag{8}$$

This equation gives us an expression for calculating the LS-RTM gradient in the frequency domain with a time-modeling code for any given frequency f_k, not just evenly spaced frequencies as obtained with an FFT. Unlike in the time domain or in equations 4 – 6, we never need to store the full time-domain wavefield $\mathbf{u}_i$ with $i = 1, \ldots, n_t$ in history. Instead, forward wavefields and gradients are computed as a running sum within forward or adjoint time loops and only require the storage of two wavefields per frequency at a time (real and imaginary parts). During a single time-stepping loop, it is of course possible to compute multiple monochromatic wavefields for different frequencies by creating an inner loop within the on-the-fly DFT over the number of frequency-domain wavefields. While this does not increase the number of time-stepping loops (PDE solves), every additional frequency increases both memory requirements and computational cost within the respective time loop.

A FORWARD-ADJOINT PAIR FOR IMAGING THE IMPEDANCE

The gradient of the LS-RTM objective function that we derived in the previous sections uses the zero-lag cross-correlation imaging condition and maps seismic reflections in the observed data to a perturbation in the medium parameters, which are in this case the velocity in squared slowness (s^{-2} km^{2}). One of the well known shortcomings of imaging velocity perturbations with the zero-lag cross-correlation imaging condition, are low frequency imaging artifacts that result from backscattering of the source wavefield (e.g. Yoon and Marfurt, 2006; Guitton et al., 2007). This issue is especially problematic for imaging salt bodies, as high velocity contrasts in the migration velocity model lead to reflections/backscattering of the down-going wavefield, thus creating strong low-frequency artifacts in the image (Figure 1). This phenomenon has been well studied using radiation pattern analysis (Zhu et al., 2009; Zhou et al., 2015) and can be addressed by directional filtering of the wavefields, reparametrizations of the image or alternative imaging conditions. Here, we follow the approach from Op't Root et al. (2012) and address this issue by deriving the gradient of the LS-RTM objective function using the linearized inverse scattering imaging condition (ISIC). The image/gradient with ISIC is given by the sum of two terms, in which the low frequency artifacts have opposite signs and cancel each other, while the reflectors have equal signs and stack coherently (Whitmore and Crawley, 2012). In the frequency domain, the imaging condition is defined as (Op't Root et al., 2012):

$$\bar{\mathbf{g}}_{jk} = -\mathrm{Re}\left[\mathrm{diag}\big(\omega_k^2 \bar{\mathbf{u}}_{jk}\big)^* \mathrm{diag}\big(\mathbf{m}_0\big)\bar{\mathbf{v}}_{jk} - \sum_{l=1}^{n_{dim}} \mathrm{diag}\left(\frac{\partial \bar{\mathbf{u}}_{jk}}{\partial \mathbf{x}_l}\right)^* \frac{\partial \bar{\mathbf{v}}_{jk}}{\partial \mathbf{x}_l}\right], \tag{9}$$

where n_{dim} is the number of spatial dimensions and $\frac{\partial}{\partial \mathbf{x}_i}$ is the first spatial derivative of the respective dimension. The first term of this equation corresponds to the cross-correlation imaging condition as defined by equation 4 with an additional pointwise multiplication with the background squared slowness vector $\mathbf{m}_0$, while the second term is the sum of pointwise products of the first spatial derivatives of forward and adjoint wavefields. In Appendix A, we show that the linearized inverse scattering imaging condition corresponds in fact to imaging the acoustic impedance, making the gradient $\bar{\mathbf{g}}_{jk}$ an impedance update, rather than a velocity perturbation (squared slowness) update. Since we want to use ISIC in the context of imaging with on-the-fly Fourier transforms, we follow the approach from the previous section and compute the gradient in a (reverse) time-stepping loop, in which the adjoint frequency-domain wavefields $\bar{\mathbf{v}}_{jk}$ are obtained through an on-the-fly Fourier transform of the adjoint time-domain wavefield $\mathbf{v}_i$:

$$\begin{aligned} \bar{\mathbf{g}}_{jk} = -\sum_{i=1}^{n_t} \Bigg\{ & (2\pi f_k)^2 \mathrm{diag}\Big[\bar{\mathbf{u}}_{jk}^{\mathrm{real}} \cos(2\pi f_k i\Delta t) - \bar{\mathbf{u}}_{jk}^{\mathrm{imag}} \sin(2\pi f_k i\Delta)\Big] \mathrm{diag}\big(\mathbf{m}_0\big)\mathbf{v}_i - \\ & \sum_{l=1}^{n_{dim}} \mathrm{diag}\left[\frac{\partial \bar{\mathbf{u}}_{jk}^{\mathrm{real}}}{\partial \mathbf{x}_l} \cos(2\pi f_k i\Delta t) - \frac{\partial \bar{\mathbf{u}}_{jk}^{\mathrm{imag}}}{\partial \mathbf{x}_l} \sin(2\pi f_k i\Delta)\right] \frac{\partial \mathbf{v}_i}{\partial \mathbf{x}_l} \Bigg\}. \end{aligned} \tag{10}$$

As before, the frequency domain source wavefields $\bar{\mathbf{u}}_{jk}$ are computed in a separate forward time-stepping loop with an on-the-fly DFT of the forward time-domain wavefield (equation 7). As discussed earlier, we can think of the gradient expression as the action of an adjoint linear operator $\mathbf{J}^*$ on the LS-RTM data residual, which maps a perturbation in the data to a perturbation in the model (the seismic image). To use ISIC/impedance imaging in the context of LS-RTM, we need to derive the corresponding forward operator, i.e.; the linear map from the image domain to the data domain (Witte et al., 2017). Once again, we model the linearized data with a time-stepping modeling code, followed either by an on-the-fly DFT of the linearized wavefield or a time-to-frequency conversion of the time-domain shot record. First, we compute the perturbed (single scattered) wavefield $\delta\mathbf{u}_i$ of the i^{th} time step, such that the expression is the (time-domain) adjoint operation of equation 9:

$$\delta\mathbf{u}_i = -\mathbf{A}(\mathbf{m}_0)^{-1}\Bigg\{\mathrm{diag}\bigg(\frac{\partial^2\mathbf{u}_i}{\partial t^2}\bigg)\mathrm{diag}\big(\mathbf{m}_0\big)\delta\mathbf{z} - \sum_{l=1}^{n_{dim}}\frac{\partial}{\partial\mathbf{x}_l}\bigg[\mathrm{diag}\bigg(\frac{\partial\mathbf{u}_i}{\partial\mathbf{x}_l}\bigg)\delta\mathbf{z}\bigg]\Bigg\}, \tag{11}$$

where the vector $\delta\mathbf{z}$ denotes the impedance. The real and imaginary parts of the linearized data are then obtained by performing the on-the-fly DFT on the scattered time-domain wavefields $\delta\mathbf{u}_i$ and by restricting the wavefield to the receiver locations:

$$\begin{aligned}\bar{\mathbf{d}}_{jk}^{\text{pred}_r} &= \mathbf{P}_r\sum_{i=1}^{n_t}\cos(2\pi f_k i\Delta t)\delta\mathbf{u}_i,\\ \bar{\mathbf{d}}_{jk}^{\text{pred}_i} &= -\mathbf{P}_r\sum_{i=1}^{n_t}\sin(2\pi f_k i\Delta t)\delta\mathbf{u}_i.\end{aligned} \tag{12}$$

To obtain linearized data for an impedance image $\delta\mathbf{z}$ in the time domain, we can simply omit the on-the-fly DFT and directly apply the receiver restriction operator $\mathbf{P}_r$ to the time-domain wavefield $\delta\mathbf{u}_i$. This allows us to use the modeling operator in equation 11 also for conventional time-domain LS-RTM with impedance imaging. The optimization algorithm for LS-RTM with on-the-fly DFTs and sparsity-promotion that is described in the in the following section, is independent of which imaging condition is used and works for imaging the impedance using Equation 10, as well as imaging velocity contrasts (Equation 8). In theory, the equations provided here for acoustic/impedance modeling and computing the gradients using on-the-fly DFTs are exact adjoints and are able to pass adjoint tests if implemented as described.

[Figure 1 about here.]

SPARSITY-PROMOTING LEAST-SQUARES MIGRATION

The expressions we derived in the last sections allow us in principle to perform frequency-domain LS-RTM using a time-modeling code and without having to store time histories of wavefields. However, for conventional LS-RTM, the gradient is given by the full sum of all frequencies and source locations. The number of frequencies is determined by the recording length of the shot records and their sampling ratio (i.e. by the corresponding Nyquist frequency) and is generally quite large. Performing on-the-fly DFTs for a large number of frequencies in a single time loop is not only computationally expensive, but also requires the storage of all those wavefields and therefore defeats the purpose of this approach. Our method is therefore most useful, when we compute the gradients of the LS-RTM objective function for a small subset of frequencies, rather than for all frequencies. I.e.; for a single source index j, we compute $\hat{n}_f \ll n_f$ frequencies within one time loop, which requires the storage of $2\hat{n}_f$ wavefields for the gradient.

In the context of FWI, computing the gradient for a subset of frequencies makes sense, as it is generally desirable to invert the velocity model from low to high frequencies, using single or few temporal frequencies at a time (Bunks et al., 1995; Sirgue and Pratt, 2004). For seismic imaging on the other hand, the goal is to obtain a high resolution image from data with a broad frequency spectrum. As pointed out in Miller et al. (1987), using all temporal frequencies for seismic imaging is generally not necessary, as frequencies are typically oversampled and the sampling ratio depends on the scattering angles. This allows us to image seismic data using subsets of evenly spaced frequencies, where the frequency interval is determined by the recording length, which in turn depends on the target depth and the overburden velocity (Mulder and Plessix, 2004). Alternatively, the field of compressed sensing (CS) has recently provided a theoretical framework for sampling signals far below the Nyquist criterion using *randomized sampling* (Donoho, 2006; Candès et al., 2006b). The core idea of CS is to break the coherency of subsampling artifacts (aliases) into incoherent noise, by sampling on a non-uniform grid and to recover the signal using denoising techniques. Applied to LS-RTM, compressed sensing translates to working with random subsets of shots and frequencies, rather than evenly spaced samples. As is well known from compressive seismic imaging in the frequency domain (Herrmann and Li, 2012; Tu et al., 2013; Tu and Herrmann, 2015), migrating data that consists of subsets of randomly selected frequencies leads to noise/crosstalk in the images, similar to artifacts from simultaneous shots with source encoding (e.g. Romero et al., 2000; Tang and Biondi, 2009). This is demonstrated in Figure 2, which compares migration results in the time and frequency domain for single and multiple shots and using frequency subsampling. While migrating a single shot record using 20 randomly selected frequencies leads to an image with seemingly strong coherent artifacts, these artifacts convert to random noise after stacking 10 migrated shots, where each shot is migrated with a different set of randomly selected frequencies. Selecting the frequencies randomly is crucial for being able to recover the true image with sparsity-promoting minimization, as it breaks the coherency of the subsampling artifacts (namely aliases). Subsampling frequencies periodically, as well as truncating the ends of the frequency spectrum, leads to coherent artifacts (aliases), which are more difficult to separate from the signal.

[Figure 2 about here.]

Due to the fact that the frequency subsampling artifacts appear as incoherent noise in the image, it is possible to apply post-migration denoising techniques (refer to Buades et al., 2005, for an overview) or to address the artifacts as part of the inversion process itself and by modifying the least-squares RTM objective function. We follow the approaches in Herrmann et al. (2008) and Herrmann et al. (2015) and formulate LS-RTM as a sparsity-promoting minimization problem of the following form:

$$\begin{aligned} &\underset{\delta\mathbf{z}}{\text{minimize}} \quad \lambda||\mathbf{C}\,\delta\mathbf{z}||_1 + \frac{1}{2}||\mathbf{C}\,\delta\mathbf{z}||_2^2 \\ &\text{subject to:} \; \sum_{j=1}^{n_s}\sum_{k=1}^{n_f} \left\| \mathbf{M}_l^{-1}\mathbf{J}(\mathbf{m}_0, \bar{q}_{jk})\mathbf{M}_r^{-1}\delta\mathbf{z} - \mathbf{M}_l^{-1}\bar{\mathbf{d}}_{jk}^{\mathrm{obs}} \right\|_2 \leq \sigma. \end{aligned} \tag{13}$$

The goal of this problem is to minimize the combined ℓ_1-ℓ_2-norm of the unknown parameters, which are in our case the curvelet coefficients of the acoustic impedance $\delta\mathbf{z}$, obtained through multiplication with the forward curvelet transform $\mathbf{C}$. This is subject to the constraint that the predicted linearized data, given by the action of the linearized modeling operator $\mathbf{J}$ on $\delta\mathbf{z}$, fits the observed reflections $\bar{\mathbf{d}}_{jk}^{\mathrm{obs}}$ within some noise level σ. The matrices $\mathbf{M}_l^{-1}$ and $\mathbf{M}_r^{-1}$ are left- and right-hand preconditioners intended to improve the condition number of the system, such as mutes, depth scalings or half integrations (Herrmann et al., 2008). In the numerical examples, we use image mutes to set the water column to zero, as well as depth scalings to compensate for spherical divergence of the amplitudes.

In terms of image transforms, it is generally possible to choose any transform that leads to sparsity of the image in the transform domain, meaning the image can be well approximated by a small subset of coefficients (Figure 3). For seismic images, we are interested in local details, such as edges and singularities, which can be captured by wavelets or first differences. However, curvelets are not only multi-scale (like wavelets), but also multi-directional and thus are able to capture both point and line singularities, as well as smoothness along curved reflectors (Candès et al., 2006a; Ma and Plonka, 2010). As such, the curvelet transform is able to preserve structures in seismic images, even for a small number of coefficients, while sparse approximations of the original image coefficients lead to gaps in the subsurface structures (Figure 3). Another structure preserving transform that has been successfully used in the context of sparse seismic imaging, is the seislet transform (Fomel and Liu, 2010; Dutta, 2017). The seislet transform requires an estimation of the local slopes of events using plane wave deconstruction, while the curvelet transform detects dips automatically (Candès et al., 2006a).

[Figure 3 about here.]

The objective function in equation 13 is a modified formulation of the basis pursuit denoise (BPDN) problem (Donoho, 2006) and consists of a combined ℓ_1- and ℓ_2-norm, where λ is a trade-off (penalty) parameter that balances the two terms. The combined ℓ_1- and ℓ_2-norm is referred to as an elastic net in machine learning and has the effect of making the objective function strongly convex (Lorenz et al., 2014a). This allows us to optimize equation 13 with the linearized Bregman method, a simple to implement solver with few hyper parameters (Yin, 2010; Lorenz et al., 2014b), in which the penalty parameter λ plays a fundamentally different role than in comparable algorithms such as iterative soft thresholding (ISTA). A comparison of these two algorithms in the context of seismic imaging and the role of the thresholding parameter can be found in Herrmann et al. (2015). Practically, the ℓ_2-norm of the curvelet coefficients has no direct influence on the final image and in fact, for a large enough value of λ, the solution of equation 13 is equivalent to the solution of the BPDN problem, which is the same problem without the ℓ_2 regularization term in the objective function. However, we have to include this term to make the objevtive function strongly convex and the ℓ_2-term has to act on the same coefficients as the ℓ_1-term. Strong convexity enables us to solve the problem with the linearized Bregman method, which has nicer numerical properties than the related ISTA algorithm. Furthermore, there exists theoretical justification for using the linearized Bregman method to solve overdetermined problems, such as LS-RTM, by working with random subsets of rows/measurements in each iteration (Lorenz et al., 2014b). In the extreme case of working with single rows (shots) of the linear operator and data, the method is then equivalent to the sparse Kaczmarz solver (Mansour and Yilmaz, 2013). The linearized Bregman method is a fairly recent optimization algorithm, but is closely related to older, well-established algorithms, such as the augmented Lagrangian method and the alternating direction method of multipliers (e.g. Yin, 2010).

Sparsity-promoting LS-RTM in the frequency domain as a BPDN problem has been described, amongst others, in Herrmann and Li (2012) and the adaption of the linearized Bregman method to this problem has been discussed in Herrmann et al. (2015). In contrast to the approach presented here, these publications solve Helmholtz equations rather than performing time-to-frequency conversion and are thus limited in their scalability to large-scale high frequency data sets. However, as the algorithm for solving equation 13 only differs in the way how the linearized data and gradients are computed, we refer to these publications for details on sparsity-promoting LS-RTM and the linearized Bregman method. For the sake of completeness and reproducibility, we include the algorithm for minimizing equation 13 with the linearized Bregman method in Algorithm 1. The method works for imaging velocity contrasts as well as acoustic impedance and basically consists of three simple steps: modeling the predicted linearized data (equation 3 or 12), computing the gradient by migrating the data residual (equation 7 or 10) and soft thresholding the curvelet coefficients of the updated image. Each iteration involves choosing a random subset of $\hat{n}_s \ll n_s$ sources and $\hat{n}_f \ll n_f$ frequencies for which the gradient is computed. The step length t can be chosen to be either constant or based on a dynamic update rule (Lorenz et al., 2014a) (a constant step size was used in all numerical examples). The penalty parameter λ is set according to the maximum amplitude of the gradient in the first iteration, i.e.; $\lambda = c\|t_1\bar{\mathbf{g}}\|_\infty$ with $\|\mathbf{x}\|_\infty = \max(|x_1|, ..., |x_n|)$ being the infinity norm. The constant c determines how many coefficients pass the threshold in the first iteration. For $c = 1$, λ is set to the magnitude of the largest coefficient, causing no coefficient to pass the threshold in the first iteration. A smaller value of c, such as $c = 0.1$, results in a threshold that keeps all coefficients with magnitudes larger than $\frac{1}{10}$ of the maximum magnitude. In practice, this results in more coefficients entering into the solution early on. A detailed geophysical interpretation of each step of the algorithm is provided in Appendix B.

NUMERICAL EXAMPLES

In the following numerical examples, we will demonstrate that the method presented here, allows to perform least-squares migration at a fraction of the cost of conventional LS-RTM and without having to store or recompute time-domain wavefields. By using time-to-frequency conversion methods, the proposed algorithm does not rely on solving Helmholtz equations and scales to almost arbitrary model sizes and high frequencies. As part of our numerical examples, we will analyze the trade-off between memory usage and computational cost through varying the number of frequencies per iteration (frequency batch size) and compare it to time-domain imaging with optimal checkpointing. All of our numerical examples are computed with the Julia Devito Inversion (JUDI) framework (Witte et al., 2019), an open-source software package for seismic modeling and inversion based on Devito, a domain-specific language compiler for automated finite-difference computations (Lange et al., 2016; Louboutin et al., 2017). All wave equations were implemented using second

Algorithm 1: The linearized Bregman method for sparsity-promoting LS-RTM with randomized subsets of shots and frequencies. For each selected shot, a different subset of frequencies is selected; thus leading to a larger number of different frequencies in each image update. The algorithms consists of modeling the predicted linearized data $\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}}$ and migrating the data residual for obtaining the gradient. The image $\mathbf{x}_i$ is updated by applying the soft-thresholding function to the dual variable $\mathbf{z}_i$.

```
Initialize $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{z}_1 = \mathbf{0}$, $q$, $\lambda$, batch sizes $\hat{n}_s \ll n_s$ and $\hat{n}_f \ll n_f$
for $i = 1, ..., n$
    Select subset of shots and frequencies $\mathcal{S} = (\int_{\text{shot}}, \int_{\text{freq}}), |\int_{\text{shot}}| = \hat{n}_s, |\int_{\text{freq}}| = \hat{n}_f$
    $\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}} = \mathbf{M}_l^{-1}\mathbf{J}_{\mathcal{S}}\mathbf{M}_r^{-1}\mathbf{x}$
    $\bar{\mathbf{g}}_{\mathcal{S}} = \mathbf{M}_r^{-\top}\mathbf{J}_{\mathcal{S}}^{\top}\mathbf{M}_l^{-\top}\mathcal{P}_{\sigma}\big(\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}} - \bar{\mathbf{d}}_{\mathcal{S}}^{\text{obs}}\big)$
    $\mathbf{z}_{i+1} = \mathbf{z}_i - t_i\bar{\mathbf{g}}_{\mathcal{S}}$
    $\mathbf{x}_{i+1} = S_{\lambda}(\mathbf{z}_{i+1})$
end
note: $S_{\lambda}(\mathbf{z}) = \text{sign}(\mathbf{z}) \cdot \max(0, |\mathbf{z}| - \lambda)$
      $\mathcal{P}_{\sigma}\big(\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}} - \bar{\mathbf{d}}_{\mathcal{S}}^{\text{obs}}\big) = \max\Big(0, 1 - \frac{\sigma}{\|\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}} - \bar{\mathbf{d}}_{\mathcal{S}}^{\text{obs}}\|}\Big) \cdot \big(\bar{\mathbf{d}}_{\mathcal{S}}^{\text{pred}} - \bar{\mathbf{d}}_{\mathcal{S}}^{\text{obs}}\big)$
```

order finite differences in time and 8th order finite differences in space, unless specificed otherwise. To simulate wave propagation in an infinite domain, we use simple absorbing boundary conditions (ABCs) using a damping mask, as described in Clayton and Engquist (1977). Optimal checkpointing in Devito is implemented through a Python wrapper around the original Revolve library by Griewank and Walther (2000; Kukreja et al., 2018). The results shown in this section are reproducible with JUDI and scripts are provided on Github (Witte et al., 2019). The framework is implemented in the Julia programming language and uses a combination of distributed memory parallelism to parallelize over the shot locations and shared memory parallelism with OpenMP for the wave equation solves.

Sigsbee 2A

For our first numerical example, we use the Sigsbee 2A velocity model (Bergsma, 2001), a challenging salt model of 24.6 by 9.2 km. Due to the size of the model and the number of time steps that are required to propagate the wavefield to all parts of the domain, storing the forward wavefields in random access memory (RAM) can already be problematic. For our experiments, we model wave propagation for 10 seconds, which corresponds to $14,095$ time steps, using the time interval provided by the Courant-Friedrichs-Lewy (CFL) condition (Courant et al., 1967). Storing $14,095$ wavefields in RAM as single precision arrays with a grid spacing of 7.62 m requires 237 GB of memory, while saving the wavefields in memory at a sampling interval of 4 ms ($2,500$ wavefields) requires 42 GB. These numbers can be prohibitively expensive, especially if we want to compute multiple gradients in parallel on the same computational node. For this reason, wavefields, or a compressed version of them, are typically written to secondary storage devices. Alternatively, optimal checkpointing and on-the-fly Fourier transforms allow us to compute the gradients using substantially less or memory or to write only a small subset of wavefields to disk. For a fair comparison, we fix the allowed amount of memory for both methods to 700 MB, which corresponds to saving 40 real-valued or 20 complex wavefields in RAM.

For practical purposes, we compute gradients with the full time-domain source wavelet and data residuals as forward and adjoint sources, rather than modeling with monochromatic sources/residuals as indicated by equations 3 and 6. In other words, instead of performing Fourier transforms of the time-domain data, extracting the required frequencies and an inverse Fourier transform back to the time-domain, we use the unmodified time-domain wavelets and data residuals for modeling. This avoids having to extract monochromatic frequencies of the source wavelet and shot records, which is cumbersome if the frequencies do not lie on the corresponding time axis and therefore need to be interpolated. Overall we only need to perform two on-the-fly DFTs per frequency to compute the gradient for one shot record: one for the forward wavefield and one for the adjoint wavefield. Using the broadband wavelets and shot records as sources is possible, as extracting a monochromatic Fourier-domain wavefield for a given frequency from its corresponding monochromatic source, yields the same result (up to a constant) as modeling with the full time-domain source (Figure 4). Strictly speaking, this modification destroys the exact adjoint property of our demigration-migration operator pair, since we inject additional energy, but the introduced error is purely a scaling error and does not affect the position of reflectors.

[Figure 4 about here.]

In our first numerical experiment, we compare time-domain SPLS-RTM with optimal checkpointing and frequency-domain SPLS-RTM with on-the-fly Fourier transforms. The data set consists of 935 observed shot records with 10 seconds recording time and a peak frequency of 15 Hz and maximum frequency of 40 Hz. We simulate a marine streamer acquisition with 100 m minimum offset, 12 km maximum offset and 1200 evenly spaced hydrophones. We perform 20 iterations of the linearized Bregman method with 100 randomly selected shots per iteration (with replacement), which corresponds to approximately two passes trough the data. This means, in expectation, every shot record is migrated only twice. We found that the effect of the trade-off between batch size and number of iterations is negligible, as long as we avoid either extremes, i.e. using a very small batch size or very few iterations. For frequency-domain SPLS-RTM with on-the-fly Fourier transforms, we randomly select 20 different frequencies for each shot and each iteration. The frequencies are selected from a continuous frequency band between 3 and 40 Hz, according to the spectrum of the source wavelet. For this, we convert the frequency spectrum of the source to a cumulative probability function, generate uniform random values between 0 and 1, and select the corresponding frequencies on the $x-$axis of the probability function. This strategy ensures that a large number of random frequencies approximates, in expectation, the full spectrum of the source wavelet. Alternatively, the spectrum of the data can be used for this process, if the source wavelet has not been estimated prior to migration. The noise level σ in the algorithm was set to zero, since the observed data is noise free and a constant step size t was used for all SPLS-RTM examples. For every run, we used the largest possible step size that preserves numerical stability of the modeling scheme during all iterations. As a reference for our results, we also compute the time- and frequency-domain RTM images, which correspond to one full data pass, since every shot record is migrated once. The RTM and SPLS-RTM results for the frequency domain are shown in Figure 5 and a close-up comparison of all images is provided in Figure 6. While the frequency-domain RTM image is noisy due to frequency-subsampling artifacts, SPLS-RTM is able to map the incoherent noise to a coherent image and provides the same high-quality image as the time-domain method. The only post-processing that was applied to the results, is a linear depth scaling, to emphasize deeper reflectors.

[Figure 5 about here.]

[Figure 6 about here.]

As mentioned, the amount of memory for both optimal checkpointing and on-the-fly Fourier transforms is fixed to 700 MB in the previos experiments (40 real-valued or 20 complex wavefields). For optimal checkpointing, the amount of memory defines the trade-off between the number of checkpoints and the computational cost for recomputing wavefields. Decreasing the memory increases the computational cost, as fewer checkpoints can be saved and more wavefields need to be recomputed. For imaging with on-the-fly Fourier transforms however, this relationship does not apply. Decreasing the available memory decreases the number of wavefields that can be stored, but it also decreases the amount of computations, since less on-the-fly DFTs have to be computed. However, in this case, the trade-off is related to the amount of frequency subsampling artifacts in the images and to how many iterations of SPLS-RTM have to be performed to achieve the same quality of the final image. To demonstrate this relationship, we carry out a second numerical experiment in which we perform frequency-domain SPLS-RTM with on-the-fly DFTs using only 10 randomly selected frequencies per shot instead of 20. As expected, the convergence of the SPLS-RTM data misfit for 10 frequencies is considerably slower than for 20 frequencies and more iterations are necessary to bring the misfit of the current subset of shots and the image error down to a comparable level (Figure 7). On the other hand, each iteration requires only half the amount of memory and half the number of DFTs, which decreases the runtime for computing gradients. This is illustrated in Figure 8, in which we plot the time-to-solution for computing the gradient for a single shot record with on-the-fly DFTs as a function of the number of frequencies. For comparison, we also provide the runtime for computing a gradient using optimal checkpointing. All timings were obtained using a single CPU with 10 threads. A detailled description of the configuration and utilized hardware is given in Appendix C.

Furthermore, we demonstrate the effect of varying the batch size of shots versus the batch size of frequencies. For this, we repeat the previous experiment, but using twice as many shots, while keeping the numbers of frequencies fixed to 10. As evident from the convergence plots of the data residual and model error (Figure 7), increasing the batch size to 200 with a frequency batch size of 10 yields almost identical results as the example with 100 shots and 20 frequencies. This once again emphasizes, that the quality of the final results is similar if the product of number of iterations, shots and frequencies is kept constant. This observation is important, as it provides the possibility to adapt the optimization parameters (batch size, number of data passes) to the available computational resources. For example, if many computational nodes are available, we can increase the batch size of shots and decrease the number of frequencies, whereas we can decrease the batch size and increase the number of frequencies if only few nodes are available. Overall the time-domain result has the smallest data residual and model error, but also comes at higher computational cost, since all all wavefields have to be saved or reconstructed for computing the gradient. The convergence rate of the linearized Bregman method has not been analyzed in the literature, but is linear at best. Standard stochastic optimization methods, such as stochastic gradient descent, have a sub-linear convergence rate $\mathcal{O}(1/n)$, with n being the iteration number. However, in practice, the sublinear convergence rate is mostly problematic at very high iteration numbers when $\frac{1}{n}$ is very small and we want to solve the problem to convergence. During early iterations, the behavior of the algorithm is mostly dominated by a constant (algorithm-dependent) factor $\mathcal{O}(1)$, rather than the asymptotic behaviour, making the algorithm effective when only a small number of data passes is affordable.

[Figure 7 about here.]

[Figure 8 about here.]

In our final experiment using the Sigsbee 2A model, we investigate the sensitivity of the results to the choice of the regularization parameter λ (i.e. the soft thresholding value) and the effect of sparsity-promoting techniques to weak events in the seismic image. For this, we repeat the previous SPLS-RTM experiments with on-the-fly DFTs and a fixed number of shots and frequencies, but with varying values of λ. Namely, we use 100 randomly selected shots and 20 randomly selected frequencies per iteration with a total number of 20 iterations. In the previous examples, we chose the thresholding parameter through parameter testing, such that after the maximum number of iterations, most reflection events were recovered, but no incoherent noise was present in the final image. We now conduct two additional experiments in which we set $\lambda = 0$ (no sparsity promotion) and $\lambda = 4e - 4$ (stronger sparsity promotion than before), while previously, we used a value of $\lambda = 1e - 4$.

The effect of sparsity promotion and the choice of the thresholding parameter on two areas of the Sigsbee model is shown in Figures 9 to 11. Figure 9 shows a close-up view of the top-of-salt region, in which we have the best illumination in the model. Figure 9 b shows the result using no sparsity promotion, which means that no thresholding is applied to the coefficients. Even though no sparsity-promotion was used, the result shows a very high signal-to-noise ratio, since most of the incoherent noise stacked out over the course of the inversion. Furthermore, we can observe that in the cases where sparsity promotion was used, the final results are expectedly not very sensitive to the choice of λ (Figures 9 c and d). However, the situation is different for areas with poor illumination, such as in the sub-salt region shown in Figure 10. Here, we can observe that no sparsity promotion leads to a worse signal-to-noise ratio (Figure 10 b), but that the result is also more sensitive to

λ. Namely, a larger value of λ (i.e. stronger thresholding) removes weak events such as the diffractors (Figure 10 c and d). This observation is further emphasized in one-dimensional well-log comparisons (Figure 11), which were extracted at 12.1 km horizontal position.

[Figure 9 about here.]

[Figure 10 about here.]

[Figure 11 about here.]

It is important to consider, that we use a fixed number of data passes and therefore iterations for our examples, and that running a sufficiently large number of iterations will eventually recover the missing coefficients. However, for a larger value of λ, more iterations are necessary to recover the small coefficients, while choosing λ too large, causes incoherent noise to enter the solution after a few iterations. The right choice of λ is therefore a trade-off between noise, missing coefficients and the number of iterations. Using a smaller batch size of shots, does not inherently increase the incoherent noise, but leads to a poorer illumination and therefore increases the sensitivity to the choice of λ and the likelihood that coherent signal is accidentally removed or that noise in introduced into the solution.

BP Synthetic 2004

The results for the Sigsbee 2A model were obtained under ideal conditions, in which the noise-free observed data was generated with the same linearized modeling operator that was used for the inversion (inverse crime). To demonstrate that our proposed approach also works in a more realistic setting and scales to large-scale models, we test our algorithm on the BP synthetic 2004 model (Billette and Brandsberg-Dahl, 2005). The model has a size of 67.4 by 11.9 km and is interpolated to a grid spacing of 6.25 m ($10,789 \times 1,911$ grid points). We generate the observed data with the same acquisition geometry as the original data released by BP, using a 15 km streamer, 12 seconds recording time and 1340 shot locations. However, unlike the original data, we model the data without surface-related multiples and with a peak frequency of 20 Hz instead of 27 Hz. To provide a non-inversion crime setting, we model the data with the acoustic forward modeling operator and not with the demigration operator. We generate the data using the true velocity and density models, whereas for inversion, we only use a smooth migration velocity model, but no density. Furthermore, we model the observed data with a 16th order finite-difference (FD) stencil, while using an 8th order FD stencil for the inversion. Changing the discretization order is easily possible in JUDI, as the software uses Devito to automatically optimize and generate completely new source code for solving a specified wave equation in every individual run (Louboutin et al., 2018; Luporini et al., 2018). Since we use a different acoustic wave equation without density for inversion and change the finite-difference stencil order, this leads to different sets of source code for modeling the observed data and running SPLS-RTM. Furthermore, we add Gaussian noise to all observed shot gathers with a signal-to-noise ratio of 17 dB (8.13 dB after muting the direct wave).

[Figure 12 about here.]

As before, we compare frequency-domain RTM with on-the-fly DFTs and 20 randomly selected frequencies per shot to frequency-domain SPLS-RTM with randomized shots (Figures 12 and 13). For this example, we run the inversion for 20 iterations, using 200 randomly selected shots per iteration with 20 frequencies per shot. As for the Sigsbee example, the frequencies are selected from a continuous frequency band between 3 and 45 Hz, using the spectrum of the source wavelet as the probability distribution. As a reference solution, we also compute the time-domain SPLS-RTM image using optimal checkpointing (Figure 13 a). As indicated by a close-up comparison of the results, SPLS-RTM is once gain able to map the frequency subsampling artifacts in the RTM image into coherent energy and provide an image of similar quality as the time-domain method (Figures 12 – 15). Since the observed data is not generated with the demigration operator that is used for inversion and is generated using a density model, the amplitudes of the predicted data can never exactly match the amplitudes of the observed data, which is why the data misfit for the current subset of shots only decays by 32 percent (Figure 16 a). This amplitude mismatch in combination with a smooth migration velocity model without sharp salt boundaries is also responsible for the the slight blur of the salt dome's top reflector.

[Figure 13 about here.]

[Figure 14 about here.]

[Figure 15 about here.]

As in our Sigsbee example, we measure the time-to-solution for computing the gradient for a single shot record as a function of the number of frequencies (Figure 16 b). The timings are obtained with the same computational set up as before, using a single Intel Xeon CPU with 10 cores (Appendix C). In this particular case, computing one gradient with optimal checkpointing takes longer than computing a gradient for 64 frequencies with on-the-fly DFTs, but less time than 128 frequencies. In the RTM and SPLS-RTM example, we only use 20 randomly selected frequencies per shot record, making the computation of a single gradient approximately three times faster than the computation of a time-domain gradient with optimal checkpointing and the same amount of computational resources. While the frequency-domain SPLS-RTM result using only 20 randomly selected frequencies per shot is considerably faster than the its time-domain equivalent, it also still exhibits a slight amount of low-amplitude noise, which underlines the trade-off between number of frequencies and quality of the result that is inherent to this approach. While increasing the number of frequencies yields a result that is closer to the time-domain image, it also diminishes the computational speed up in comparison to optimal checkpointing.

[Figure 16 about here.]

DISCUSSION

The main challenges of least-squares reverse-time migration are the large number of shots that have to be migrated in each iteration of gradient-based optimization algorithms, as well as the necessary access to the forward wavefields in reverse order for computing the gradient. A straight-forward implementation of time-domain LS-RTM is to forward propagate the source wavefield for all time steps, store the wavefields in memory and access them in reverse order during the adjoint time loop. The obvious drawback of this approach is that the required memory grows linearly with the number of time steps and the approach quickly becomes infeasible for any realistically sized models and recording times. Similarly, saving and reconstructing the wavefields from the boundary scales linearly with the number of time steps as well, but the asymptotic behavior has a smaller constant than saving the full wavefields, as the wavefield is only saved in a subset of the domain. Alternatively, the problem can be addressed by storing only a small subset of forward wavefields and by recomputing the in-between wavefields from the last checkpoint during the reverse time loop. Checkpointing therefore provides the user with a possible trade-off between memory usage and the number of time steps that have to be recomputed, which is captured in the recomputation ratio. This parameter is given by the total number of time steps (including recomputations) divided by the original number of forward time steps. In an important series of publications, Griewank and Walther (2000) describe an algorithm for computing the *optimal* trade-off between these quantities and show that the amount of required memory and the recomputation ratio for optimal checkpointing grow logarithmically with the number of time steps n_t (Table 1).

[Table 1 about here.]

For compressive imaging with on-the-fly Fourier transforms, the asymptotic behavior of the required memory and additional computations is fundamentally different, as these quantities grow as a function of the number of frequencies and not with the number of time steps (Table 1). Both memory and computational cost grow linearly with the number of frequencies and therefore have worse asymptotic behavior than optimal checkpointing, but they are independent of the number of time steps. This leads us to the critical question of how many frequencies n_f are required for computing the gradient. One could argue that for a fair comparison, the number of frequencies should be equal to the number of computational time steps—i.e. the LS-RTM gradient should be computed as the sum over all $n_f = n_t$ frequencies, as this yields the same gradient as in the time-domain, where the gradient is computed as a sum over all computational time steps. However, our numerical experiments have shown that, in practice, we can get away with a much smaller number of frequencies than time steps and still achieve satisfactory imaging results, if we cast LS-RTM as an ℓ_1-norm minimization problem. Furthermore, our examples show that the number of frequencies influence the convergence of the algorithm and determine how many iterations we need to run to obtain results of comparable quality. In general, computing the gradient with a smaller subset of frequencies leads to stronger subsampling artifacts and requires a more aggressive thresholding of the image's curvelet coefficients to remove the noise, which in turn increases the necessary number of iterations. However, a quantitative relationship between the number of frequencies, the amount of noise and the required number of iterations is at this point not available and will require further investigations. A possible reference point is the compressive sensing literature itself, which provides theoretical relationships between the sparsity of a signal, the amount of necessary measurements and the reconstruction error. In particular, Donoho (2006) shows that the the number of required measurements grows with $\mathcal{O}(n_s \log n_l)$, where n_s is the number of most important coefficients of the signal in some transform domain and n_l is the signal length. For compressive imaging with on-the-fly DFTs, this means that the number of necessary frequencies and shots will depend on the sparsity of the unknown image and the number of gridpoints, but not on the number of time steps.

In our analysis (Table 1), we assume that the number of grid points is fixed, but it is worth mentioning that the methods have different asymptotic behaviors as a function of the model size. Namely, assuming we have n grid points in each dimension, the boundary reconstruction method scales with $\mathcal{O}(n)$ in 2D and $\mathcal{O}(n^2)$ in 3D, while all other methods scale with $\mathcal{O}(n^2)$ in 2D and $\mathcal{O}(n^3)$ in 3D. In practice, the asymptotic behavior of the memory requirements can only serve as a guideline, as it describes its limiting behavior and the question of which approach requires the least amount of memory, needs to be evaluated on a case by case basis and depends on the specific number of dimensions, time steps and grid points. However, for a large enough number of time steps, optimal checkpointing will eventually always require less memory than the saving the wavefields at the boundary.

The second important question is how much additional computations have to be carried out for a given amount of memory, which will determine how fast the two approaches perform in practice. In our first numerical example, we fix the available memory for both on-the-fly DFTs and optimal checkpointing to 700 MB, which corresponds to 20 frequency-domain wavefields or 40 time-domain checkpoints. The recomputation ratio with 40 checkpoints for $14,095$ time steps is estimated by the Revolve library as 3.06, which means that approximately $2n_t$ additional forward time steps have to be modeled for computing a gradient (Griewank and Walther, 2000). For time-to-frequency conversion with on-the-fly DFTs, the main additional computational cost results from multiplying the time-domain wavefield of the current time step with the sine and cosine terms of equation 7. Multiple frequencies can be computed in a single time loop, but result in an inner loop over the number of frequencies at each time step. The additional computational cost for on-the-fly DFTs therefore depends on the number of frequencies and how the DFT is implemented. Evaluating trigonometric functions is generally expensive, but since the sine and cosine terms in the equation 7 are not spatially varying, they can be precomputed and used for all grid points. Furthermore, it is possible to compute the DFT on a coarser time grid than the computational time axis (e.g. at the Nyquist rate), which reduces the number of floating point operations. For optimal checkpointing, the computational cost of remodeling a given number time steps depends on the type of wave equation and the order of the finite-difference (FD) stencil. Modeling anisotropic or elastic wave equations is more expensive than solving acoustic wave equations and the amount of floating point operations increases further for higher order FD stencils. On the other hand, on-the-fly DFTs are independent of the discretization order (at least for a fixed grid spacing), but they also become more expensive for wave equations with coupled wavefields, as the cost increases linearly with the number of DFTs that have to be computed for each wavefield. In our numerical examples with the Sigsbee 2A and BP Synthetic 2004 model, we found that computing the gradient for one source using optimal checkpointing with $\log n_t$ checkpoints, was timewise equivalent to computing a frequency-domain gradient with approximately 70 to 100 frequencies. Since we only used 10 or 20 frequencies per shot record for our imaging examples with on-the-fly DFTs, we were able to reduce the time-to-solution per gradient by a factor of 3 to 4 in comparison to optimal checkpointing, with the same amount of computational resources.

Regarding the effects of sparsity-promoting minimization in the context of seismic imaging, our numerical examples demonstrate that seismic images are generally well approximated by a small percentage of curvelet coefficients, but that not running the optimization algorithm to convergence can harm the reconstruction quality. Similar to large-scale machine learning problems, seismic imaging is computationally too expensive to run optimization algorithms to convergence and typically only a fixed number of data passes (i.e. PDE solves) are possible. In this scenario, the reconstruction is sensitive to the choice of hyper-parameters. Specifically, for the linearized Bregman method, a value of λ that is too large removes too many coefficients from the image, while a value that is too small, allows noise to re-enter the solution. Running the algorithm for a fixed number of data passes therefore does not guarantee that all coefficients such as weak reflectors or diffractors are able to re-enter the solution. Our examples show that these effects are typically less severe in parts of the image with good illumination, but that areas with poor illumination are more sensitive to hyper-parameters. On the other hand, the areas of poor illumination are also the parts of the images that exhibit the strongest subsampling artifacts and where the benefit of sparsity-promotion is the most apparent. Generally, making exact predictions about the behavior of the approximation error is challenging, as non-linear image approximations (i.e. keeping a fixed percentage of sorted coefficients), are not as well understood as linear image approximations.

CONCLUSIONS

Least-squares reverse-time migration in the frequency domain avoids the problem of having to store or recompute a large number of time-domain wavefields for computing gradients with the adjoint-state method. Conventionally, the gradient has to be computed for each frequency separately by solving the corresponding Helmholtz equation. On-the-fly Fourier transforms offer the possibility to compute monochromatic wavefields with a time-domain modeling code and to obtain an arbitrary number of frequencies within a single time-stepping loop. Formulating least-squares migration as a sparsity-promoting minimization problem allows us to work with small random subsets of shots and frequencies, thus making it possible to perform least-squares imaging at a fraction of the cost of conventional LS-RTM, using as few as two passes through the data set and without having to save or recompute time-domain wavefields.

On-the-fly discrete Fourier transforms offer a fast and easy-to-implement alternative to optimal checkpointing, in which the amount of memory and computational overhead does not depend on the number of time steps, but on the number of frequencies for which the gradient is computed. By varying the batch size of shots or frequencies, the method allows to choose a trade-off between the amount of computations and memory versus the number of iterations and the quality of the final result. Optimal checkpointing on the other hand, provides a trade off between memory usage versus the amount of additional computations, as fewer checkpoints require more time steps that need to be recomputed. This makes imaging with on-the-fly Fourier transforms interesting for applications where both storage and CPU time are expensive, such as cloud computing, as we can trade image quality for computational resources. Another advantageous scenario for our approach is the case where only a small amount of computational resources are available for a long time, in which case we can trade computational resources for a large number of iterations, where each iteration only uses a small number of frequencies and is cheap in terms of both storage and compute.

ACKNOWLEDGMENTS

We would like to acknowledge Charles Jones (Osokey Ltd.) for the feedback and support of this project, as well as Henryk Modzelewski (The University of British Columbia) for the technical support. Furthermore we would like to thank Stéphane Operto (CNRS), Daniel Trad (University of Calgary), Musa Maharramov (ExxonMobile) and the two remaining anonymous reviewers for their feedback and help with improving this paper. This research was carried out as part of the SINBAD project with the support of the member organizations of the SINBAD Consortium. Finally, we would like to acknowledge the University of British Columbia (UBC), where part of this research was carried out.

APPENDIX A

A comparison of reverse time-migration with the linearized inverse scattering imaging condition (ISIC) (Op't Root et al., 2012; Whitmore and Crawley, 2012) and seismic imaging with the acoustic impedance (Douma et al., 2010), reveals that the respective sensitivity kernels at equivalent. The sensitivity kernel (i.e. the image) for the acoustic impedance $K_Z(\mathbf{x})$ is defined in Douma et al. (2010) as the sum of the sensitivity kernels of the spatially varying bulk modulus $\boldsymbol{\kappa}$ and density $\boldsymbol{\rho}$:

$$K_Z(\mathbf{x}) = K_\kappa(\mathbf{x}) + K_\rho(\mathbf{x}), \tag{A.1}$$

where the sensitivity kernel is defined as:

$$K_\kappa(\mathbf{x}) = -\frac{1}{\boldsymbol{\kappa}} \sum_{i=1}^{n_t} \mathrm{diag}\big(\dot{\mathbf{u}}_i\big)\dot{\mathbf{v}}_i, \tag{A.2}$$

and $\dot{\mathbf{u}}_i, \dot{\mathbf{v}}_i$ are first time-derivatives of forward and adjoint wavefields. The sensitivity kernel for the density is given by:

$$K_\rho(\mathbf{x}) = \frac{1}{\boldsymbol{\rho}} \sum_{i=1}^{n_t} \mathrm{diag}\big(\nabla\mathbf{u}_i\big)\nabla\mathbf{v}_i, \tag{A.3}$$

where the second term denotes the pointwise products of the spatial derivatives of forward and adjoint wavefields. Combining the two equations and substituting the bulk modules by the velocity and density yields:

$$K_Z(\mathbf{x}) = -\sum_{i=1}^{n_t} \left[\frac{1}{\boldsymbol{\rho}\mathbf{v}^2} \mathrm{diag}\big(\dot{\mathbf{u}}_i\big)\dot{\mathbf{v}}_i - \frac{1}{\boldsymbol{\rho}} \mathrm{diag}\big(\nabla\mathbf{u}_i\big)\nabla\mathbf{v}_i \right]. \tag{A.4}$$

A comparison of this expression with equation 9 reveals that the impedance kernel is equivalent to the linearized inverse scattering imaging condition for $\rho(\mathbf{x}) = 1$. This is independent of whether we consider the frequency-domain formulation of ISIC (Op't Root et al., 2012) or its time-domain equivalent (Whitmore and Crawley, 2012). Multiplication of the frequency-domain source wavefield with ω^2 corresponds to a second time derivative of the forward time-domain wavefield, or to first time derivatives of both forward and adjoint wavefields.

APPENDIX B

The linearized Bregman method used in our paper to solve the ℓ_1-minimization problem in Equation 13 is a specialized case of a broader class of optimization problems for solving convex, but potentially non-differentiable objective functions with (in-)equality constraints. Namely, the linearized Bregman method is a simplification of the more general Bregman iterative regularization method, and can be derived by linearizing the quadratic data fidelity term in classic Bregman iterations (Yin et al., 2008).

The advantage of the linearized Bregman algorithm in comparison to the more general Bregman iterative regularization or the augmented Lagrangian method, is that every iteration involves only two matrix-vector products; $\mathbf{Jx}$ and $\mathbf{J}^\top\big(\mathbf{d}_{\text{pred}} - \mathbf{d}_{\text{obs}}\big)$. In the case of seismic imaging, where $\mathbf{J}$ is the linearized Born scattering operator, these matrix-vector products correspond precisely to linearized Born modeling (Step 4) and reverse-time migration (Step 5). The Born scattering operator for a full seismic survey is overdetermined, i.e. there are (significantly) more observed data points than coefficients in the seismic image. However, because working with the full Born scattering operator in each iteration is prohibitively expensive, as it involves the demigration/migration of all shots, we can work with random subsets of frequencies and shots (or simultaneous shots) (Lorenz et al., 2014b). This has the effect of turning the overdetermined problem, into an underdetermined problem, but, as demonstrated in Figure 16, also leads to a noisy image (variable $\mathbf{z}$ in the algorithm). By using sparsity-promoting minimization, it is possible to remove the noise and recover the true image in the subsequent iterations. However, in order for sparsity promotion to be successful, it is crucial that the noise is in fact incoherent and does not contain any aliases. By choosing the subsets of frequencies (and shots) randomly in each iteration, we guaran-

Algorithm 2: Simplified version of the linearized Bregman algorithm from algorithm 1 without preconditioners.

```
1. Initialize $\mathbf{x}_1 = \mathbf{0}$, $\mathbf{z}_1 = \mathbf{0}$, $q$, $\lambda$, batch sizes $\hat{n}_s \ll n_s$ and $\hat{n}_f \ll n_f$
2. for  $i = 1, ..., n$
3.         Select subset of shots and frequencies $\mathcal{S} = (\int_{\text{shot}}, \int_{\text{freq}}), |\int_{\text{shot}}| = \hat{n}_s, |\int_{\text{freq}}| = \hat{n}_f$
4.         $\bar{\mathbf{d}}^{\text{pred}}_{\mathcal{S}} = \mathbf{J}_{\mathcal{S}}\mathbf{x}$
5.         $\bar{\mathbf{g}}_{\mathcal{S}} = \mathbf{J}^{\top}_{\mathcal{S}}(\bar{\mathbf{d}}^{\text{pred}}_{\mathcal{S}} - \bar{\mathbf{d}}^{\text{obs}}_{\mathcal{S}})$
6.         $\mathbf{z}_{i+1} = \mathbf{z}_i - t_i\bar{\mathbf{g}}_{\mathcal{S}}$
7.         $\mathbf{x}_{i+1} = \mathbf{C}^{\top}S_{\lambda}(\mathbf{C}\mathbf{z}_{i+1})$
8. end
with $S_{\lambda}(\mathbf{z}) = \text{sign}(\mathbf{z}) \cdot \max(0, |\mathbf{z}| - \lambda)$
```

tee that the image contains only incoherent noise and no aliases or wrap-around effects, as would be the case for periodic subsampling.

In step 7 of the algorithm, we compute the curvelet transform of the noisy image $\mathbf{z}_i$, since we want to promote sparsity of the image in the curvelet domain. This is followed by applying the soft-thresholding function to the noisy coefficients, which effectively sets all coefficients smaller than λ to zero and shrinks the magnitude of the remaining values by λ. In the early iterations, this sets not only the noise, but also coefficients of reflectors to zero. During the subsequent iterations, the amplitude of the reflector coefficients is continuously increased, such that toward the final iterations, the soft thresholding function removes (ideally) only the noise. This is illustrated in Figure 17, which shows the variables $\mathbf{z}_1$ and $\mathbf{x}_1$ in comparison to $\mathbf{z}_{20}$ and $\mathbf{x}_{20}$, i.e. both variables during the first and final iteration.

[Figure 17 about here.]

APPENDIX C

The timings shown in Figure 8 were computed for the Sigsbee 2A velocity model with a grid spacing of 7.62 m, which corresponds to $3,201 \times 1,201$ grid points. The time stepping interval according to the CFL condition is 0.71 ms, resulting in $14,095$ time steps for 10 seconds modeling time. The timings for the BP model (Figure #f9b) were computing using a grid spacing of 6.25 m ($10,789 \times 1,911$ grid points) and 12 seconds modeling time with a time stepping interval of 0.548 ms ($21,898$ time steps).

All timings were computed with an Intel Xeon E5 v2 processors (2.8 GHz) with 10 cores and 128 GB RAM. Each shown time measurement is the smallest runtime of three individual runs. The examples were computed using 10 threads, in which each thread is pinned to a specific core (thread pinning).

The following software was used for the timings and numerical case studies: Julia (v0.6.3), JUDI (v0.2.1:dft-paper), Python (v3.6.5), Devito (v3.2.0:dft-paper), Intel compiler (v16.0.3).

REFERENCES

Baysal, E., D. D. Kosloff, , and J. W. C. Sherwood, 1983, Reverse time migration: GEOPHYSICS, **48**, 1514–1524.

Bergsma, E., 2001, Sigsbee2a 2D synthetic dataset: `http://www.delphi.tudelft.nl/SMAART/sigsbee2a.htm`. (Publicly Released 'SMAART' Data Sets).

Billette, F., and S. Brandsberg-Dahl, 2005, The 2004 BP velocity benchmark., *in* 67th Annual International Meeting, EAGE, Expanded Abstracts: EAGE, B035.

Buades, A., B. Coll, and J.-M. Morel, 2005, A review of image denoising algorithms, with a new one: Multiscale Modeling & Simulation, **4**, 490–530.

Bunks, C., F. M. Saleck, S. Zaleski, and G. Chavent, 1995, Multiscale seismic waveform inversion: GEOPHYSICS, **60**, 1457–1473.

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

Candès, E. J., J. K. Romberg, and T. Tao, 2006b, Stable signal recovery from incomplete and inaccurate measurements: Communications on Pure and Applied Mathematics, **59**, 1207–1223.

Chen, Y., J. Yuan, S. Zu, S. Qu, and S. Gan, 2015, Seismic imaging of simultaneous-source data using constrained least-squares reverse time migration: Journal of Applied Geophysics, **114**, 32–35.

Clayton, R., and B. Engquist, 1977, Absorbing boundary conditions for acoustic and elastic wave equations: Bulletin of the seismological society of America, **67**, 1529–1540.

Courant, R., K. Friedrichs, and H. Lewy, 1967, On the partial difference equations of mathematical physics: International Business Machines (IBM) Journal of Research and Development, **11**, 215–234.

Dai, W., P. Fowler, and G. Schuster, 2012, Multi-source least-squares reverse time migration: Geophysical Prospecting, **70**, no. 4, 681–695.

Dai, W., Y. Huang, and G. T. Schuster, 2013, Least-squares reverse time migration of marine data with frequency-selection encoding: GEOPHYSICS, **78**, S233–S242.

Dai, W., W. Xin, and G. Schuster, 2011, Least-squares migration of multisource data with a deblurring filter: GEOPHYSICS, **76**, R135–R146.

Dong, S., J. Cai, M. Guo, S. Suh, Z. Zhang, B. Wang, and Z. Li, 2012, Least-squares reverse time migration: towards true amplitude imaging and improving the resolution: 82nd Annual International Meeting, SEG, Expanded Abstracts, 1–5.

Donoho, D., 2006, Compressed sensing: Institute of Electrical and Electronics Engineers (IEEE) Transactions on Information Theory, **52**, 1289–1306.

Douma, H., D. Yingst, I. Vasconcelos, and J. Tromp, 2010, On the connection between artifact filtering in reverse-time migration and adjoint tomography: GEOPHYSICS, **75**, S219–S223.

Dutta, G., 2017, Sparse least-squares reverse time migration using Seislets: Journal of Applied Geophysics, **136**, 142 – 155.

Etienne, V., S. Operto, J. Virieux, Y. Jia, et al., 2010, Computational issues and strategies related to full waveform inversion in 3D elastic media: Methodological developments: 2010 SEG Annual Meeting, Society of Exploration Geophysicists, 1050–1054.

Fomel, S., and Y. Liu, 2010, Seislet transform and seislet frame: GEOPHYSICS, **75**, V25–V38.

Furse, C. M., 1998, Faster than Fourier-ultra-efficient time-to-frequency domain conversions for FDTD: Institute of Electrical and Electronics Engineers (IEEE): Antennas and Propagation Society International Symposium, 536–539 vol.1.

Griewank, A., and A. Walther, 2000, Algorithm 799: Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation: Association for Computing Machinery (ACM) Transactions on Mathematical Software, **26**, 19–45.

Guitton, A., B. Kaelin, and B. Biondi, 2007, Least-squares attenuation of reverse-time-migration artifacts: GEOPHYSICS, **72**, S19–S23.

Ha, W., S.-G. Kang, and C. Shin, 2015, 3D Laplace-domain waveform inversion using a low-frequency time-domain modeling algorithm: GEOPHYSICS, **80**, R1–R13.

Herrmann, F. J., 2010, Randomized sampling and sparsity: Getting more information from fewer samples: GEOPHYSICS, **75**, WB173–WB187.

Herrmann, F. J., and X. Li, 2012, Efficient least-squares imaging with sparsity promotion and compressive sensing: Geophysical Prospecting, **60**, 696–712.

Herrmann, F. J., P. P. Moghaddam, and C. Stolk, 2008, Sparsity- and continuity- promoting seismic image recovery with Curvelet frames: Applied and Computational Harmonic Analysis, **24**, 150–173.

Herrmann, F. J., N. Tu, and E. Esser, 2015, Fast ”online” migration with Compressive Sensing: 77th Conference and Exhibition, EAGE, Expanded Abstracts.

Huang, W., and H.-W. Zhou, 2014, *in* Stochastic conjugate gradient method for least-square seismic inversion problems: 4003–4007.

Kim, Y., C. Shin, H. Calandra, and D.-J. Min, 2013, An algorithm for 3D acoustic time-Laplace-Fourier-domain hybrid full waveform inversion: GEOPHYSICS, **78**, R151.

Kukreja, N., J. Hückelheim, M. Lange, M. Louboutin, A. Walther, S. W. Funke, and G. Gorman, 2018, High-level Python abstractions for optimal checkpointing in inversion problems: ArXiv e-prints.

Lambaré, G., J. Virieux, R. Madariaga, and S. Jin, 1992, Iterative asymptotic inversion in the acoustic approximation: GEOPHYSICS, **57**, 1138–1154.

Lange, M., N. Kukreja, M. Louboutin, F. Luporini, F. Vieira, V. Pandolfo, P. Velesko, P. Kazakas, and G. Gorman, 2016, Devito: Towards a generic finite difference DSL using Symbolic Python: Computing Research Repository.

Li, C., J. Huang, Z. Li, and R. Wang, 2018, Plane-wave least-squares reverse time migration with a preconditioned stochastic conjugate gradient method: GEOPHYSICS, **83**, S33–S46.

Liu, Y., 2013, Multisource least-squares extended reverse time migration with preconditioning guided gradient method: 83rd Annual International Meeting, SEG, Expanded Abstracts, 3709–3715.

Lorenz, D. A., F. Schöpfer, and S. Wenger, 2014a, The Linearized Bregman method via Split Feasibility Problems: Analysis and Generalizations: Society for Industrial and Applied Mathematics (SIAM): Journal on Imaging Sciences, **7**, no. 2, 1237–1262.

Lorenz, D. A., S. Wenger, F. Schöpfer, and M. Magnor, 2014b, A sparse Kaczmarz solver and a Linearized Bregman method for online compressed sensing: Institute of Electrical and Electronics Engineers (IEEE): International Conference on Image Processing, 1347–1351.

Louboutin, M., M. Lange, F. Luporini, N. Kukreja, P. A. Witte, F. J. Herrmann, P. Velesko, and G. J. Gorman, 2018, Devito: An embedded domain-specific language for finite differences and geophysical exploration: ArXiv preprints, **arXiv:1808.01995**.

Louboutin, M., P. Witte, M. Lange, N. Kukreja, F. Luporini, G. Gorman, and F. J. Herrmann, 2017, Full-waveform inversion, Part 1: Forward modeling: The Leading Edge, **36**, 1033–1036.

Lu, X., L. Han, J. Yu, and X. Chen, 2015, L1 norm constrained migration of blended data with the FISTA algorithm: Journal of Geophysics and Engineering, **12**, 620–628.

Luporini, F., M. Lange, M. Louboutin, N. Kukreja, J. Hückelheim, C. Yount, P. Witte, P. Kelly, G. Gorman, and F. Herrmann, 2018, Architecture and performance of Devito, a system for automated stencil computation.

Ma, J., and G. Plonka, 2010, The Curvelet Transform: Institute of Electrical and Electronics Engineers (IEEE), Signal Processing Magazine, **27**, 118–133.

Mansour, H., and Ö. Yilmaz, 2013, A fast randomized kaczmarz algorithm for sparse solutions of consistent linear systems: Computing Research Repositýory, **abs/1305.3803**.

McMechan, G. A., 1983, Migration by extrapolation of time-dependent boundary values: Geophysical Prospecting, **31**, 413–420.

Miller, D., M. Oristaglio, and G. Beylkin, 1987, A new slant on seismic imaging: Migration and integral geometry: GEOPHYSICS, **52**, 943–964.

Mulder, W. A., and R.-E. Plessix, 2004, How to choose a subset of frequencies in frequency-domain finite-difference migration: Geophysical Journal International, **158**, 801–812.

Nemeth, T., C. Wu, and G. T. Schuster, 1999, Least-squares migration of incomplete reflection data: GEOPHYSICS, **64**, 208–221.

Nguyen, B. D., and G. A. McMechan, 2015, Five ways to avoid storing source wavefield snapshots in 2D elastic prestack reverse time migration: GEOPHYSICS, **80**, S1–S18.

Nihei, K. T., and X. Li, 2007, Frequency response modelling of seismic waves using finite difference time domain with phase sensitive detection (TDPSD): Geophysical Journal International, **169**, 1069–1078.

Op't Root, T. J., C. C. Stolk, and M. V. de Hoop, 2012, Linearized inverse scattering based on seismic reverse time migration: Journal de Mathematiques Pures et Appliquees, **98**, 211–238.

Plessix, R.-E., 2006, A review of the adjoint-state method for computing the gradient of a functional with geophysical applications: Geophysical Journal International, **167**, 495–503.

Pratt, R. G., 1999, Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model: GEOPHYSICS, **64**, 888–901.

Romero, L. A., D. C. Ghiglia, C. C. Ober, and S. A. Morton, 2000, Phase encoding of shot records in prestack migration: GEOPHYSICS, **65**, 426–436.

Schuster, G. T., 1993, Least-squares cross-well migration: 63th Annual International Meeting, SEG, Expanded Abstracts, 110–113.

Sirgue, L., J. Etgen, U. Albertin, and S. Brandsberg-Dahl, 2010, System and method for 3D frequency domain waveform inversion based on 3D time-domain forward modeling. (US Patent 7,725,266).

Sirgue, L., and R. G. Pratt, 2004, Efficient waveform inversion and imaging: A strategy for selecting temporal frequencies: GEOPHYSICS, **69**, 231–248.

Symes, W. W., 2007, Reverse time migration with optimal checkpointing: GEOPHYSICS, **72**, SM213–SM221.

Tang, Y., and B. Biondi, 2009, Least-squares migration/inversion of blended data: 79th Annual International Meeting, SEG, Expanded Abstracts, 2859–2863.

Tarantola, A., 1984, Inversion of seismic reflection data in the acoustic approximation: GEOPHYSICS, **49**, 1259.

Tu, N., A. Y. Aravkin, T. van Leeuwen, and F. J. Herrmann, 2013, Fast least-squares migration with multiples and source estimation: EAGE Annual Conference Proceedings.

Tu, N., and F. Herrmann, 2015, Fast imaging with surface-related multiples by sparse inversion: Geophysical Journal International, **201**, 304–417.

Valenciano, A. A., 2008, Imaging by wave-equation inversion: PhD thesis, Stanford University.

van Leeuwen, T., A. Y. Aravkin, and F. J. Herrmann, 2011, Seismic Waveform Inversion by Stochastic Optimization: International Journal of Geophysics.

Watanabe, K., 2015, *in* Green's Functions for Laplace and Wave Equations: Springer International Publishing, 33–76.

Whitmore, N. D., 1983, Iterative depth migration by backward time propagation: 1983 SEG Annual Meeting, Expanded Abstracts, 382–385.

Whitmore, N. D., and S. Crawley, 2012, Applications of RTM inverse scattering imaging conditions: 82nd Annual International Meeting, SEG, Expanded Abstracts, 1–6.

Witte, P. A., M. Louboutin, and F. J. Herrmann, 2019, The Julia Devito Inversion Framework (JUDI): `https://github.com/slimgroup/JUDI.jl/tree/master/examples/compressive_splsrtm`.

Witte, P. A., M. Louboutin, N. Kukreja, F. Luporini, M. Lange, G. J. Gorman, and F. J. Herrmann, 2019, A large-scale framework for symbolic implementations of seismic inversion algorithms in julia: GEOPHYSICS, **84**, F57–F71.

Witte, P. A., M. Yang, and F. J. Herrmann, 2017, Sparsity-promoting least-squares migration with the linearized inverse scattering imaging condition: 79th Conference and Exhibition, EAGE, Expanded Abstracts.

Xu, K., and G. A. McMechan, 2014, 2d frequency-domain elastic full-waveform inversion using time-domain modeling and a multistep-length gradient approach: GEOPHYSICS, **79**, R41–R53.

Yin, W., 2010, Analysis and Generalizations of the Linearized Bregman Method: Society for Industrial and Applied Mathematics (SIAM): Journal on Imaging Sciences, **3**, 856–877.

Yin, W., S. Osher, D. Goldfarb, and J. Darbon, 2008, Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing: Society for Industrial and Applied Mathematics (SIAM): Journal on Imaging sciences, **1**, 143–168.

Yoon, K., and K. J. Marfurt, 2006, Reverse-time migration using the Poynting vector: Exploration Geophysics, **37**, 102–107.

Zeng, C., S. Dong, and B. Wang, 2014, Least-squares reverse time migration: Inversion-based imaging toward true reflectivity: The Leading Edge, **33**, no. 9, 962–964,966,968.

Zhou, W., R. Brossier, S. Operto, and J. Virieux, 2015, Full waveform inversion of diving & reflected waves for velocity model building with impedance inversion based on scale separation: Geophysical Journal International, **202**, 1535–1554.

Zhu, H., Y. Luo, T. Nissen-Meyer, C. Morency, and J. Tromp, 2009, Elastic imaging and time-lapse migration based on adjoint methods: GEOPHYSICS, **74**, WCA167–WCA177.

LIST OF FIGURES

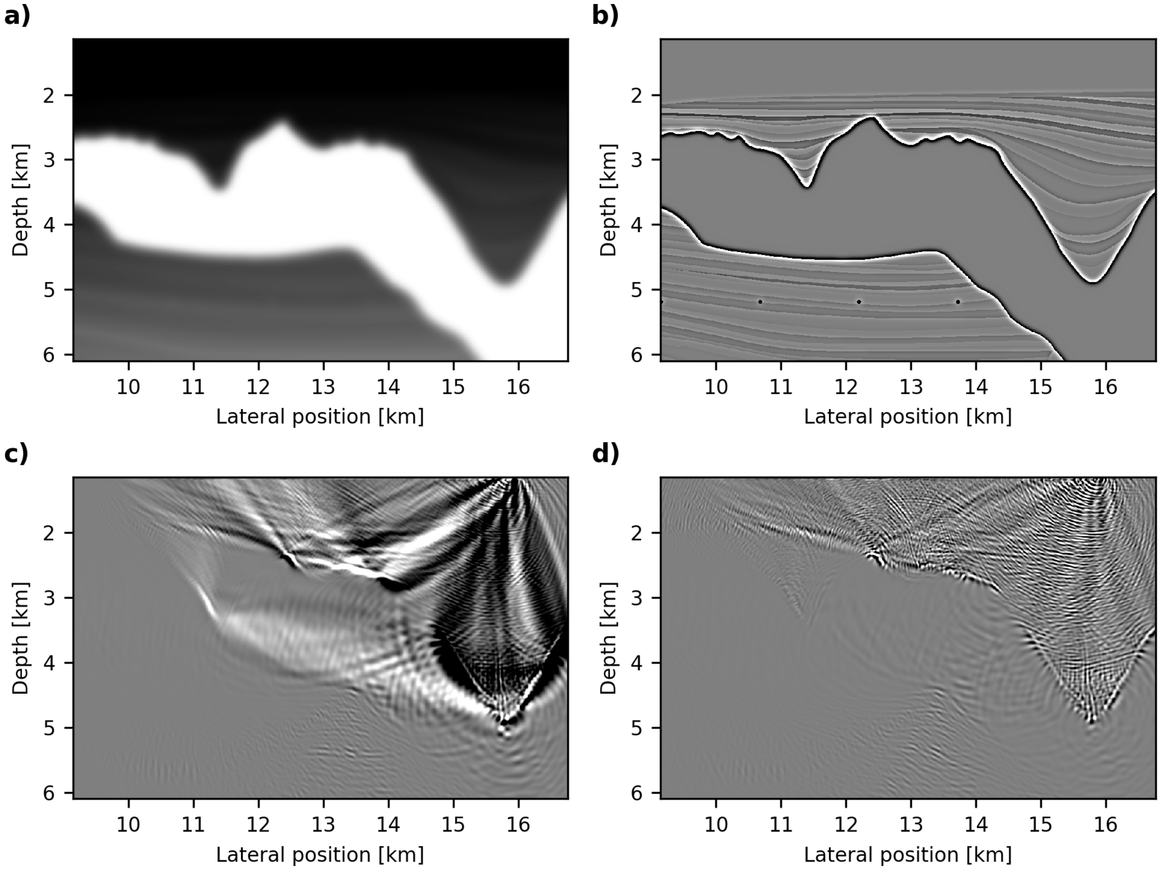

Figure 1: Comparison of RTM with the zero-lag cross-correlation imaging condition (c) versus the linearized inverse scattering imaging condition (d). The top row shows the smooth migration velocity model (a) and the true image (b). Both results were computed for a single shot record using 20 randomly chosen frequencies, which expectedly leads to crosstalk (spectral leakage) in the image and one-sided illumination of the salt body. However, the migration result with the cross-correlation imaging condition also suffers from strong low-frequency backscattering artifacts, whereas the inverse-scattering imaging condition is able to successfully suppress this energy.

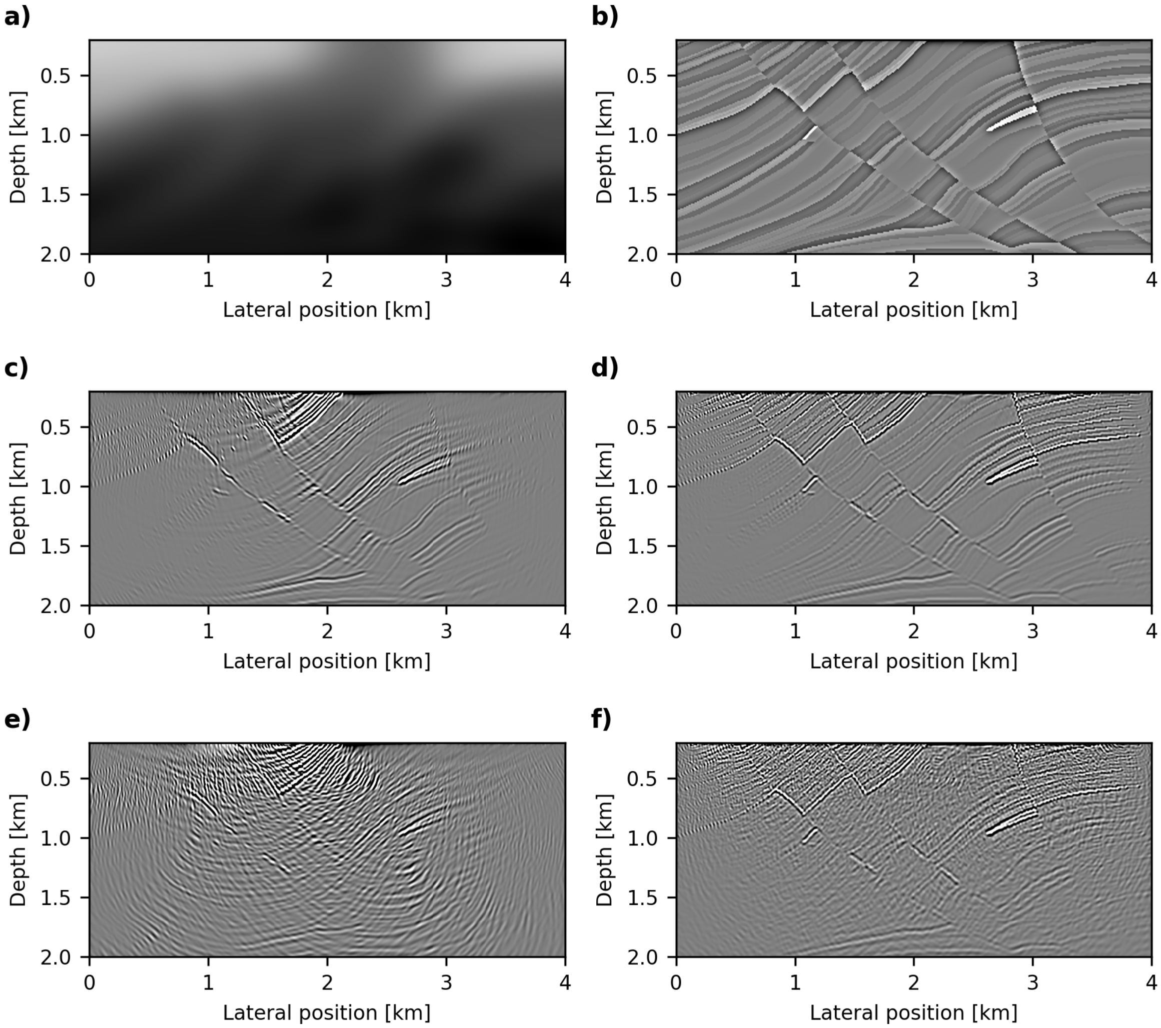

Figure 2: Comparisons of reverse-time migration in the time and frequency domain with on-the-fly Fourier transforms and frequency subsampling. Figure (a) is the migration velocity model and (b) is the true image. Figures (c) and (d) are the results of migrating a single shot and 10 shots in the time domain. Figures (e) and (f) are the corresponding results in the frequency domain with a subset of 20 randomly selected frequencies per shot. The migrated shot record in the frequency domain for 20 randomly selected frequencies (e) shows a very weak signal-to-noise ratio in comparison to its time-domain equivalent (c). However, when stacking the migration results of 10 shots, where each migrated shot consists of a different set of randomly selected frequencies, the reflectors stack coherently, while the subsampling artifacts appear as incoherent noise (f).

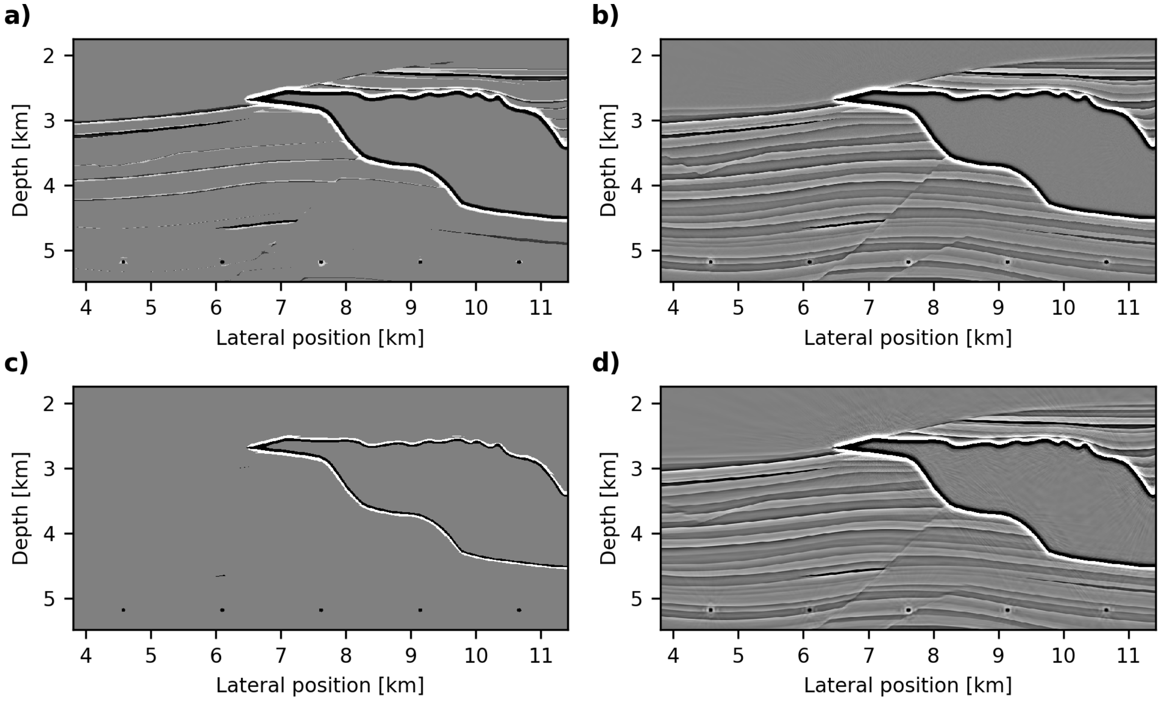

Figure 3: Sparse approximations of the true image in the image domain itself (left-hand column) and the curvlet domain (right-hand column). Figures (a) and (b) are sparse approximations using the largest 5 percent coefficients of the images in their respective domain and figures (c) and (d) are approximations using the largest 1 percent coefficients. The seismic image can be almost perfectly approximated by only 1 percent of the curvelet coefficients, as the sorted coefficients decay faster by magnitude than the original image coefficients. This makes the curvelet transform well suited for seismic imaging based on sparsity promotion.

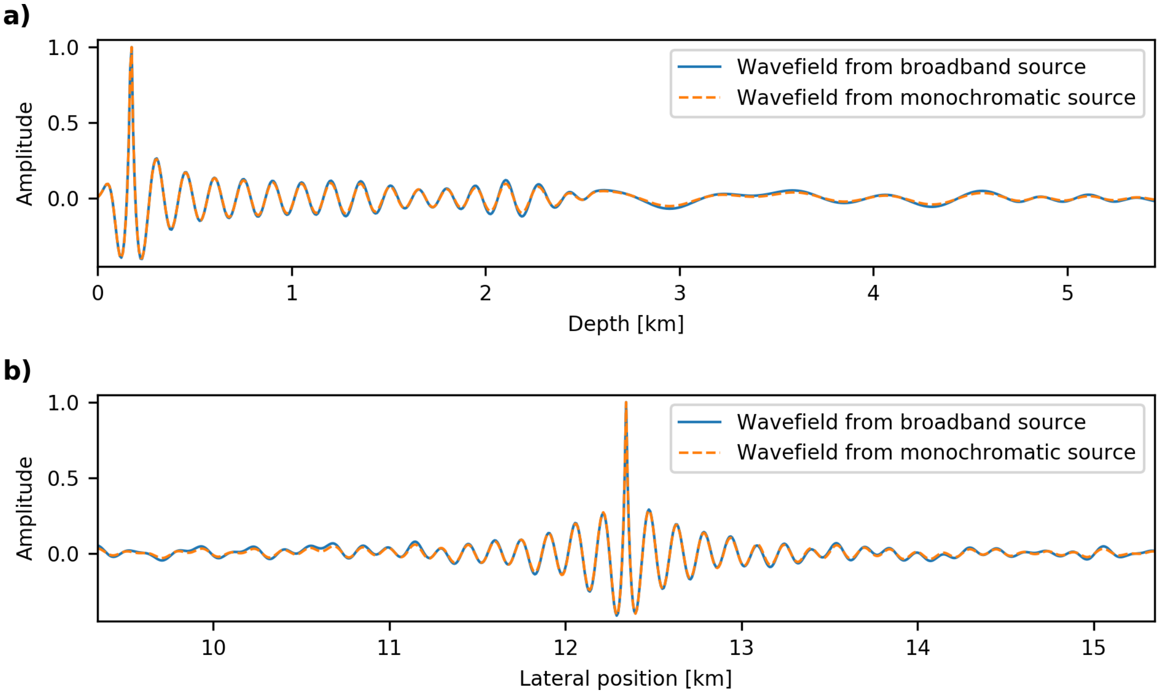

Figure 4: Comparison of monochromatic wavefields computed with on-the-fly DFTs using the full broadband source wavelet or the corresponding monochromatic source. The plots show vertical (a) and horizontal (b) slices of a monochromatic 10 Hz wavefield from the Sigsbee 2A model.

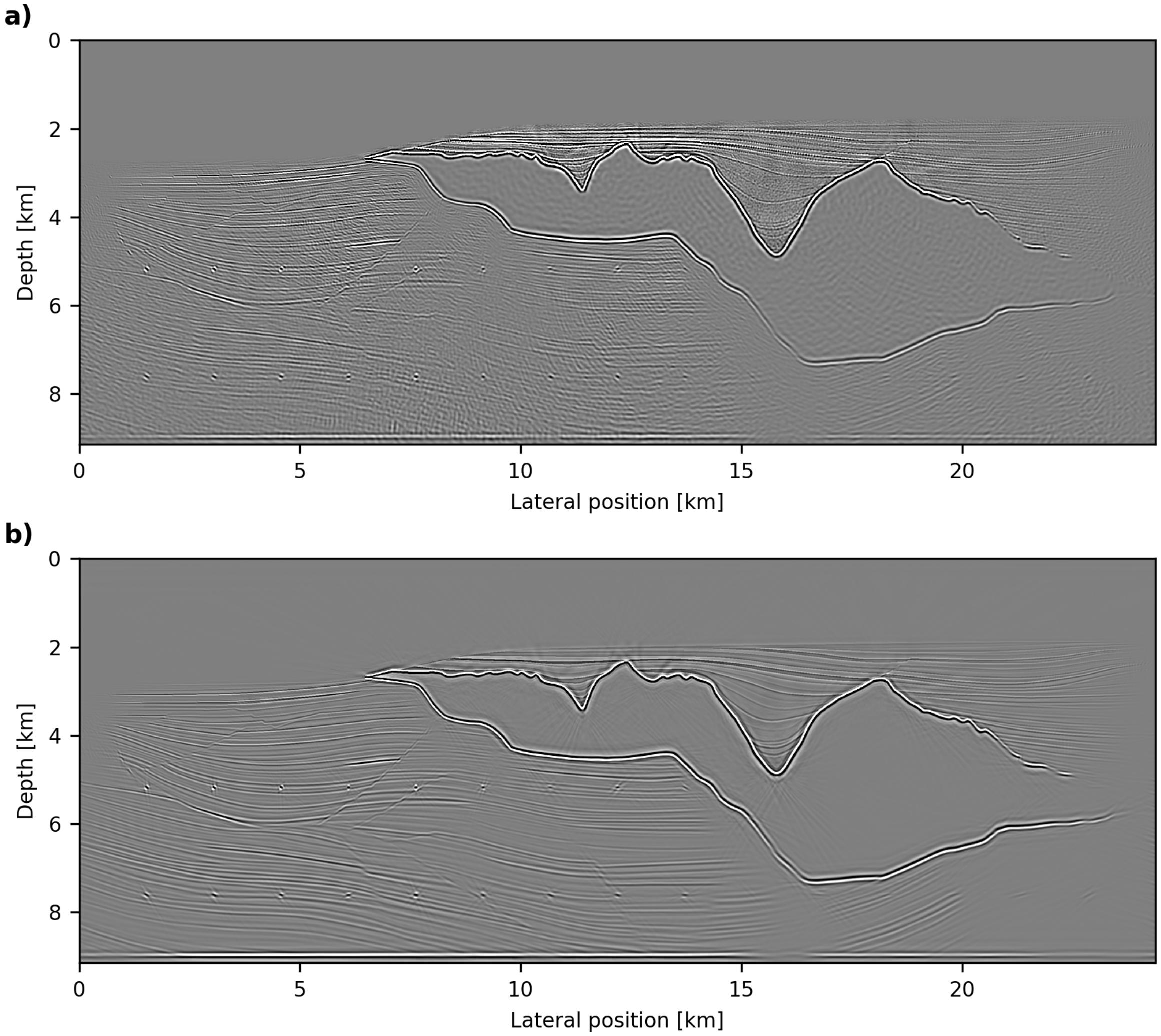

Figure 5: Reverse-time migration with 20 randomly selected frequencies per shot (a), in comparison to sparsity-promoting LS-RTM after 20 iterations, using 100 shots with 20 frequencies each per iteration (b). With only two passes through the full dataset, SPLS-RTM is able to remove the noise from frequency randomization, as well as the imprint of the source wavelet. The only post-processing that was applied to the results, is a linear depth scaling.

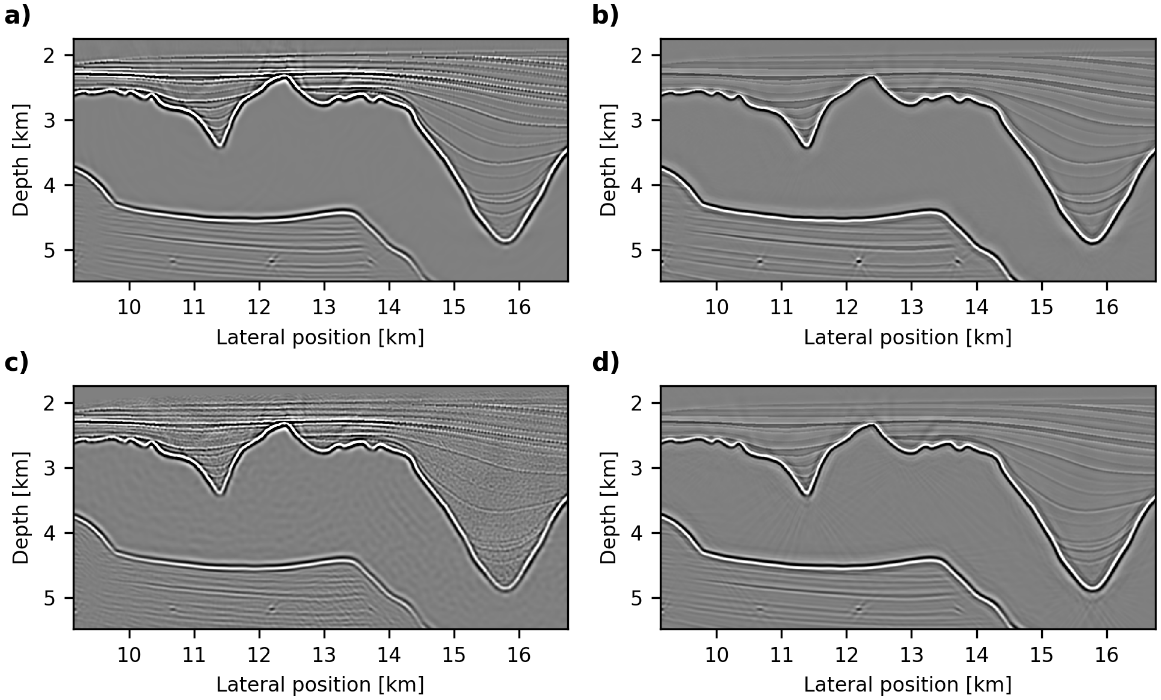

Figure 6: A close-up comparison of the images from time-domain RTM (a), time-domain SPLS-RTM (b), frequency-domain RTM (c) and frequency-domain SPLS-RTM (d). While RTM with on-the-fly Fourier transforms and randomized subset of frequencies leads to a noisy image, sparsity-promoting LS-RTM is able to convert noise to coherent reflectors and provide the same high-fidelity image as time-domain SPLS-RTM with optimal checkpointing. However, due to the limited number of iterations, not all energy is converted back into coherent energy, as apparent by the slightly weaker diffractors in (d). A comparison between the additional computational cost of checkpointing and on-the-fly DFTs is provided in the discussion.

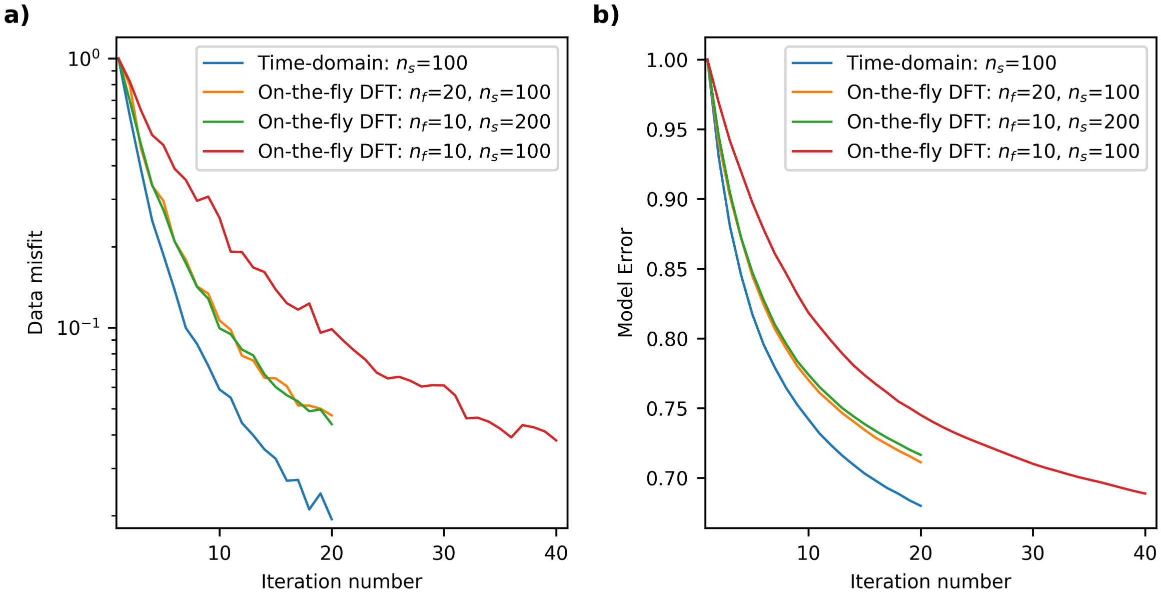

Figure 7: Normalized ℓ_2-norm data misfit (a) and ℓ_2-norm reconstruction error (b) for compressive LS-RTM in the time domain and with on-the-fly DFTs, using the linearized Bregman method. For a smaller number of frequencies n_f, more iterations have to be performed to reduce the data misfit to a comparable level, but each iteration requires less memory and computations. Keeping the product of the batch size of shots n_s and frequencies n_f constant, yields results of comparable quality.

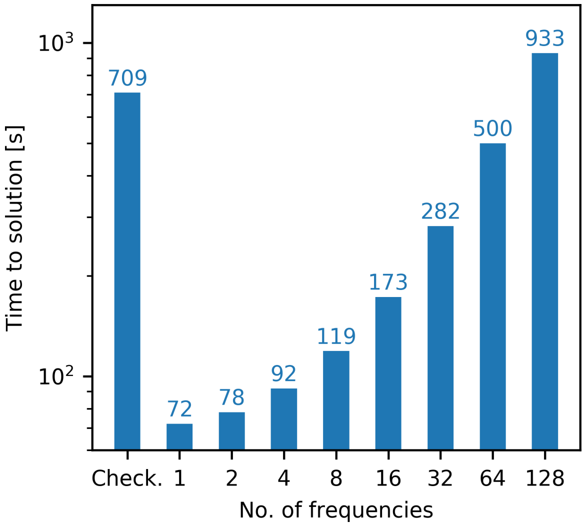

Figure 8: Timings for computing the gradient of the full Sigsbee model for a single shot record as a function of the number of frequencies and in comparison to optimal checkpointing (leftmost bar).

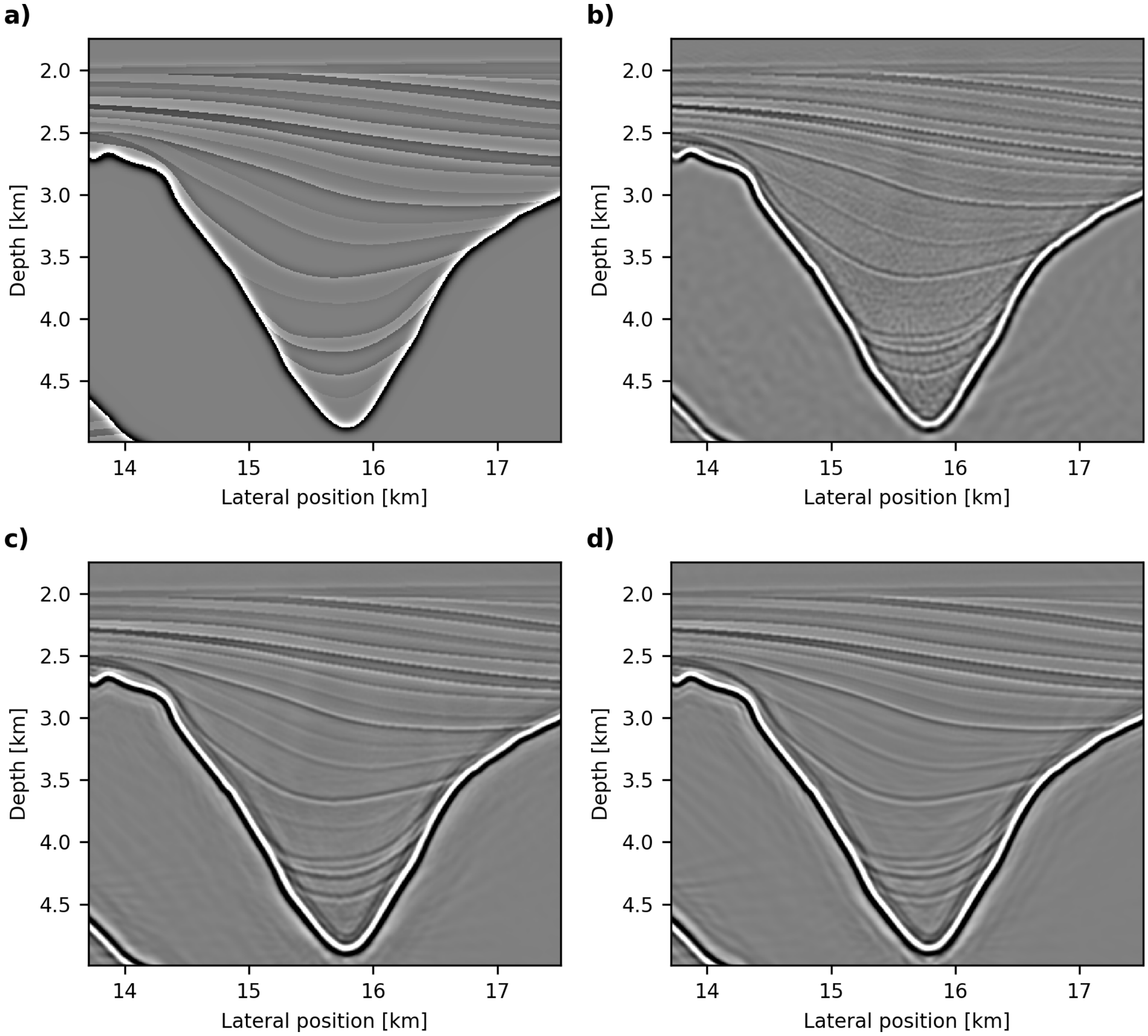

Figure 9: Close-up views of the top-of-salt region for different values of the regularization parameter λ. Figure (a) is the true image, (b) is the result for $\lambda = 0$ (no sparsity-promotion), (c) is the result for $\lambda = 1e-4$ and (d) for $\lambda = 4e-4$. Since the top-of-salt region is well illuminated, the resulting image is not very sensitive to the choice of the thresholding parameter and the effect of sparsity promotion is less apparent than in the sub-salt area.

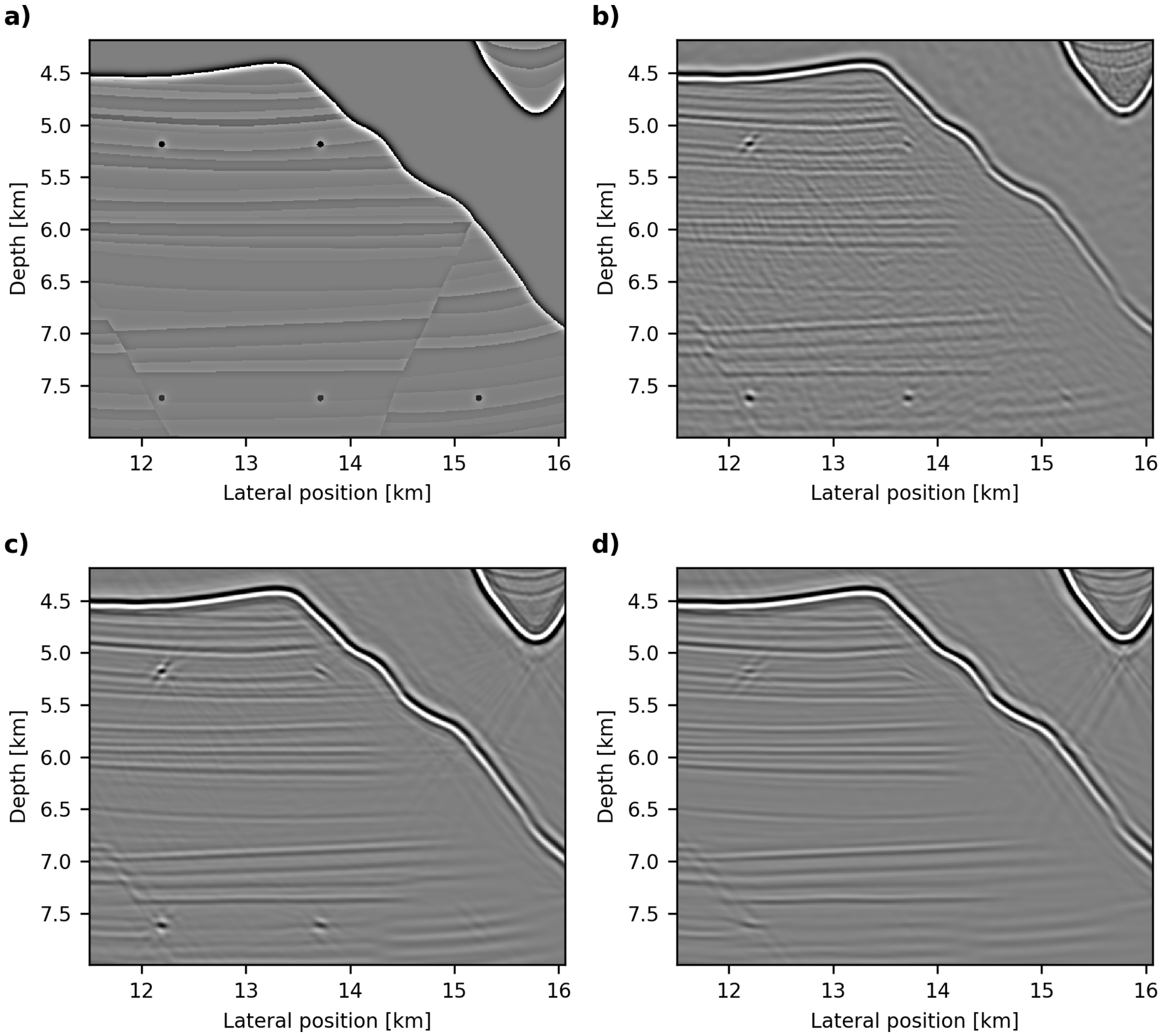

Figure 10: Close-up views of the sub-salt region of the true image (a) and results after 20 iterations of SPLS-RTM with the linearized Bregman method using $\lambda = 0$ (b), $\lambda = 1e - 4$ (c) and $\lambda = 4e - 4$ (d). Compared to the top-of-salt area, the benefit of sparisty-promotion is greater, but the result is also more sensitive to the choice of λ.

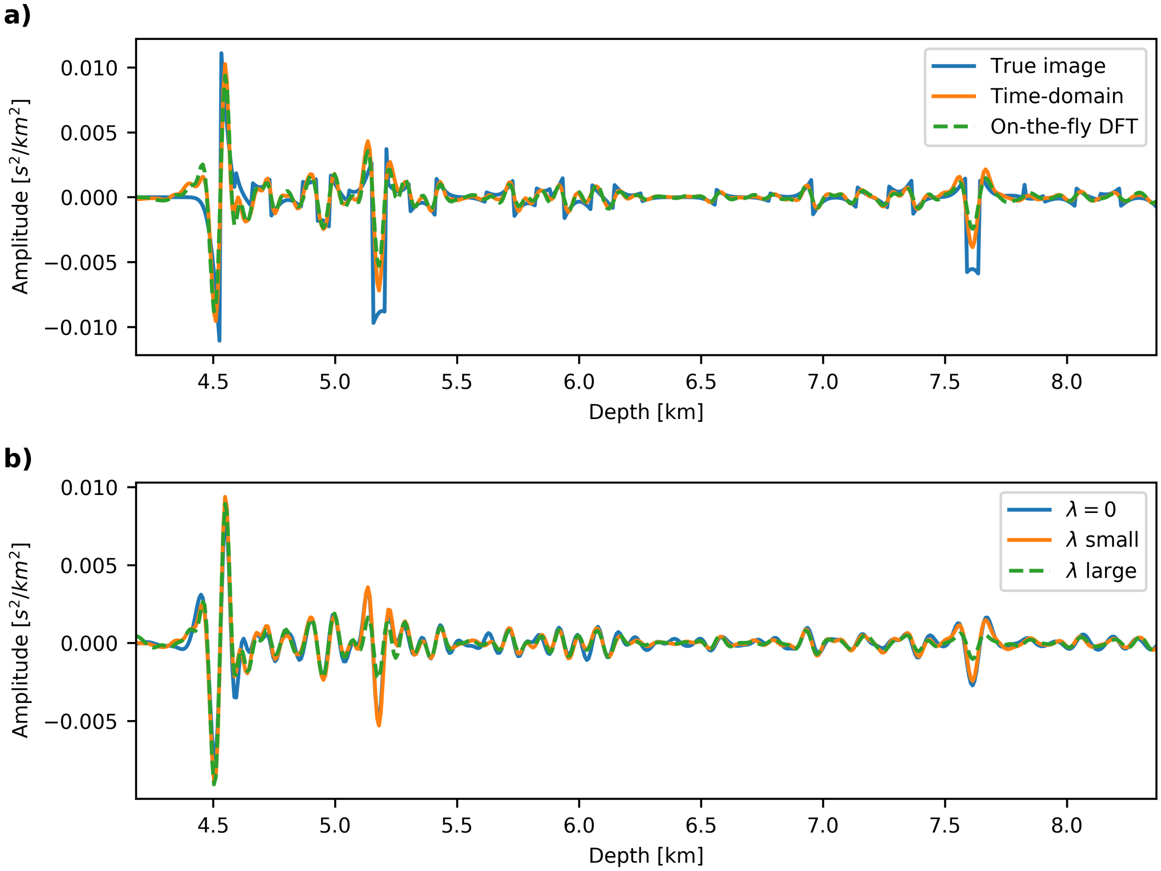

Figure 11: Trace comparisons of the results after 20 iterations of SPLS-RTM using time-domain modeling and on-the-fly DFTs (a) and for different values of λ (b).

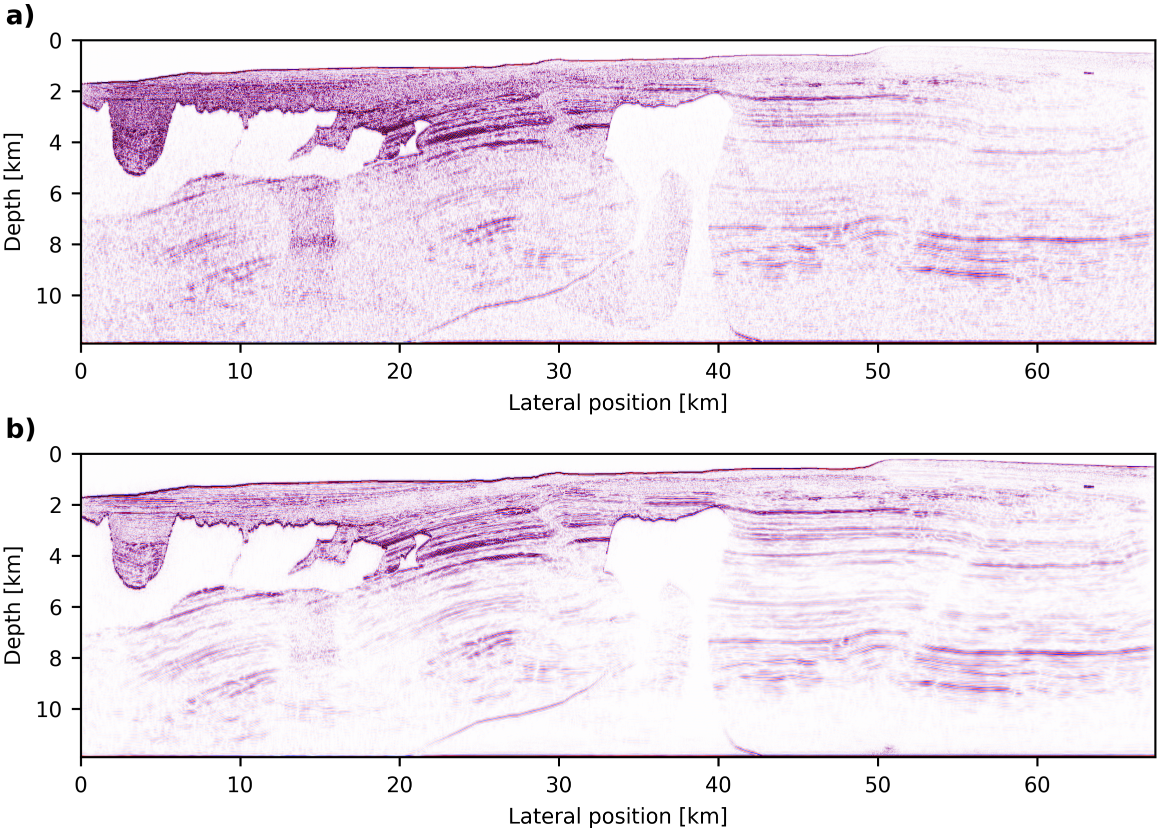

Figure 12: Comparisons of frequency-domain RTM (a) and SPLS-RTM (b) using on-the-fly Fourier transforms with 20 randomly selected frequencies per shot record. The SPLS-RTM image is shown after 20 iterations of the linearized Bregman method, using 200 random shots per iteration, which corresponds to three passes through the data.

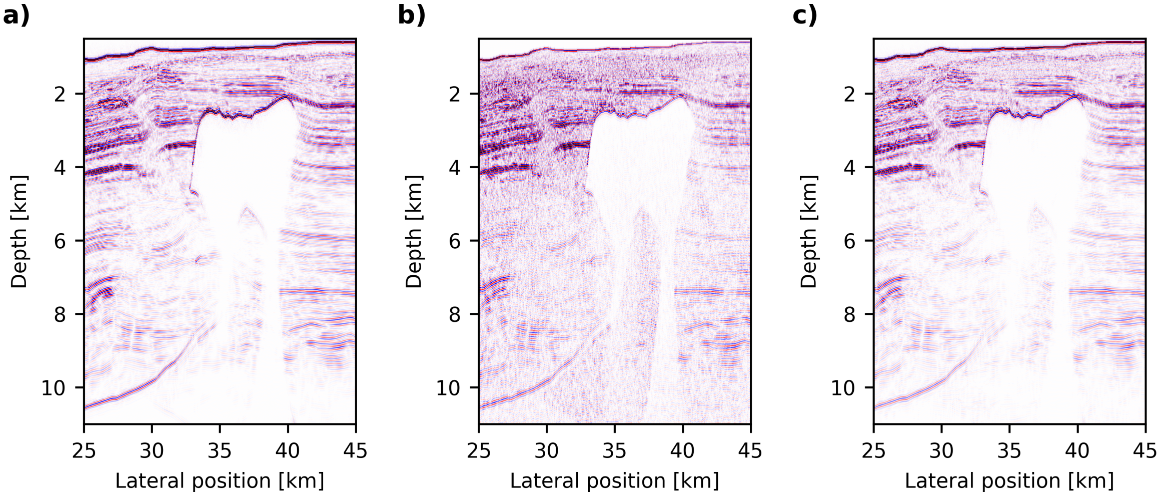

Figure 13: Close-up comparison of time-domain SPLS-RTM with optimal checkpointing (a), frequency-domain RTM (b) and frequency-domain SPLS-RTM (c). The results for frequency-domain RTM and SPLS-RTM were computed with 20 randomly selected frequencies per shot record. A depth scaling was applied to the RTM image, to make up for the lack of the depth-scaling pre-conditioner that was used for SPLS-RTM.

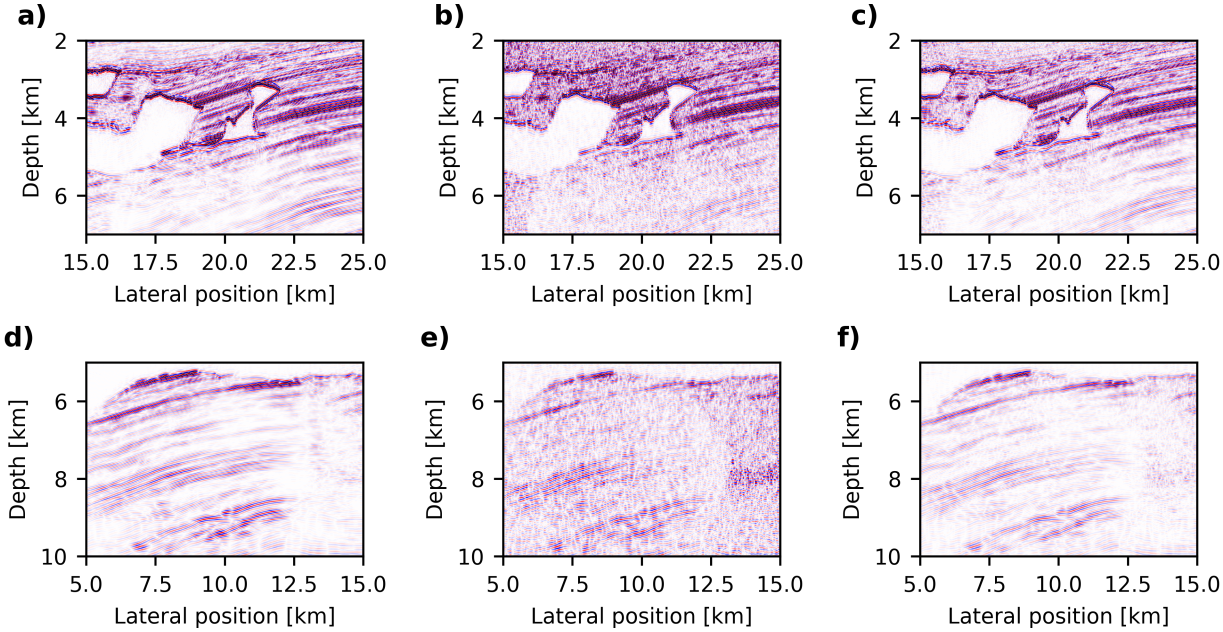

Figure 14: Close-up views of the time-domain SPLS-RTM result (a, d), frequency-domain RTM (b, e) and frequency-domain SPLS-RTM (c, f). The top row shows a shallow part of the image with good illumination in comparison to the sub-salt area, which is affected stronger by frequency subsampling (bottom row).

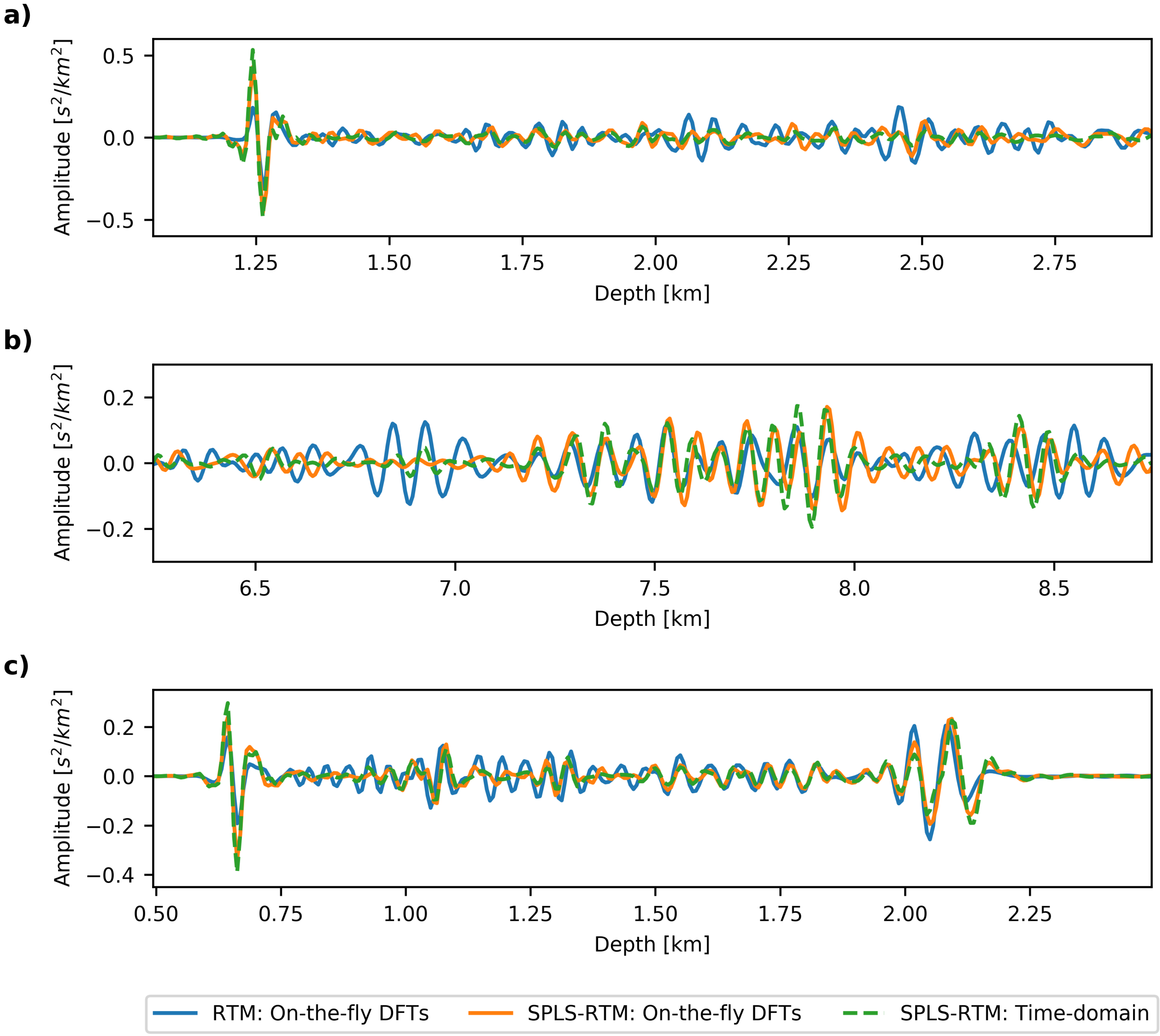

Figure 15: Trace comparisons of the imaging results at various locations of the model. Plots (a) and (b) are well-log plots of different depths at 10 km lateral position. The traces in (c) were extracted at 40 km lateral position.

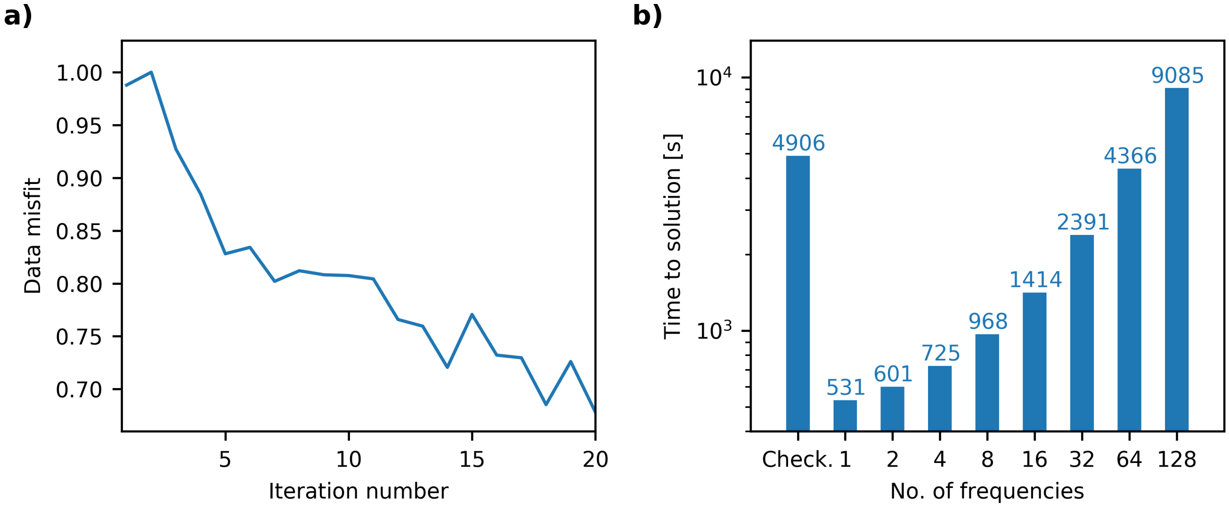

Figure 16: Relative data misfit for the current subset of shots during SPLS-RTM with on-the-fly DFTs (a) and timings for computing the gradient of the BP model for one shot record as a function of the number of frequencies (b). The bar on the left-hand side denotes the corresponding time-to-solution using optimal checkpointing.

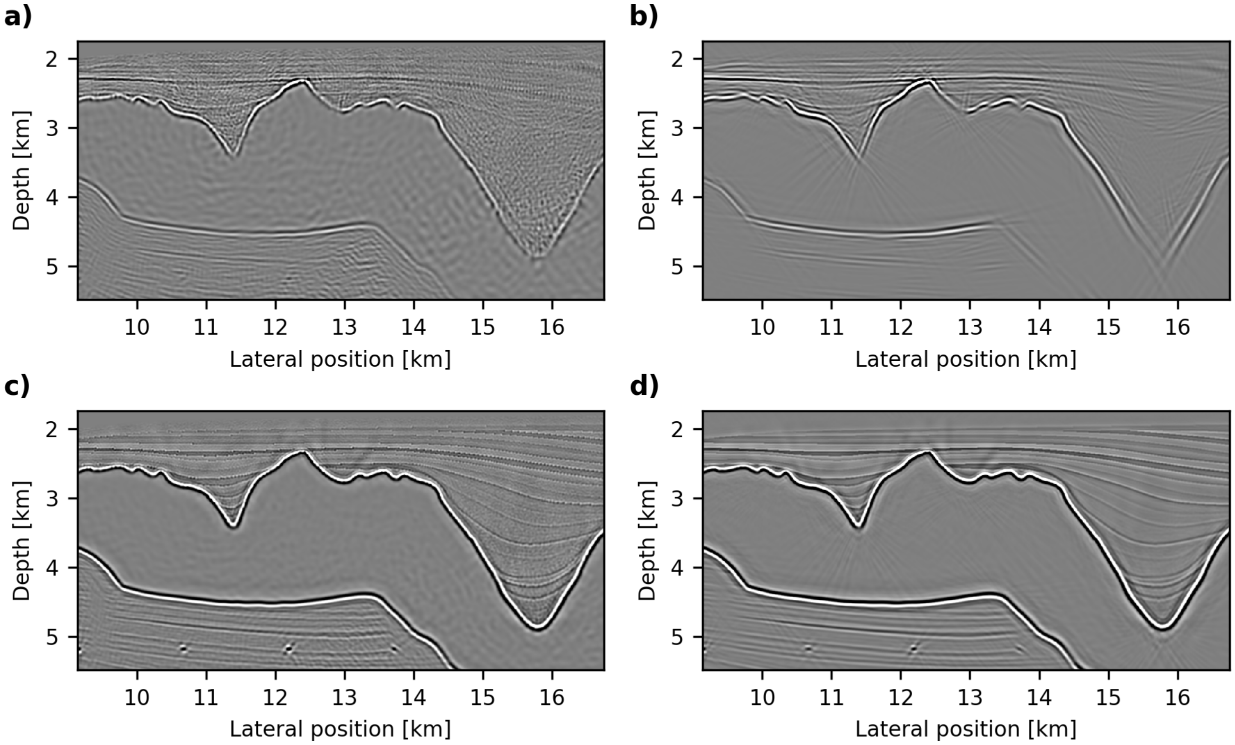

Figure 17: After the first iteration of the linearized Bregman method, the dual variable $\mathbf{z}$ (a) is a noisy version of the seismic image. The sparse primal variable $\mathbf{x}$ (b) is obtained by soft-thresholding of the curvelet coefficients of (a). In the early iterations, $\mathbf{x}$ contains only reflectors with the largest (curvelet) coefficients, but the smaller coefficients re-enter the solution in the subsequent iterations. After the final iteration, all reflectors have been restored in the primal variable (d), while the dual variable (c) is still noisy (but less than after the initial iteration).

LIST OF TABLES

Strategy	Memory	Additional cost
TD: save all wavefields	$\mathcal{O}(n_t)$	-
TD: optimal checkpointing	$\mathcal{O}(\log n_t)$	$\mathcal{O}(\log n_t)$
TD: boundary reconstruction	$\mathcal{O}(n_t)$	$\mathcal{O}(n_t)$
FD: on-the-fly DFT	$\mathcal{O}(n_f)$	$\mathcal{O}(n_f)$

Table 1: Asymptotic behaviour of memory requirements and additional computational cost for different strategies to compute adjoint-state gradients in the time domain (TD) and frequency domain (FD). The total number of model grid points in this analysis is assumed to be constant and is therefore excluded from the analysis. However, while reconstructing wavefields from the boundary scales linearly with the number of time steps, it requires substantially less memory than saving the full wavefield (i.e. the asymptotic behavior has a smaller constant). The analysis in this table holds for both 2D and 3D domains.