GEOPHYSICS, VOL. 87, NO. 6 (NOVEMBER-DECEMBER 2022); P. 1–19, 23 FIGS., 3 TABLES.
10.1190/GEO2021-0491.1

Least-squares reverse time migration via deep learning-based updating operators

Kristian Torres[1] and Mauricio Sacchi[1]

ABSTRACT

Two common issues of least-squares reverse time migration (LSRTM) consist of the many iterations required to produce substantial subsurface imaging improvements and the difficulty of choosing adequate regularization strategies with optimal hyperparameters. We investigate how supervised learning can mitigate these shortcomings by solving the LSRTM problem through an iterative deep learning framework inspired by the projected gradient descent algorithm. In particular, we develop an image-to-image approach interlacing the gradient steps at each iteration with blocks of residual convolutional neural networks (CNNs) that capture the prior information in the training phase. By including the least-squares data-misfit gradient into the learning process, we force the solution to comply with the observed seismic data, while the CNN projections implicitly account for the regularization effects that lead to high-resolution reflectivity updates. After training with 900 randomly generated instances, our network ensemble can estimate accurate reflectivity distributions in only a few iterations. To demonstrate the effectiveness and generalization properties of the method, we consider three synthetic cases: a folded and faulted model, the Marmousi model, and the Sigsbee2a model. We empirically find that it is possible to obtain an improved reflectivity model for out-of-distribution instances by using the learned reconstructions as warm starts for the conjugate gradient algorithm and bridging the gap between the learned and conventional LSRTM schemes. Finally, we apply the proposed network with transfer learning on a 2D towed-streamer Gulf of Mexico field data set, producing high-resolution images comparable with traditional LSRTM but drastically reducing the required number of iterations.

INTRODUCTION

Seismic migration workflows are the cornerstone of subsurface structural imaging. From the family of migration techniques implemented in such workflows, reverse time migration (RTM) (Baysal et al., 1983; Whitmore, 1983) is considered as the most effective algorithm for imaging the complex geologic settings. Its superiority in providing accurate subsurface images stems from the fact that it uses the adjoint of the linearized two-way wave equation, which is a more accurate approximation of the wave-propagation phenomena than its asymptotic or one-way counterparts, thus imposing no limitations on the dip angle of reflectors (Etgen et al., 2009). Despite being a sophisticated migration method, RTM images often suffer from backscattering low-frequency noise (Díaz and Sava, 2016); incomplete illumination due to defocusing velocity anomalies (Buur and Kühnel, 2008); and artifacts produced by a coarse source-receiver sampling and corrupted data (Zhang and Sun, 2009), thereby exposing the limitations of using adjoint (rather than inverse) operators to map from seismic data to reflectivity models.

Least-squares migration (LSM) (Lailly and Bednar, 1983; Nemeth et al., 1999) seeks to overcome the limitations of adjoint formulations by posing the seismic migration as a linear inversion problem, in which a generalized inverse approximates the exact inverse of the modeling operator. Under the assumption of a sufficiently correct background velocity model and adequate preconditioning of the data, LSM can reduce migration artifacts, remove acquisition marks, compensate for illumination, and increase the resolution of seismic sections (Lambaré et al., 1992; Kühl and Sacchi, 2003; Schuster, 2017). When LSM uses the RTM engine to perform demigration/migration sequences for imaging, we call it least-squares RTM (LSRTM) (Dai et al., 2012; Wang et al., 2017). LSRTM delivers true amplitude images and sharpens subsurface reflectors (Dong et al., 2012).

Conventional LSRTM implementations use iterative methods to obtain a reflectivity model that minimizes the l_2-norm misfit,

Manuscript received by the Editor 28 July 2021; revised manuscript received 10 June 2022; published ahead of production 3 August 2022.
[1]University of Alberta, Department of Physics, Edmonton, Canada. E-mail: torresba@ualberta.ca (corresponding author); msacchi@ualberta.ca.

defined in either the data domain, where the input is the recorded data, or the image domain, where the input is the migrated image (Fletcher et al., 2016; Schuster and Liu, 2019). In addition, due to the ill-posed nature of the LSM problem, a regularization term is frequently incorporated into the misfit to impose a priori information on the model (such as sparseness or other structure constraints) and stabilize the inverted solution (e.g., Clapp, 2005; Wang et al., 2005; Wang and Sacchi, 2007, 2009; Dutta, 2017; Witte et al., 2017; Li et al., 2020a). Regularized LSRTM methods often produce acceptable results but can be challenging to deploy in practice due to the difficulty of hyperparameter selection and the high computational cost of iteratively evaluating the forward and adjoint operators. Moreover, the selection and design of an appropriate regularization term remain highly nontrivial for complex reflectivity distributions, where a poor choice could compromise the correctness of the inversion. Such limitations open the way to a new generation of methods.

With the recent advent of deep learning (LeCun et al., 2015), machine learning techniques have emerged as powerful alternatives for solving the seismic inverse problems with remarkable empirical success. One common approach in the field of velocity model building is to use deep neural networks (DNNs) to directly approximate the data-to-model inverse mapping in an end-to-end fashion (e.g., Araya-Polo et al., 2018, 2019; Yang and Ma, 2019; Li et al., 2020b; Liu et al., 2021a; Vantassel et al., 2021). Although this approach circumvents the need for iterative reconstruction, it relies exclusively on vast amounts of sometimes elusive training data to achieve high-quality results and good generalization properties. It also scales poorly for models with a large number of trainable parameters due to memory limitations — an issue analogous to explicit matrix-based formulations of seismic imaging problems (Yao and Jakubowicz, 2016).

For these reasons, alternative strategies aim to combine data-driven applications with traditional inversion processes, often by substituting parts of the original computations or by adopting an unrolled implementation of an iterative algorithm. For instance, a category of deep learning methods bypasses the need for labeled data sets by using the partial differential equation kernels to guide the unsupervised training of physics-based neural networks (PBNNs), in which the seismic records are regarded as the training data and the subsurface parameters are estimated through convolutional neural network (CNN) (Biswas et al., 2019) or recurrent (Richardson, 2018; Sun et al., 2020) neural network. Alfarraj and AlRegib (2019) and Sun et al. (2021) extend PBNN to a hybrid approach that considers a weighted sum between model and data misfits as the components of the loss function, with the ability to learn from partially labeled data in semisupervised training. These techniques highlight the benefits of integrating machine learning operations with known deterministic physical models in the form of forward and backward wave-propagation codes.

In recent years, deep learning also has been implemented into LSM workflows. Liu et al. (2020b) show that the iterative solution of the image-domain LSM problem with sparsity constraints resembles a forward pass through a multilayered neural network, establishing analogies between the filters and feature maps of a CNN and the migration Green's functions and reflectivity coefficients. Kaur et al. (2020) approximate the effect of the inverse Hessian on poststack-migrated images by training a generative adversarial network using the background velocity model as a conditioner. Most recently, Vamaraju et al. (2021) use mini-batch gradients and adaptive learning rate optimizers to improve the resolution and reduce the computational burden of conventional prestack LSRTM. Further implementations of learning-based techniques in seismic imaging include using a CNN to suppress crosstalk artifacts from dip-angle gathers in elastic RTM images (Lu et al., 2020) and training CNNs as structural (Cheng et al., 2020) or denoising (Liu et al., 2020a) preconditioners to assist in the LSM inversion. For a comprehensive review of deep learning methods in other geophysical applications, we refer the reader to Yu and Ma (2021) and Adler et al. (2021).

Exploiting the universal approximation property of CNNs (Hornik et al., 1989) and the fact that they can learn structural features of images from representative examples, we introduce a supervised strategy that interfaces CNN-based machine learning with prestack LSRTM, which we have baptized as deep LSRTM. This scheme can be regarded as a deep learning extension of the conventional projected gradient-descent method because it substitutes the projection operators with sets of CNNs, learning an update function at each iteration. Our approach incorporates the linearized modeling and migration wave operators into the reconstruction process, evolving in response to the data-misfit gradient and allowing for interpretability in the network design. By detaching the forward and adjoint mappings from the learning step, deep LSRTM favorably reduces the networks' parameter complexity and the amount of training data and also improves the robustness and generalizability (Boink et al., 2019; Maier et al., 2019). The efficiency of the method is twofold: once trained, the proposed networks can predict accurate reflectivity models within only a few iterations, and second, it implicitly learns the step size and the effect of regularization from the training data. We emphasize that Chen et al. (2021) explore a similar idea based on the learning generalized projection mappings in the context of blind high-resolution inversion of poststack data; however, to the best of our knowledge, this is the first fully iterative application of learned updating projections on seismic prestack LSRTM.

In the following sections, we will first describe how the theory of constrained LSRTM connects with deep LSRTM. Then, we will present the proposed network architecture and the supervised training procedure, where we use 900 reflectivity models derived from folded and heavily faulted pseudorandom velocity distributions as true labels. After the training, we evaluate the performance of deep LSRTM on three synthetic examples and compare it against a single-step learned reconstruction and a conventional LSRTM solved by the conjugate gradient least-squares (CGLS) algorithm (Hestenes and Stiefel, 1952). We also use field data from the Gulf of Mexico (GOM) to test how a direct implementation of our method and a modification that includes transfer learning perform on a data set with characteristics not perfectly represented by the training samples. We observe that our workflow produces high-resolution migration images comparable with traditional LSRTM but drastically reduces the required number of iterations. Finally, we present the conclusions and directions for future work.

METHOD

Least-squares reverse time migration

The forward problem of seismic imaging can be represented in a compact matrix notation by

$$\mathbf{d} = \mathbf{L}\mathbf{m}, \tag{1}$$

where $\mathbf{d}$ denotes the $M \times 1$ vector of modeled data, $\mathbf{m}$ is the $N \times 1$ vectorized reflectivity or model perturbation, and $\mathbf{L}$ is the $M \times N$ matrix that describes a linearized forward-modeling operator dependent on the acquisition geometry, the source wavelet, and a known background model of the medium (e.g., velocity) (Tarantola, 1984). In this work, $\mathbf{L}$ defines the forward (or demigration) operator that encapsulates the first Born approximation of the acoustic two-way wave equation for a medium without density variations (Lailly and Bednar, 1983). Theoretically, RTM uses the adjoint of the Born-modeling operator to migrate the preprocessed observed seismic data $\mathbf{d}_{\text{obs}}$:

$$\mathbf{m}_{\text{mig}} = \mathbf{L}^T \mathbf{d}_{\text{obs}}, \tag{2}$$

where $\mathbf{m}_{\text{mig}}$ is the migrated image, an estimate of the true model perturbation (Xu and Sacchi, 2018). Seismic imaging is plagued by noisy data and a nontrivial null space due to bandwidth, illumination, and offset limitations. Thus, LSRTM is formulated as a linear ill-posed inverse problem aided by regularization. Regularized LSRTM entails finding the reflectivity model that minimizes the mismatch between the observed and modeled data in a least-squares sense and is consistent with the available prior knowledge. This amounts to solving the following unconstrained optimization problem:

$$\min_{\mathbf{m}} \left\{ J(\mathbf{m}) + \lambda R(\mathbf{m}) = \frac{1}{2} \|\mathbf{L}\mathbf{m} - \mathbf{d}_{\text{obs}}\|_2^2 + \lambda R(\mathbf{m}) \right\}, \tag{3}$$

where $J(\mathbf{m})$ is the data-misfit function; $\|\cdot\|_2$ is the l_2 norm that implicitly designates summation over time samples, receivers, and sources; $R(\mathbf{m})$ represents any generic regularization/penalty term that encodes knowledge about the structure of the model, and $\lambda > 0$ is the damping parameter that controls the trade-off between the data misfit and the regularization term. Depending on the smoothness and convexity properties of the regularizer, a myriad of specialized optimization methods have been developed to obtain optimal solutions to different variants of equation 3. For instance, a common strategy consists of minimizing a nonquadratic norm on some sparsity-promoting representation of the reflectivity model or its gradient, which can be solved with proximal algorithms or the iterative-reweighted least-squares method (Wang and Sacchi, 2007; Liu et al., 2021b). Although the nonquadratic regularization approaches lead to high-resolution images, they still require many iterations, and using large regularization weights might result in excessive smoothing and underestimated property values. Obtaining an optimal λ also is a cumbersome task and it is generally estimated by a trial-and-error approach. The more rigorous L-curve procedure is not feasible for industrial-scale LSRTM problems because it demands the solution of the inverse problem multiple times (Calvetti et al., 2000).

In the particular case of $R(\mathbf{m}) = (1/2)\|\mathbf{m}\|_2^2$ (i.e., assuming zero-order quadratic regularization), equation 3 leads to the classical LSRTM with damping. The latter has a closed-form solution given by

$$\mathbf{m} = [\mathbf{L}^T\mathbf{L} + \lambda \mathbf{I}]^{-1} \mathbf{L}^T \mathbf{d}_{\text{obs}}, \tag{4}$$

which is equivalent to the image-domain LSRTM formulation:

$$\mathbf{m} = \mathbf{H}^{-1} \mathbf{m}_{\text{mig}}. \tag{5}$$

Due to the prohibitive cost of explicitly inverting and storing the Hessian matrix plus prewhitening given by $\mathbf{H} = \mathbf{L}^T\mathbf{L} + \lambda \mathbf{I}$, the operators $\mathbf{L}$ and $\mathbf{L}^T$ are constructed in implicit form rather than as matrices, and equation 4 is solved iteratively or by approximating the Hessian by a manageable size operator (Gao et al., 2020). Considering a simple gradient descent scheme, we can calculate reflectivity updates by

$$\mathbf{m}_{k+1} = \mathbf{m}_k - \alpha \nabla (J(\mathbf{m}_k) + \lambda \mathbf{m}_k), \tag{6}$$

where

$$\nabla J(\mathbf{m}_k) = \mathbf{L}^T(\mathbf{L}\mathbf{m}_k - \mathbf{d}_{\text{obs}}) \quad = \mathbf{L}^T \mathbf{r} \tag{7}$$

denotes the gradient of the data misfit with respect to the model parameters, $\mathbf{r}$ is the data residual, $\alpha > 0$ is the step length, and k is the iteration number. In practice, we can find stable solutions to the LSRTM problem with the more efficient CGLS algorithm, which allows us to use an analytical step length, weighted l_2 norms, and quadratic regularization (Kühl and Sacchi, 2003).

Constrained LSRTM via gradient projections

A different approach to regularized minimization arises when the prior knowledge is introduced in the form of projected constraints (Peters et al., 2019). In such a case, the inverse problem can be rephrased as

$$\min_{\mathbf{m} \in \mathcal{C}} \left\{ J(\mathbf{m}) = \frac{1}{2} \|\mathbf{L}\mathbf{m} - \mathbf{d}_{\text{obs}}\|_2^2 \right\}, \tag{8}$$

where $\mathcal{C} \subset \mathbb{R}^N$ is a closed nonempty convex set with the desired physical constraints. When an efficient method for projecting onto $\mathcal{C}$ is available, we can use the projected gradient descent (PGD) method as a cost-effective algorithm to iteratively solve equation 8:

$$\mathbf{m}_{k+1} = \mathcal{P}_{\mathcal{C}}(\mathbf{m}_k - \alpha \nabla J(\mathbf{m}_k)), \tag{9}$$

with the projection operator formally defined by

$$\mathcal{P}_{\mathcal{C}}(\mathbf{x}) = \arg\min_{\mathbf{m} \in \mathcal{C}} \left\{ \frac{1}{2} \|\mathbf{m} - \mathbf{x}\|_2^2 \right\}, \tag{10}$$

which translates into finding the closest $\mathbf{m} \in \mathcal{C}$ to a given input model update $\mathbf{x}$. In practice, a variety of techniques might be used as projections. For example, Cheng et al. (2016) use PGD with the singular spectrum analysis filter (Oropeza and Sacchi, 2011) as the projection operator to perform prestack LSM for blended data, where a deblending constraint is imposed in the shot-index common-image domain to separate the coherent multiples from source crosstalk.

Because PGD converges at a linear rate under a constant step size, authors have explored the application of the projected gradient method to seismic waveform inversion (e.g., Becker et al., 2015; Xiang et al., 2016; Peters et al., 2019). In general, convergence properties of PGD are given for convex projections. However, as pointed out by Peters et al. (2019), one can use nonconvex projections and rely on empirical testing to examine convergence properties.

Deep LSRTM: LSRTM via learned updates

We build the deep-LSRTM framework upon recent research that suggests that unrolling classical model-based techniques with the learned projection networks trained from data may significantly optimize inverse problem formulations (e.g., Gregor and LeCun, 2010; Chang et al., 2017; Meinhardt et al., 2017; Putzky and Welling, 2017; Zhang and Ghanem, 2017). In general, learned projection formulations avoid the explicit computation of regularized constraints. Instead, neural network architectures with K prescribed blocks are constructed, where each block mimics one iteration of a model-based gradient method. Naturally, the PGD scheme presented in the previous section inspires us to replace the projection operator $\mathcal{P}_{\mathcal{C}}(\cdot)$ or any generalized projection that improves the current model update by a neural network $\mathcal{P}_{\Theta}(\cdot)$, parameterized by the set of trainable weights $\Theta = [\theta_0, \ldots, \theta_{K-1}]$. With this analogy established, we arrive at our deep-LSRTM formulation, which has appeared in the nongeophysical literature (Adler and Öktem, 2017; Hauptmann et al., 2018). Specifically, we modify the iterative solution of the PGD method (equation 9) with CNN updates, designed to encode the prior knowledge represented by the complex distribution of a training data manifold. We choose a CNN-based architecture because the convolution kernels and weight-sharing structures allow the learned operators to exploit spatial correlations, naturally enforcing a regularization effect.

Deep LSRTM iteratively estimates reflectivity updates using the following expression:

$$\mathbf{m}_{k+1} = \mathcal{P}_{\theta_k}(\mathbf{m}_k, \nabla J(\mathbf{m}_k)), \quad k = 0, \ldots, K-1, \tag{11}$$

where each $\mathcal{P}_{\theta_k} \subset \mathcal{P}_{\Theta} : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}^N$ is an updating function trained to map the pair of inputs $(\mathbf{m}_k, \nabla J(\mathbf{m}_k))$ to a single model update $\mathbf{m}_{k+1}$ that has characteristics similar to the training instances. Passing through the multiple set of blocks $\mathcal{P}_{\Theta}$ is analogous to executing the PGD algorithm with a finite number of K iterations. As can be inferred from equation 11, we let the updating operators learn how to combine the current model iterate with the gradient of the data-misfit term instead of explicitly computing a step length. The model features also are implicitly learned from the data set during training rather than imposed as constraints or regularizing terms. This is a differentiating factor compared to traditional LSRTM, where it often is complicated to choose optimal projection or regularization strategies capable of expressing desirable geologic and structural elements. Deep LSRTM also avoids introducing model-dependent tuning parameters.

Compactly, we define each updating operator as N stacked 2D convolutional layers and nonlinear activation functions (see Appendix A). In addition, we incorporate concatenation operators, residual connections, and batch normalization (BN) (Ioffe and Szegedy, 2015) to increase the expressivity of the network and to help gradient propagation in the training phase.

Figure 1 illustrates the first two iterations of the deep-LSRTM framework. Different from end-to-end implementations, our approach does not need to learn the forward or adjoint operators. This would add complexity in terms of learnable parameters and poses hardware difficulties when working with large images. Instead, the wave operators are computed through the data-misfit gradient $\nabla J(\mathbf{m}_k)$ using equation 7 before feeding each updating block. This allows using shallower network architectures, which are less prone to overfitting and require less training data. Moreover, as empirically showed in Adler and Öktem (2017) and Putzky and Welling (2017), interweaving explicitly known operators in learned reconstruction schemes provide more detail to the solution and reduce the training error. The information passed through the data-misfit gradient indicates where the seismic image needs improvement to fit the observed data, progressively refining the model at each iteration (Zeng et al., 2017). This means that, within the limitations of operator mismatches, $\nabla J(\mathbf{m})$ partially reconstructs the measurable components, whereas $\mathcal{P}_{\Theta}$ attempts to recover the null space components (limited aperture and missing frequencies) similar to the regularizer in classical optimization techniques.

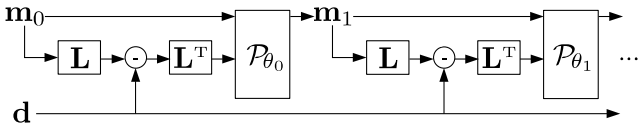


Figure 1. Unrolled diagram of the implemented iterative learned network for the first two iterations.

In our approach, the inputs contain artifacts and illumination deficiencies. The admissible model update is the one that has reduced artifacts and boosted resolution. Hence, we train the network parameters to make efficient projections by minimizing a loss function between the predictions and the training data set. Typically, in small-scale deep learning inversions, all the parameters of unrolled schemes are trained jointly for all iterations (i.e., end-to-end). This provides an optimal set $\hat{\Theta}$ for the predefined maximum number of K iterations (Monga et al., 2021), provided that the ground-truth solutions to the optimization problem are available. Considering the mean-squared error (MSE) loss, this yields the following optimization problem:

$$\begin{aligned}\hat{\Theta} &= \arg\min_{\Theta} \frac{1}{\mathfrak{I}} \sum_{i=1}^{\mathfrak{I}} \|\mathbf{m}_K^i - \mathbf{m}_{\text{true}}^i\|_2^2, \\ &= \arg\min_{\theta_0,\ldots,\theta_{K-1}} \frac{1}{\mathfrak{I}} \sum_{i=1}^{\mathfrak{I}} \|(\mathcal{P}_{\theta_{K-1}} \circ \ldots \circ \mathcal{P}_{\theta_0}(\mathbf{m}_0^i, \nabla J(\mathbf{m}_0^i))) - \mathbf{m}_{\text{true}}^i\|_2^2,\end{aligned} \tag{12}$$

where $\mathfrak{I}$ is the total number of training instances and $\mathbf{m}_{\text{true}}^i$ indicates the ground-truth model of the ith training instance. To solve equation 12, stochastic gradient methods commonly used in machine learning compute an approximation of the MSE loss gradient with respect to the network parameters. For large-scale problems such as deep LSRTM, this approach is unfeasible because the loss gradient also requires the computation of $\nabla J(\mathbf{m}_k)$ for each evaluation of the loss function (i.e., multiple calls of $\mathbf{L}$ and $\mathbf{L}^T$ per training instance). To make deep LSRTM computationally feasible, we adopt a greedy training approach (Hauptmann et al., 2018), which is grounded in optimization but produces a solution that is only iterate-wise optimal. The greedy training defines a minimization problem for each individual set of parameters θ_k, preventing the gradients from flowing between the updating blocks by optimizing

$$\begin{aligned}&\min_{\theta_k} \frac{1}{\mathfrak{I}} \sum_{i=1}^{\mathfrak{I}} \|\mathbf{m}_{k+1}^i - \mathbf{m}_{\text{true}}^i\|_2^2 \\ &= \min_{\theta_k} \frac{1}{\mathfrak{I}} \sum_{i=1}^{\mathfrak{I}} \|\mathcal{P}_{\theta_k}(\mathbf{m}_k^i, \nabla \tilde{J}(\mathbf{m}_k^i)) - \mathbf{m}_{\text{true}}^i\|_2^2.\end{aligned} \tag{13}$$

Compared with equation 12, equation 13 only requires one call of $\mathbf{L}$ and $\mathbf{L}^T$ per training instance at each iteration k. In this framework, to increase network capacity, each $\mathcal{P}_{\theta_k}$ shares the same architecture as the other updating operators, but each operator has its own set of learned parameters. The updating operators are trained sequentially, and the parameters are fixed after training, meaning that the outputs predicted by an updating block $\mathcal{P}_{\theta_{k-1}}$ from a previous iteration will be part of the data set used to train $\mathcal{P}_{\theta_k}$ in the next iteration. Finally, to gain substantial improvements in the imaging step, we substitute the raw data-misfit gradient with a preconditioned version (Xu and Sacchi, 2018), given by

$$\nabla\tilde{J}(\mathbf{m}_k) = \mathbf{D}\mathbf{P}_m\mathbf{L}^T\mathbf{P}_d\mathbf{r}, \tag{14}$$

where $\mathbf{P}_m$ is the illumination compensation operator that includes diagonal weights proportional to the inverse of the source-side wavefield, $\mathbf{P}_d$ is a linear filter that eliminates the diving wave energy from the data residuals, and $\mathbf{D}$ represents the 2D Laplacian filter, which removes low-frequency noise from the gradient.

Network design

We propose to assemble each updating operator $\mathcal{P}_{\theta_k}$ with the network architecture shown in Figure 2, following an encoder-decoder sequence.

The encoding part uses two consecutive convolutional layers with a rectified linear unit (ReLU) function to extract features from the two input branches containing $\mathbf{m}_k$ and $\nabla J(\mathbf{m}_k)$. These branches do not share intermediary layers until they are merged by a concatenation operator, which fuses the independent feature maps on the channel axis, followed by a BN layer. BN along the channel axis is needed to equalize contributions from the merged branches because the feature maps extracted from $\mathbf{m}_k$ have a different scale than those from $\nabla J(\mathbf{m}_k)$. This is crucial for successful training as it avoids the vanishing gradient problem in backward propagation (Ioffe and Szegedy, 2015). The encoder step ends with another convolutional layer that extracts information from the fused feature maps. Subsequently, the decoder part reassembles the feature maps by gradually decreasing the number of channels. The last convolutional layer does not use a ReLU function as the reflectivity updates can be positive or negative. We also add a skip connection between the input layer containing $\mathbf{m}_k$ and the output layer, forcing the network to learn residual updates (i.e., it learns to compute an update perturbation to the initial reconstruction). Because there is a high structural correlation between inputs and output, learning a residual mapping seems intuitively easier than learning the direct mapping (He et al., 2016).

In more detail, we use convolutional kernels of size 3×3 for all the convolutional layers. For the first iteration, we initialize the parameters following He et al. (2015). For subsequent iterations, while there is no sharing of parameters across blocks, we use the trained weights of the previous iteration to initialize the weights of the next iteration to speed up the training process. All the layers use a stride of size one and are zero padded such that all feature maps have the same size as the inputs. We obtain an optimal hyperparameter setting using a simple trial and error procedure. Different network components (such as filter size, number of filters, type of activation function, batch size, and learning rate) are changed, and the final network architecture is determined based on the validation data set performance. We stress that these settings rely purely on empirical experimentation without any robust mathematical proof; thus, other hyperparameter combinations and similar architectures could produce similar results.

Data set and training procedure

The training samples determine the features that the network learns. In a Bayesian context, the training data set selection becomes crucial because it defines the prior distribution. Nonetheless, the main limitation of applying supervised deep learning methods in geophysical imaging problems is the lack of databases with realistic and generalized ground-truth labels. Consequently, we prepare a synthetic data set generating 1200 pseudorandom reflectivity models, acknowledging that the restricted nature of the training instances will affect the quality of the reconstruction. In this work, the reflectivity model is expressed as velocity perturbations in squared slowness units. Therefore, the ground-truth labels are derived from velocity distributions that mimic fractured and folded sedimentary structures ranging from 1500 to 5500 m/s (Figure 3). Velocity, folding amplitude, and fault size are set to increase with depth gradually, and all instances have a variable number of layers. Each model has a fixed grid size of 400×200 points with a regular grid spacing of 10 m. For migration, we calculate the background field of every velocity model using a 2D Gaussian filter with standard deviations from 2 to 6 grid points chosen from a discrete uniform distribution (equivalent to the range of 10–50 m in intervals of 10 m). In the following numerical examples, we separate 900 reflectivity models for training, 100 models for validation, and 200 models for testing. The validation set is used to optimize hyperpara-

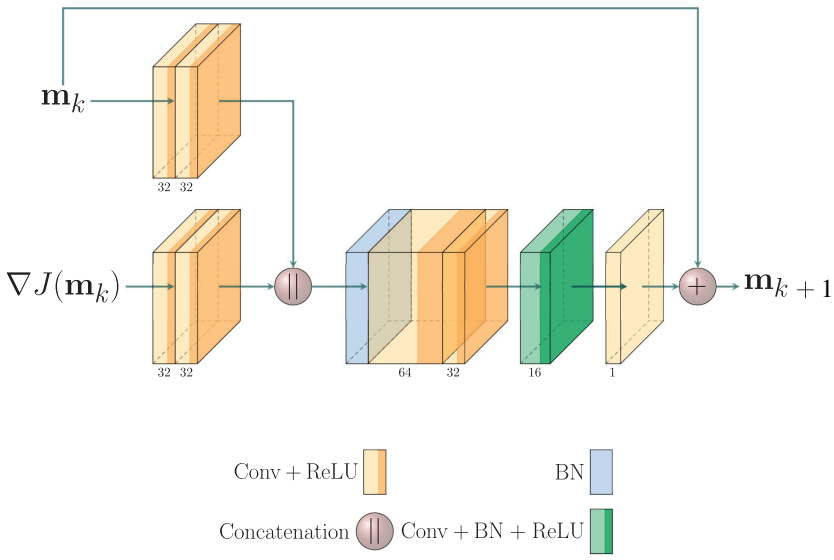


Figure 2. The proposed network architecture for each learned updating operator $\mathcal{P}_{\theta_k}$, based on an encoder-decoder scheme. The number of channels is shown below each convolutional layer. ReLU and BN refer to the rectified linear unit function and the batch normalization function, respectively.

meters while training and the test models were used to evaluate inference performance.

Because the prestack LSRTM gradient requires observed seismic data in the shot-gather domain, we generate the shot gathers by simulating an acquisition on every model before starting the training process and store them on disk. Even for such a small data set, including the data-misfit gradient is the most demanding part of our algorithm during training, and therefore, we consider acquisitions with a reduced number of sources. Specifically, the numerical simulations of wave propagation for each sample model involve a fixed-spread geometry with only 15 shots spanning from 100 to 3900 m in the horizontal direction and 30 m in depth and 400 collocated receivers evenly placed at 30 m in depth and 10 m spacing. Due to the coarse source distribution, aliasing artifacts are expected in the migrated images. The seismic data are computed using a time-domain finite-difference wavefield propagator with a surface absorbing boundary condition to simulate data with removed surface-related multiple. Each shot is recorded for 2.2 s with a time sampling of 1 ms, resulting in 2200 time steps of modeling time. A 20 Hz Ricker wavelet is used as the seismic source signature.

To further decrease the computational burden, we perform the gradient calculation step on a multi-graphics processing unit (GPU) cluster with four NVIDIA GeForce RTX 2080 Ti computing processors to generate gradient images simultaneously. All the models and their corresponding data (i.e., the observed seismic data, the background velocity models, and the seismic modeling parameters) are distributed among the devices using four threads, such that each GPU calculates the data-misfit gradient associated with a different model in parallel. Due to the relatively small size of the models and the number of time steps required to propagate the wavefield to all parts of the domain, we calculate each gradient using the traditional adjoint state method, storing the forward wavefield within the GPU. With this configuration, computing a demigration/migration sequence for 15 sources on each model takes approximately 25 s. Compared with the cost of multiple forward and backpropagation calls of the gradient step, the optimization of equation 13 is negligible.

Finally, we only consider the updating operator of the first deep-LSRTM iteration for hyperparameter selection, supported by the observation that the first iteration is the one that performs a more severe transformation.

After the hyperparameter tuning, we set the maximum number of iterations to $K = 5$, as further increasing the number of blocks does not lead to better reconstructions. We train each updating operator with the Adam optimizer (Kingma and Ba, 2014) to minimize equation 13, using a learning rate of 0.001, a batch size of two, and 111 epochs, where an *epoch* is defined as one pass through the training data. Training five iterations of deep LSRTM takes approximately 8 h in the multi-GPU cluster previously mentioned. Furthermore, we do not use data augmentation or other standard training techniques such as dropout or max pool layers. With such configuration, this architecture leads to a total of 341,441 trainable parameters per updating operator.

Algorithms 1 and 2 describe the complete deep-LSRTM training and inference processes, respectively. Equally to conventional LSRTM algorithms, we assume a zero reflectivity vector ($\mathbf{m}_0 = \mathbf{0}$) as the initial guess.

Figure 4 shows the loss curves of the training and validation data sets versus the number of epochs after five iterations of deep LSRTM.

It is clear that the error of both data sets reduces drastically in the first epochs of the first iteration and then gradually stagnates until the next iteration takes place. A similar pattern between training and validation losses with no signs of overfitting indicates good generalization properties of the model. Furthermore, the continued decrease in both losses with each iteration verifies the benefits of the iterative scheme.

Transfer learning for real applications

A crucial assumption in supervised deep learning algorithms is that the testing data must be in the same feature space and have the same distribution as the training data (Pan and Yang, 2010). However, when dealing with applications to real data, such condition is difficult to ensure if the training data set is only composed of synthetically generated samples. Hence, a direct application of the trained network usually will not produce satisfactory results because seismic data can have dramatically distinct features than those represented by the synthetic samples. To mitigate this out-of-distribution problem and to enable the successful implementation of deep LSRTM to real data, we resort to the transfer learning strategy based on model fine tuning (Park and Sacchi, 2020).

We devise a three-step workflow for a limited set of available measurement data. First, we pretrain the updating operators using Algorithm 1 and the synthetically generated training samples described in the previous section. The objective of the pretraining stage is to build a good prior for the targets of interest. Then, we set apart a reduced number of shots from the real observed data to obtain ground-truth samples by solving the inverse problem via preconditioned CGLS.

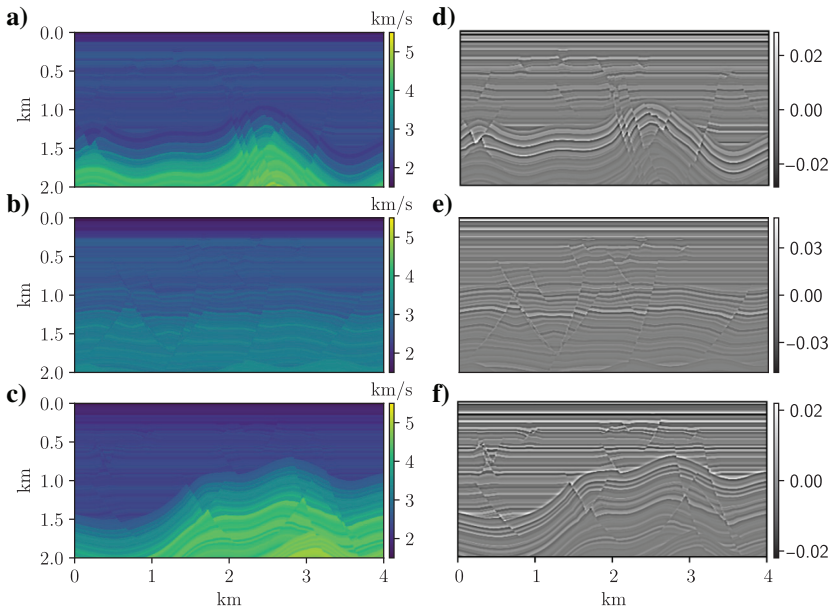


Figure 3. (a–c) Three examples of synthetically generated velocity distributions. (d–f) Ground-truth reflectivity samples derived from the velocity models in the first column.

We perform fine tuning by retraining each updating operator with a lower learning rate and fewer epochs, using the pretrained weights as the initial state. As a result, we have a network model that can be used to predict the target region's reflectivity. The updated parameters are finally tested on a different group of shots to obtain the final image.

Single-step image postprocessing via U-net reconstruction

We compare the iterative deep LSRTM with a two-step CNN-based postprocessing technique, in which first the seismic data are migrated to form an initial reconstruction, and then, we train a modified version of the U-net (Ronneberger et al., 2015) architecture to remove artifacts from the RTM images. It represents an efficient noniterative alternative to deep LSRTM because it does not require multiple computations of the data-misfit gradient, making the input-output demands considerably lower, so it is cheaper to train.

This second formulation substitutes equation 5 with

$$\mathbf{m} = \Lambda_{\Phi}(\mathbf{m}_{\text{mig}}), \tag{15}$$

where $\Lambda_{\Phi}: \mathbb{R}^N \rightarrow \mathbb{R}^N$ is an image-to-image transformation represented by the modified U-net network shown in Figure 5 and $\mathbf{m}$ is the filtered reflectivity image. We recognize that equation 15 is close in spirit to image-domain LSM methods that attempt to approximate the effect of the inverse Hessian in a single iteration (Guitton, 2004).

The U-net can be described as a multiscale convolutional auto-encoder. The encoder part of the network decomposes and resizes the image through a series of downsampling (max-pooling) layers to capture more extensive features, akin to wavelet decomposition. The spatial features are then up-sampled via transposed convolutions to reconstruct the final image. Moreover, a series of multilevel skip connections between the encoder and decoder parts helps preserve the different ranges of structures and avoids vanishing gradients (Ronneberger et al., 2015). Compared to the original U-net architecture, the modified U-net implemented in this study includes an additional skip connection between input and output layers for residual learning and a reduction from five to three scales to work with smaller images and fewer trainable parameters (517,409 in total). Similar to deep LSRTM, this architecture is a residual network because of the extrinsic skip connection. For consistency, our U-net also is trained with the Adam optimizer maintaining the same hyperparameter configuration as described in the previous section. Figure 6 shows the loss curves for this method, in which we also see a decrease of the loss functions with the number of epochs and training and validation losses with close behavior.

We train the parameters of the U-net architecture by minimizing the MSE loss:

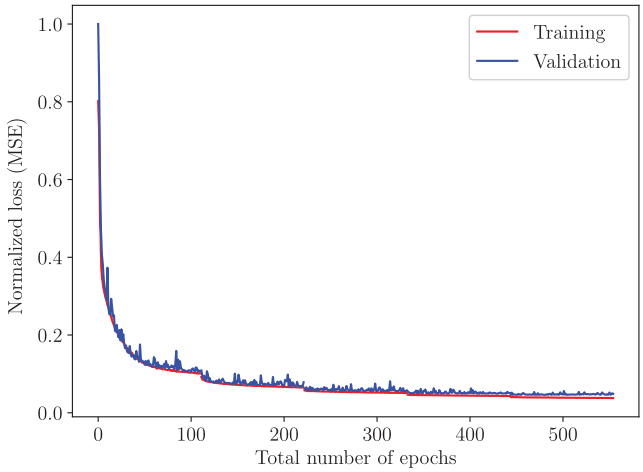


Figure 4. The normalized training and validation loss functions versus the number of epochs for the deep-LSRTM architecture.

Algorithm 1. Deep LSRTM (training).

Inputs: Data set $\Im$, initial reflectivity estimates $\mathbf{m}_0^i$, diagonal preconditioners $\mathbf{P}_m^i$, observed data $\mathbf{d}^i$, number of iterations K

Set $k = 0$

while $k < K$ **do**

$\mathbf{r}^i = (\mathbf{L}\mathbf{m}_k^i - \mathbf{d}^i)$ ▷ Compute residuals for all models

$\nabla\tilde{J}(\mathbf{m}_k^i) = \mathbf{D}\mathbf{P}_m^i\mathbf{L}^T\mathbf{P}_d\mathbf{r}^i$ ▷ Calculate preconditioned LSRTM gradients for all models

if $k > 0$ **then**

Initialize θ_k with θ_{k-1}

end if

$\min_{\theta_k}(1/\Im)\sum_{i=1}^{\Im}\|\mathcal{P}_{\theta_k}(\mathbf{m}_k^i, \nabla\tilde{J}(\mathbf{m}_k^i)) - \mathbf{m}_{\text{true}}^i\|_2^2$ ▷ Train $\mathcal{P}_{\theta_k}$ until stopping criteria

$\mathbf{m}_{k+1}^i = \mathcal{P}_{\theta_k}(\mathbf{m}_k^i, \nabla\tilde{J}(\mathbf{m}_k^i))$ ▷ Update reflectivity for all models

$k = k + 1$

end while

Output Trained parameters $\Theta = [\theta_0, \ldots, \theta_{K-1}]$

Algorithm 2. Deep LSRTM (inference).

Inputs: Trained parameters Θ, initial reflectivity estimate $\mathbf{m}_0$, diagonal preconditioner $\mathbf{P}_m$, observed data $\mathbf{d}$, number of iterations K

Set $k = 0$

while $k < K$ **do**

$\mathbf{r} = (\mathbf{L}\mathbf{m}_k - \mathbf{d})$ ▷ Compute residuals

$\nabla\tilde{J}(\mathbf{m}_k) = \mathbf{D}\mathbf{P}_m\mathbf{L}^T\mathbf{P}_d\mathbf{r}$ ▷ Calculate preconditioned LSRTM gradient

$\mathbf{m}_{k+1} = \mathcal{P}_{\theta_k}(\mathbf{m}_k, \nabla\tilde{J}(\mathbf{m}_k))$ ▷ Update reflectivity

$k = k + 1$

end while

Output $\mathbf{m}_K$

$$\min_{\Phi} \frac{1}{\Im} \sum_{i=1}^{\Im} \|\mathbf{m}^i_{\text{mig}} - \mathbf{m}^i_{\text{true}}\|_2^2, \tag{16}$$

where the training data pairs correspond to RTM images $\mathbf{m}_{\text{mig}}$ and ground-truth reflectivity models $\mathbf{m}_{\text{true}}$.

NUMERICAL EXAMPLES

This section reports the results of deep-LSRTM reconstruction in four different scenarios. The first three scenarios explore a direct implementation of deep LSRTM on synthetic models. In the first three scenarios, transfer learning was not used. The fourth scenario entails a field data set where we use a direct implementation of our method and the modification that includes the transfer learning component. In our numerical examples, we also consider LSRTM solutions computed by the preconditioned CGLS algorithm and the results obtained by the U-net reconstruction for baseline comparisons. To make fair comparisons between iterative methods, we run the same number of iterations (five) for deep-LSRTM and CGLS algorithms (unless otherwise stated) and use the same data and model space preconditioners. The terms preconditioned LSRTM and conventional LSRTM are used interchangeably henceforth, referring to the inversion via the preconditioned CGLS method. To avoid the inverse crime, all the observed data of the synthetic examples are generated with the solution of the discretized acoustic wave equation instead of Born acoustic modelling (Tarantola, 1984). We use Tensorflow (Abadi et al., 2015) for the numerical implementation of the CNN architectures. To quantify the imaging performance of each algorithm, we compute the peak signal-to-noise ratio (PS/N) (Huynh-Thu and Ghanbari, 2008) and the structural similarity index measure (SSIM) (Wang et al., 2004) (see Appendix B).

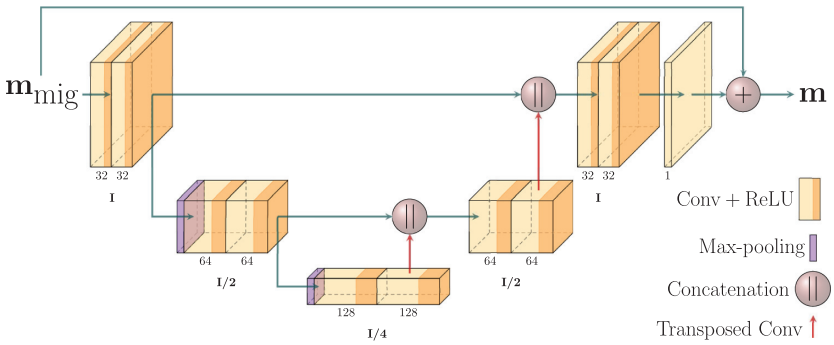


Figure 5. Architecture of the residual U-net adopted in this study. The number at the bottom of each convolutional layer indicates the number of channels. **I** refers to the original image size.

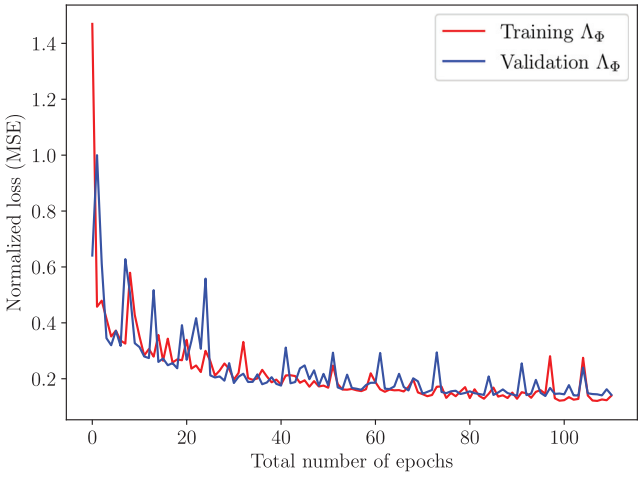


Figure 6. The normalized training and validation loss functions versus the number of epochs for the U-net architecture.

Example 1: Test data set

The first scenario examines the reconstruction of the test data set. The test samples were generated in the same way as the training and validation samples but were not included in the training phase. Hence, this experiment showcases the performance of deep LSRTM on models that have the same prior distribution as the training samples. Table 1 presents a quantitative comparison of the different reconstruction methods for noiseless data. Even though the U-net approach can produce good-quality images, the quantitative measures indicate that deep LSRTM delivers more competitive results for this data set. On average, both learned methods perform better than the preconditioned CGLS algorithm.

Figure 7 shows the results obtained for one sample of the test data set (Figure 7a). For this model, the migration velocity field was obtained by convolving the true velocity with a 50 m standard deviation 2D Gaussian filter (shown in Figure 7b). Figure 7c displays the RTM section after illumination compensation and filtering by a Laplacian operator. The adjoint image contains a strong acquisition footprint and unresolved areas at deeper parts of the model because of limited aperture. The preconditioned CGLS result (Figure 7d) shows slightly increased illumination and frequency content, but its main contribution after five iterations is the partial attenuation of the near-surface artifacts.

In contrast, deep LSRTM (Figure 7e) considerably improves the amplitude balance, the structural continuity, and the resolution of the image in comparison to the CGLS result for the same number of iterations. Moreover, it satisfactorily learns to remove the acquisition footprint at the top layers. The residual U-net (Figure 7f) also delivers a good-quality reconstruction but tends to smear out the reflectors at shadow regions (an example is the anticline structure marked with the blue rectangle in Figure 7), and the sharpening of steep faults is limited (indicated by the red rectangle in Figure 7).

We particularly notice that the deconvolution effect of the post-processing approach degrades in regions where the RTM image has poor illumination, which indicates a strong dependence between the U-net reconstruction and its inputs. However, the superior deconvolution effect of the iterative learned approach in this example seems to highlight the benefits of residual imaging. By comparing the gradients of the first and last iterations in Figure 8, we can observe that the magnitude of the gradient decreases with iterations due to a better match between the observed and calculated data, which is a reassuring, albeit expected, result.

Figure 9 shows a comparative analysis between the conventional LSRTM and the proposed method by means of the relative model error defined as

$$e(\mathbf{m}_k) = \frac{\|\mathbf{m}_k - \mathbf{m}_{\text{true}}\|_2}{\|\mathbf{m}_{\text{true}}\|_2}, \tag{17}$$

which is calculated for the inversion results of the sample model in Figure 7. Because both algorithms use the zero reflectivity vector as initial guess, they share the same resulting error for the initialization at $k = 0$. The conventional LSRTM converges to a relative model error of 0.36 after 20 iterations. In contrast, deep LSRTM achieves a considerably faster convergence to a smaller error value of 0.15 in only five iterations.

Example 2: Cropped Marmousi

The following scenario comprises a more complex setting extracted from the central part of the Marmousi model (Martin et al., 2006). The purpose of this test is to explore the generalization capacity of deep LSRTM to environments whose distributions differ from the ones used at training time. Similar to the first experiment, Figure 10a shows the true reflectivity model, and Figure 10b displays the background velocity, computed with a 50 m standard deviation 2D Gaussian filter. Figure 10c shows the RTM image, which possesses unfocused reflectors at the deeper part of the model and strong source-sampling aliasing artifacts that are uncorrelated with the geology, manifested as spurious tails. The least-squares method improves the content of high-frequency features, so the preconditioned CGLS image shows higher resolution with deblurring capability, especially at the shallow layers. Although the learned approaches offer increased resolution in deeper parts of the model, it can be noticed that, once again, the focusing properties of the U-net reconstruction (Figure 10) are limited by the RTM result. In this case, the U-net also introduces jittered high-frequency noise at the top layers. We attribute this behavior to the high structural complexity of the Marmousi model in the near-surface region compared with the less complex training samples. In this regard, the iterative learned approach achieves better generalization properties showing enhanced continuity of the reflectors and not introducing high-frequency artifacts at the top layers. Our qualitative assessment is supported by the PS/N and SSIM scores, presented in Table 2, in which the deep LSRTM attains superior performance.

Learned reconstructions as warm starts for CGLS

Exploiting the adaptability of the preconditioned CGLS algorithm, we illustrate the possibility of efficiently improving the results attained by the learned methods based on warm-start reconstruction. Warm start enables the preconditioned CGLS method to initialize the recursive updates with an initial reflectivity guess $\mathbf{m}_0 = \mathbf{0}$ provided by the user. Using the learned reconstruction results as feasible solutions to warm-start conventional LSRTM can help obtain an optimal solution in a reduced number of iterations. Figure 11a and 11b displays the results of five CGLS iterations (considering again a coarse source acquisition of only 15 shots) using the deep-LSRTM reconstruction and the U-net reconstruction as warm-start models, respectively. In general, the

Table 1. Quantitative measures for the reconstruction of the test data set.

Example 1 — test data set

Method	PS/N (dB)	SSIM
CGLS	24.81 ± 0.82	0.65 ± 0.089
U-net	31.63 ± 0.74	0.77 ± 0.041
Deep LSRTM	34.91 ± 0.38	0.87 ± 0.029

PS/N and SSIM scores are computed in comparison with the ground-truth label (average results ± standard deviation more than 200 different samples).

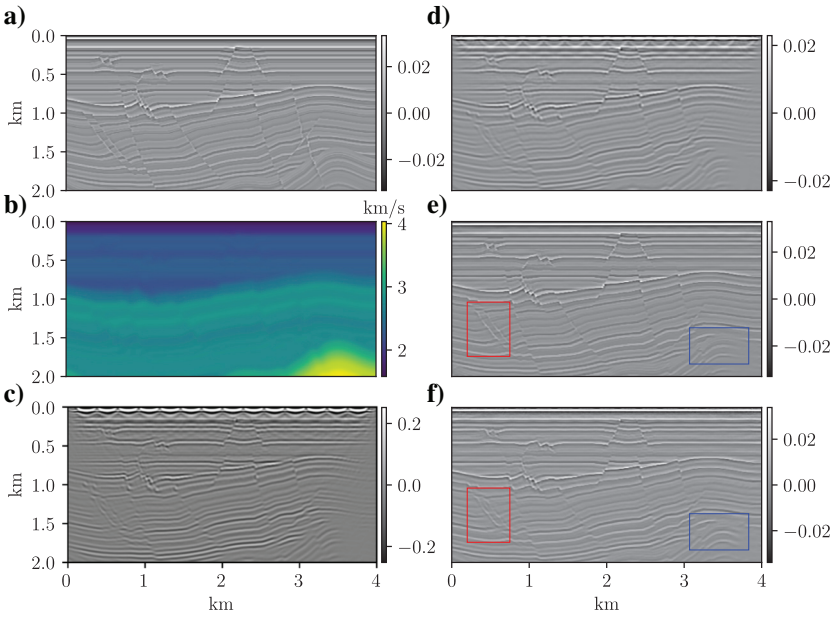


Figure 7. Imaging results for a sample of the test data set in experiment 1. (a) True reflectivity, (b) migration velocity model, (c) RTM image, (d) CGLS and (e) deep-LSRTM methods after five iterations, respectively, and (f) the U-net reconstruction. The red and blue rectangles exemplify reconstruction at poorly illuminated step faults and anticline structures, respectively.

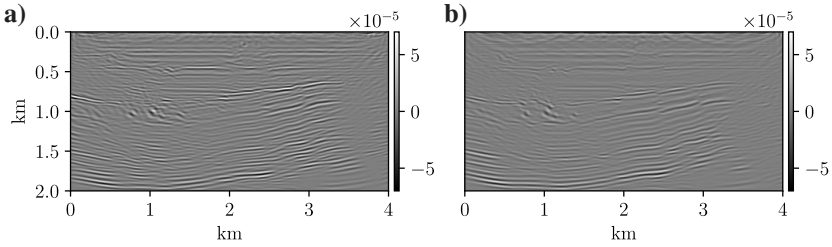


Figure 8. Deep-LSRTM gradients of the (a) first and (b) last iterations for the sample of the test data set in experiment 1, shown in Figure 7a.

results show enhancement in the recovery of model features not entirely reconstructed by the networks, especially at shallower regions, which yields higher PS/N and SSIM scores in both cases (see Table 2). Figure 11b also exhibits a reduction of the spurious high-frequency artifacts present in the initial U-net result. The convergence plot of the least-squares data misfit shown in Figure 12 verifies that the warm starting the CGLS algorithm with deep learning results produces faster convergence than conventional LSRTM, with the inversion started with the deep-LSRTM result decreasing slightly faster when compared to its U-net counterpart.

All of the tested methods cannot completely remove the remaining migration artifacts in the image, specifically in the central part of the model. We anticipate that increasing the number of sources can mitigate such artifacts at the expense of longer turnaround times.

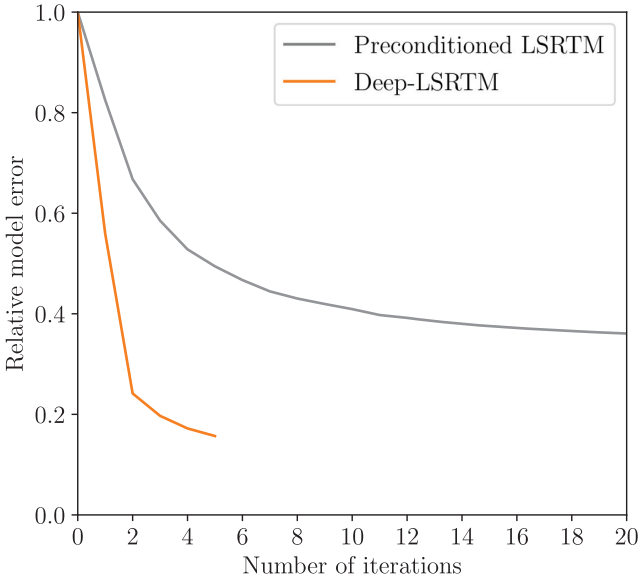


Figure 9. Convergence curve of the relative model error for comparative analysis between the preconditioned LSRTM and deep LSRTM.

Sensitivity to background model errors

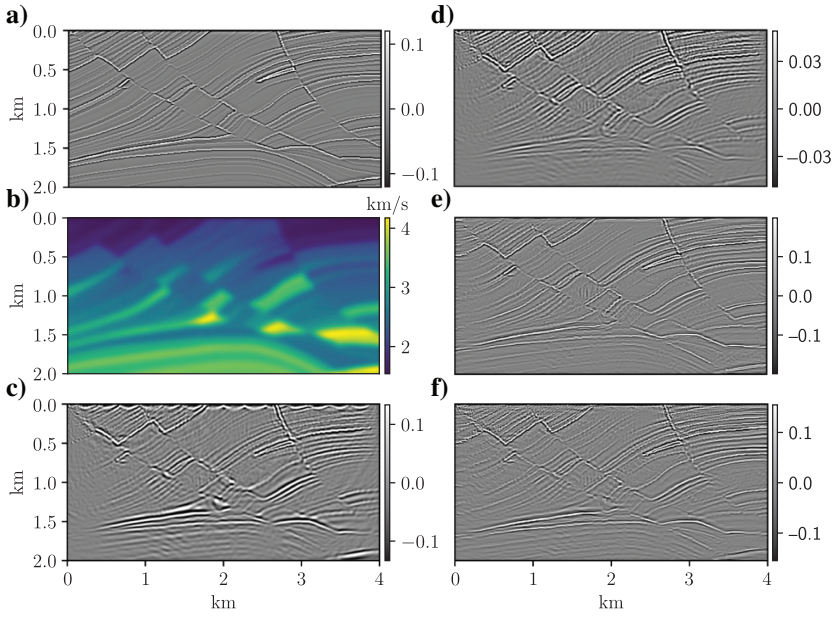
Conventional LSRTM is highly sensitive to incorrect background models by limiting its capacity to flatten and focus events at regions with significant velocity errors. In light of the lack of robustness of DNN-based inversion methods to slight variations in the reconstruction process (Antun et al., 2020), we test the proposed learning approaches against background models calculated with different degrees of spatial smoothing. The test is performed on the same setup as the previous section using the Marmousi model. To further explore the influence of the velocity in the inference process, we also consider a variant of the deep-LSRTM architecture that makes direct use of the background velocity field. Although the migration velocity is already embedded in the forward and adjoint operators, this modification is designed to help recover missing information in the model space by explicitly introducing the background velocity as a third complementary branch. In other words, before the evaluation step, we train the

Table 2. Quantitative measures for the reconstruction of a cropped version of the Marmousi model.

Example 2 — cropped Marmousi		
Method	PS/N (dB)	SSIM
CGLS	27.46	0.47
U-net	28.37	0.53
Deep LSRTM	29.87	0.65
Warm-started CGLS		
$CGLS_{U\text{-}net}$	29.96	0.58
$CGLS_{Deep\text{-}LSRTM}$	30.37	0.69

PS/N and SSIM scores are computed in comparison with the ground-truth label.

Figure 10. Imaging results for the cropped Marmousi model. (a) True reflectivity, (b) migration velocity model, (c) RTM image, (d) CGLS and (e) deep-LSRTM methods after five iterations, respectively, and (f) the U-net reconstruction.



modified deep-LSRTM architecture to take inputs of the form $(\mathbf{m}_k^i, \nabla\tilde{J}(\mathbf{m}_k^i), \mathbf{v}_0^i)$, where $\mathbf{v}_0^i$ is the fixed migration velocity model of the ith training sample. Explicitly using the migration velocity model as an extra input yields the modified optimization problem:

$$\min_{\theta_k} \frac{1}{\mathfrak{F}} \sum_{i=1}^{\mathfrak{F}} \|\mathcal{P}_{\theta_k}(\mathbf{m}_k^i, \nabla\tilde{J}(\mathbf{m}_k^i), \mathbf{v}_0^i) - \mathbf{m}_{\text{true}}^i\|_2^2. \tag{18}$$

Figure 13 shows the PS/N metric for all the methods considering a background velocity model smoothed with a 2D Gaussian filter of standard deviation varying from 50 to 110 m in intervals of 10 m.

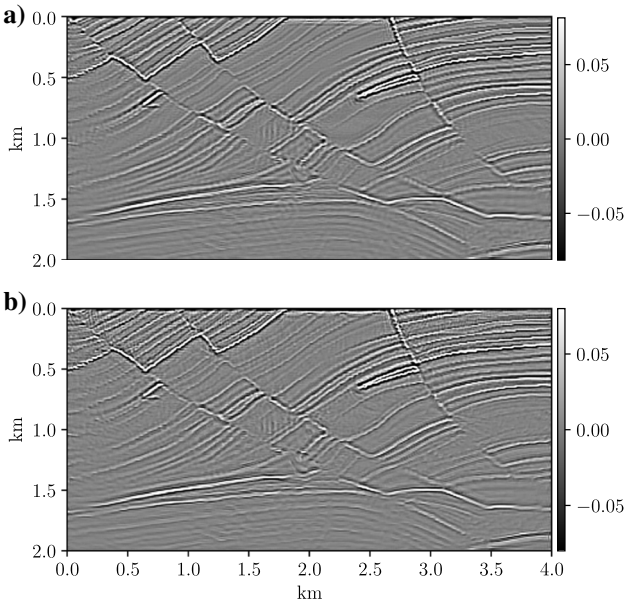


Figure 11. Results after five iterations of the warm-started CGLS algorithm using (a) the fifth iteration of the deep-LSRTM method and (b) the U-net reconstruction, as initial guesses, respectively.

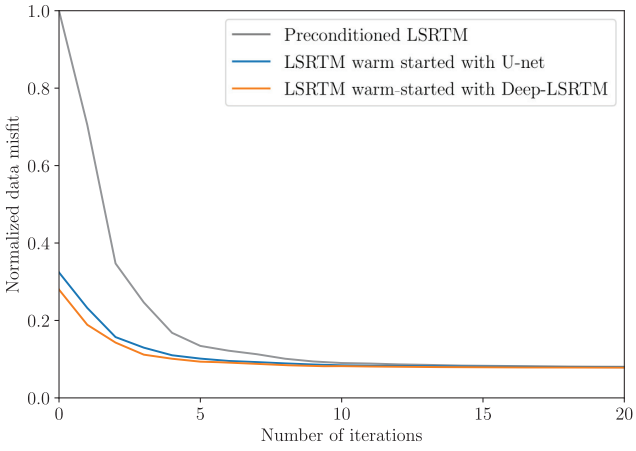


Figure 12. Convergence curves for the preconditioned LSRTM, LSRTM warm started with the prediction of the U-net post processing approach, and LSRTM warm started with the prediction of the deep-LSRTM result. The curves are normalized with respect to the initial value of the preconditioned LSRTM data misfit.

We observe a general degradation of the reconstruction quality as smoothing increases for all the techniques under consideration. It can be noticed that providing a correct velocity model as direct input in the modified deep-LSRTM architecture helps recover reflectors that are undetectable by the adjoint operator. This result is shown in Figure 14a. However, for background velocities with high levels of smoothing not seen during training, the postprocessing U-net and the original deep-LSRTM architecture are more robust than the three-branch deep-LSRTM version, which deteriorates much quicker. Figure 14b shows the artifact-prone prediction of the modified deep LSRTM with a smoother velocity model.

Finally, we test the reconstruction of our method using a migration velocity model with 5% faster velocity everywhere.

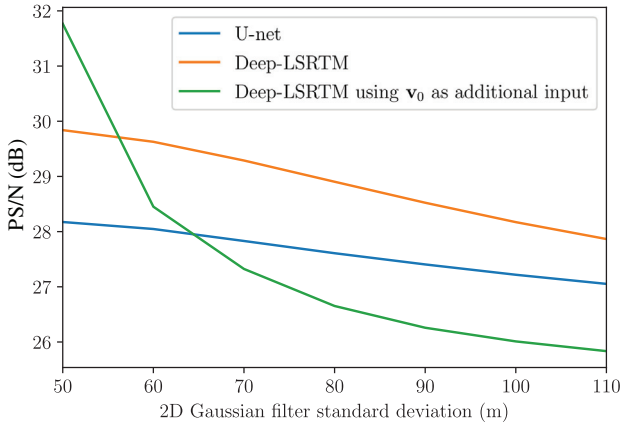


Figure 13. Comparison of reconstruction quality for the learned approaches on the central part of the Marmousi model using migration velocity models with different levels of smoothing.

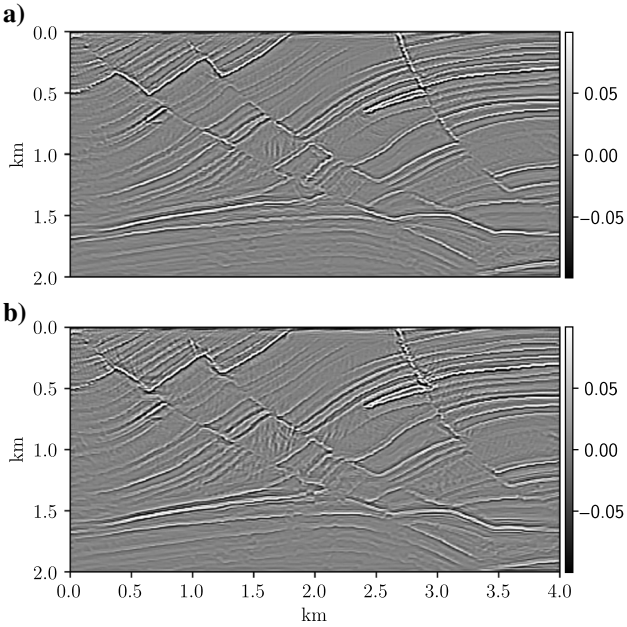


Figure 14. Modified deep-LSRTM architecture results: (a) reconstruction with a 50 m deviation 2D Gaussian filter and (b) reconstruction with a 110 m deviation 2D Gaussian filter.

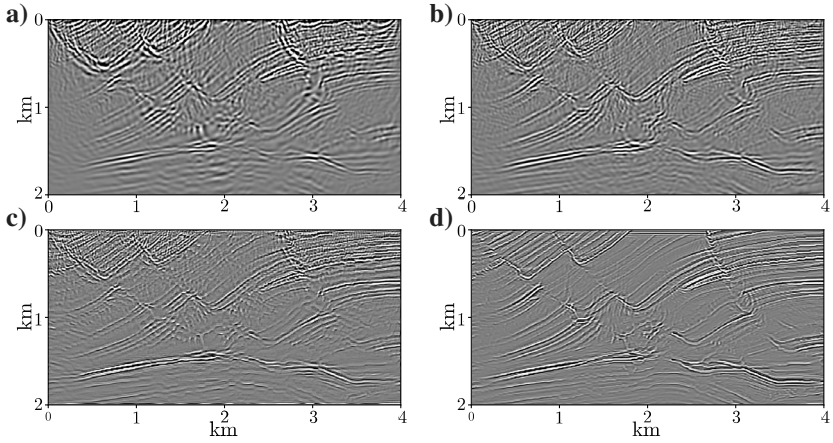


Figure 15. The migration images for an inaccurate migration velocity model with 5% faster velocity everywhere: (a) the RTM, (b) 20 iterations of preconditioned LSRTM, (c) the U-net reconstruction, and (d) the deep-LSRTM method.

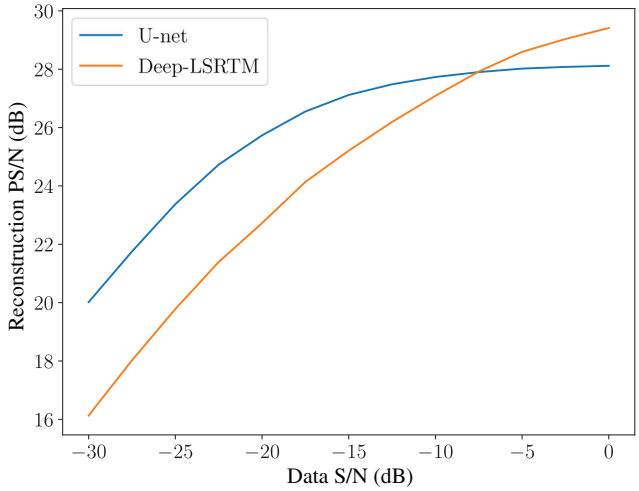


Figure 16. Robustness study of the learned methods against noisy data reconstruction. The horizontal axis indicates the level of S/N.

Figure 15a–15d shows the results of the RTM, 20 iterations of conventional LSRTM, the U-net method, and five iterations of deep LSRTM, respectively. Although all the methods fail to correctly position the reflection events due to significant inaccuracies on the velocity model, it is apparent that deep LSRTM provides a result with better spatial resolution than the traditional LSRTM, which produces a blurred and incoherent image, and a superior capability to filtering spurious artifacts compared to the learned post-processing technique.

Sensitivity to random noise

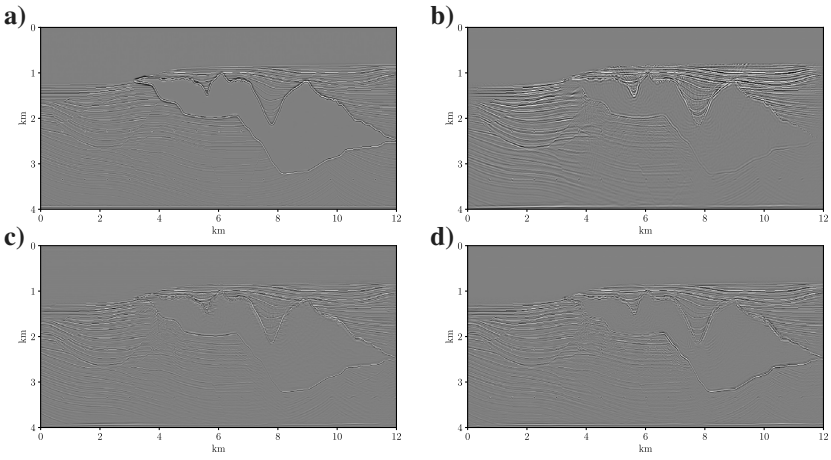
We examine the robustness of the learned approaches concerning additional noise in the seismic data. Accordingly, we synthetically contaminate the observed data with different levels of normally distributed random noise. Figure 16 displays the evolution of PS/N scores of the reconstructions for different levels of signal-to-noise ratio (S/N). Interestingly, the residual U-net proves to be more robust for the addition of noise variations, whereas deep LSRTM degrades faster under decreasing S/Ns. We attribute this behavior to the higher number of parameters and multiscale architecture of the U-net, which has been proven efficient in removing and detecting artifacts in images.

Table 3. The PS/N (dB) scores of deep LSRTM for different configurations of dominant frequency f_{dom} and grid spacing h tested in the Marmousi example.

	$f_{\text{dom}} = 10$ Hz	$f_{\text{dom}} = 15$ Hz	$f_{\text{dom}} = 20$ Hz
$h = 10$ m	26.75	28.47	29.87
$h = 15$ m	28.75	28.91	26.61
$h = 20$ m	29.52	26.62	26.59

The filled cells designate the values of f_{dom} and h used for the training stage.

Figure 17. Imaging results for the Sigsbee2a model. (a) True reflectivity, (b) LSRTM image, (c) U-net image, and (d) deep-LSRTM image.



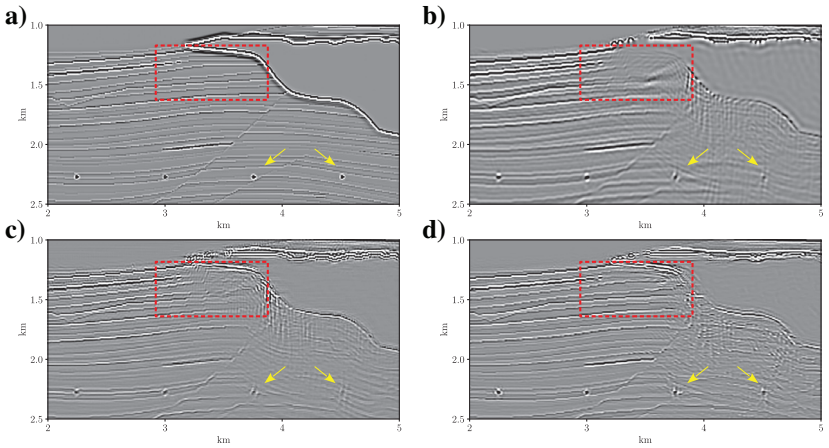


Figure 18. A magnified comparison of the reconstruction methods for the Sigsbee2a example. (a) True reflectivity, (b) conventional LSRTM image after 20 iterations, (c) U-net image, and (d) deep-LSRTM image. The dashed red box indicates a shadow zone region near the top left flank of the salt body. The yellow arrows indicate two point diffractors under the salt body.

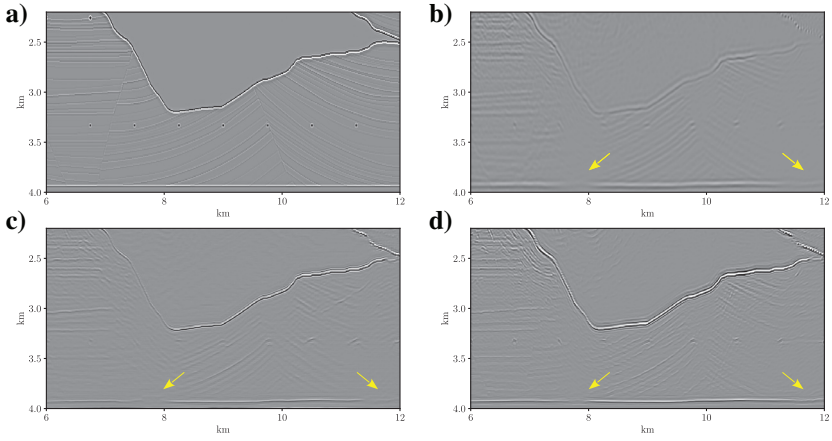


Figure 19. Magnified view of the reconstructions for the bottom right region of the Sigsbee2a model. (a) True reflectivity, (b) conventional LSRTM image after 20 iterations, (c) U-net image, and (d) deep-LSRTM image. The yellow arrows indicate regions where it is challenging to image the bottom reflector due to illumination limitations.

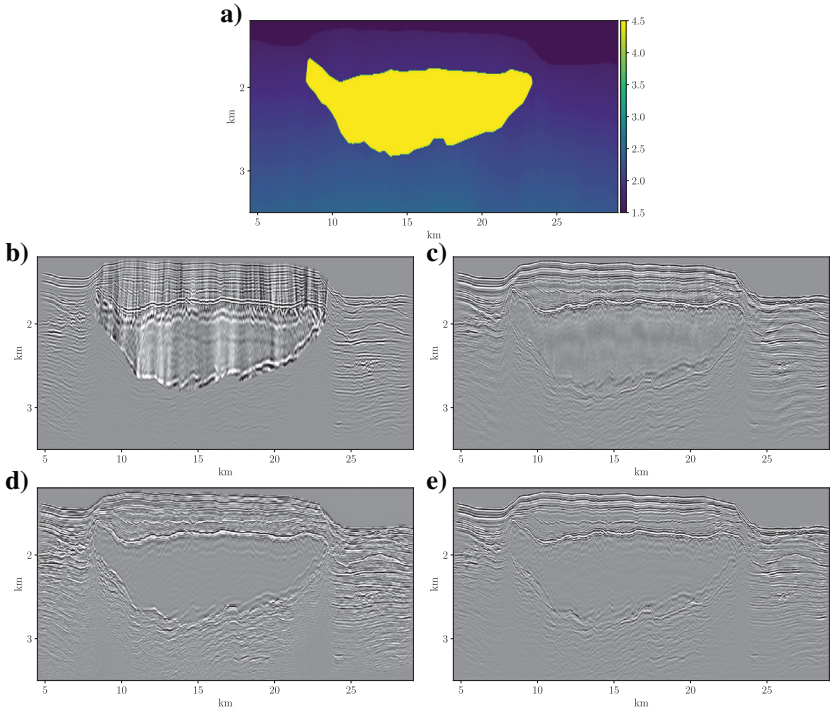


Figure 20. The GOM field data example. (a) Migration velocity field, (b) RTM result, (c) LSRTM result, (d) deep LSRTM without transfer learning, and (e) deep LSRTM after transfer learning.

Sensitivity to dominant frequency and grid spacing

To further investigate the generalization ability of the proposed workflow in terms of other relevant parameters, we experimentally quantify the prediction performance regarding variations in the dominant frequency of the modeling Ricker wavelet and the grid spacing. We perform these tests on the Marmousi example. The results are presented in Table 3. We see a general degradation of the prediction results for values that are different from the baseline (i.e., $f_{\text{dom}} = 20$ Hz, $h = 10$ m) because all the experiments from the training data set used a fixed value for the dominant frequency and fixed grid spacing. We can notice less degradation in the prediction performance for values closer to the baseline configuration. The phenomenon of training generalization still requires further research because it may be influenced by a variety of other factors not considered in this study (e.g., the subsurface complexity of the training models and the number of training samples).

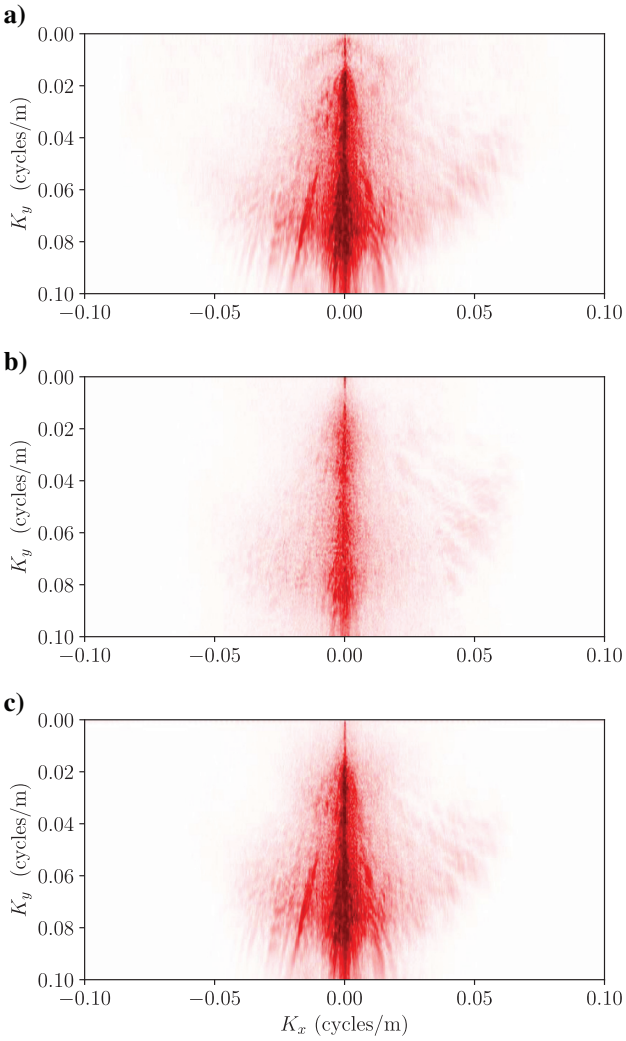


Figure 21. The 2D wavenumber spectrum of (a) LSRTM, (b) deep LSRTM before transfer learning, and (c) deep LSRTM after transfer learning. All three plots use the same color scale.

Example 3: Sigsbee2a model

For our third example, we use the resampled Sigsbee2a model (Paffenholz et al., 2002), a challenging salt model with 12 km in horizontal distance and 4 km in depth with a grid interval of 10 m (Figure 17a). We simulate an off-end acquisition geometry where the positions of the receivers move with the positions of the sources from left to right at zero depth. The data set consists of 100 observed shot records with 5 s recording time and 183 evenly spaced hydrophones. The maximum and minimum offsets are 4874 and 100 m, respectively. The wavelet frequency and time sampling are kept the same as the training data.

The high-velocity contrast and irregular shape of the intruding salt body challenge the generalization of our trained networks, which have not seen complex examples with salt during training. Figure 17 shows the inversion results for the Sigsbee2a example. For comparison, we include the results after 20 iterations of the preconditioned LSRTM (Figure 17b) and the U-net postprocessing approach (Figure 17c). The result after five iterations of our iterative network is shown in Figure 17d. The predicted deep-LSRTM image produced good results, particularly on the faulted sedimentary region at the leftmost part of the model. We believe this is because the samples used for training are closer to representing the full space solution on this specific region of the model by correctly describing all features of interest. We also observe improved reconstruction performance at the shallow layers and at the top of the salt region, where we have the best illumination in the model compared with the other two methods.

To show how deep LSRTM copes with severe illumination issues in subsalt areas, Figure 18 presents a magnified view of a shadow zone region (indicated by the dashed red box) near the top left flank of the salt body. Structures in this shadow zone produce events with very little or no illumination, and they cannot be modeled using Born modeling and restored using LSRTM (Figure 18b). In contrast, the regularization performance of our method is remarkable, showing better continuity than the LSRTM inversion after 20 iterations, from which we can identify the reflectors under the salt overburden. In addition, there is better focusing on the two point diffractors located beneath the salt (indicated by the yellow arrows).

To further address illumination loss challenges, we present a magnified view of a deeper subsalt region in Figure 19. Although none of the methods achieves a perfect amplitude balancing of the bottom reflector near the subsalt areas (indicated by the yellow arrows), we observe higher spatial resolution and sharper events in the predictions of the learned approaches compared with those of the LSRTM image.

The Sigsbee2a example exhibits three essential points about the proposed method. First, deep LSRTM works in the image domain rather than the data domain, and therefore, it is robust against changes in the acquisition geometry. Second, because the architecture of the projection operators is fully convolutional, our technique also can handle inputs of different sizes than the ones used in training. Moreover, input image size has no bearing on the number of trainable parameters; they are determined by the number of hidden layers and feature maps. Finally, although our framework is trained on a simulated data set containing only simple structures with shallow depths, it remains applicable to more complex data featuring salt bodies and regions with severe illumination loss. We attribute this behavior to the iterative nature of the algorithm, which leads to

more accurate inversion results because it incorporates a feedback mechanism that promotes consistency with the physics of the problem. We stress that our method may not perform well for all sorts of data. However, this experiment demonstrates that the reconstructions may still be acceptable for some testing data that differs from the training set.

Example 4: GOM data set

The last example comprises the implementation of our method on a 2D GOM marine line, known as the Mississippi Canyon data set (Dragoset, 1999). Such a data set is useful to test the generalization capability of our learned iterative approach on true earth models with inevitably inaccurate migration velocities and hugely different structures than those learned during training. The velocity model used for migration is shown in Figure 20a (Guitton et al., 2012), containing a shallow salt body in a deepwater environment. The model is discretized in a regular grid with 2908×350 points, corresponding to a distance of 29.08 km and a depth of 3.5 km with 10 m of grid spacing. The acquisition setup is off-end, with 809 sources and 180 receivers per shot moving from left to right at the surface. Receiver and shot spacing is 26.67 m. The farthest offset is 4875 m, and the nearest offset is 100 m. The seismic data underwent surface-related multiples elimination prior to imaging. To decrease the computational cost, we selected only 179 evenly separated shots of the original 809 shots for migration, 133 m apart. The shot decimation also introduces truncation artifacts, which can be used to assess how successful the proposed method is at reducing acquisition artifacts (Guitton, 2012; Rocha et al., 2018). Figure 20b shows the RTM result. The strong velocity contrast around the top of the salt produces strong amplitude low-frequency artifacts in this model region.

Furthermore, subsalt events suffer from uneven illumination, visible migration artifacts, and limited resolution. Figure 20c shows the LSRTM result after 20 iterations of preconditioned CGLS, which provides a noticeable improvement in the image, displaying more bandwidth content, reduced artifacts, and better-resolved reflectors, especially near and beneath the salt body. Figure 20d shows a direct implementation of deep LSRTM, which produces an artificial enhancement of the amplitude in depth. However, the output synthesized by the learned iterative approach still displays patterns seen in the training set, clearly indicating the risk of performing learned reconstructions on data that have undergone a significant distributional shift.

We use the previously discussed transfer learning scheme to improve this result further. The reference model used as the ground-truth image is extracted using a different group of 60 shots from the same

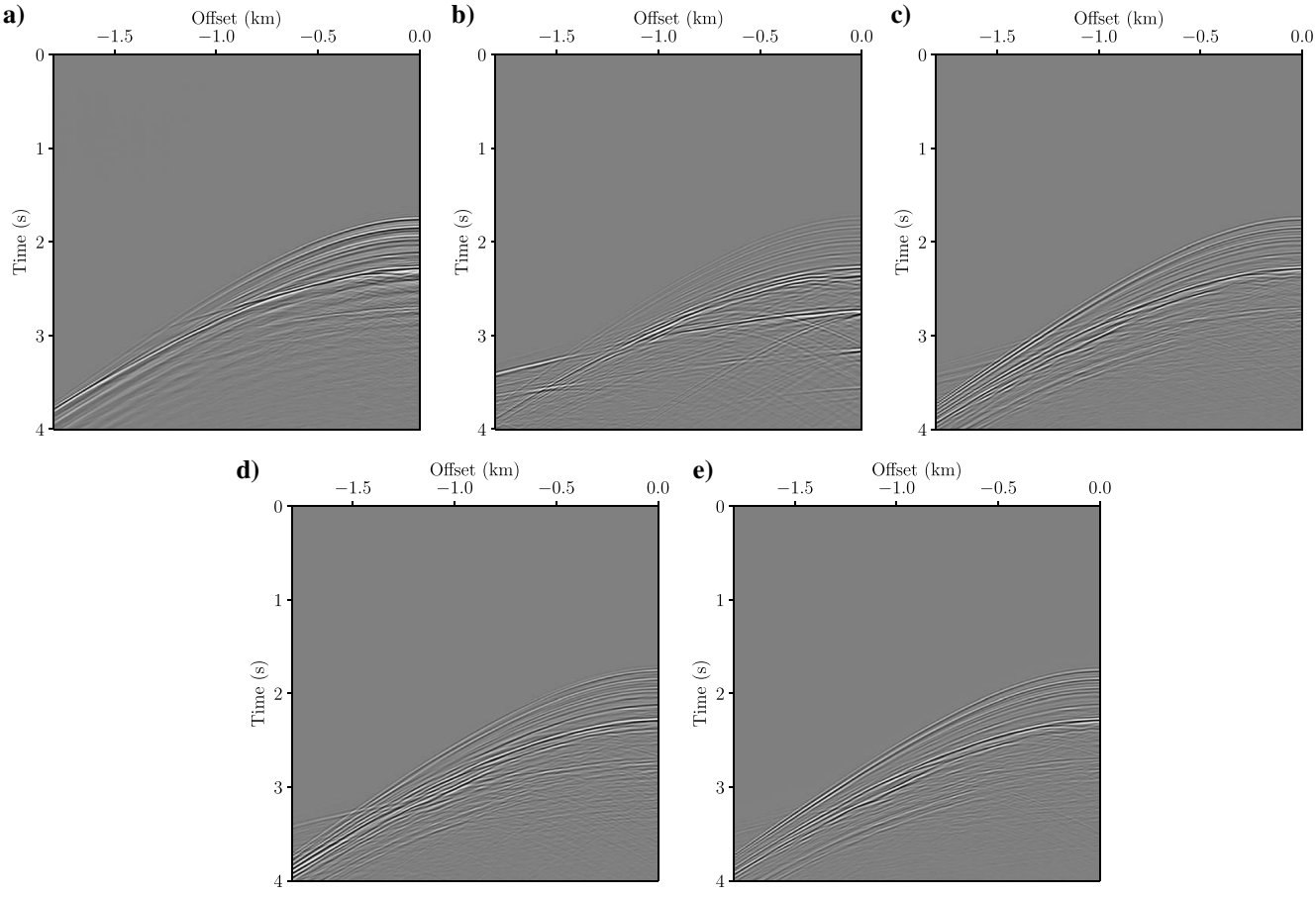


Figure 22. The shot gather of a source located at $x = 15.6$ km for (a) the observed data set, (b) the demigrated data using the RTM image, (c) the demigrated data using the LSRTM image, (d) the demigrated data using the deep LSRTM without transfer learning, and (e) the demigrated data using the deep LSRTM after transfer learning.

data set with source spacing of 400 m and performing 20 iterations of the preconditioned CGLS. Then, we retrain the weights of each updating operator with only 20 additional epochs and a reduced learning rate of 1e-5. The result of deep LSRTM after transfer learning is shown in Figure 20e. Transfer learning has a significant impact on the predictions of the learned iterative approach. Interestingly, the outcome of the updated networks has better preservation and continuity of reflectors at the top of the salt and produces a cleaner image with fewer low-frequency contents than the conventional LSRTM result. This might be because the original weights were trained with clean labels, so deep LSRTM already possesses a filtering effect by learning how to remove migration artifacts efficiently (Yosinski et al., 2014). Deep LSRTM also is more efficient in removing unwanted artifacts product of significant velocity errors as illustrated in example 2.

Because there is no label for the real data case to validate the results, we show the wavenumber spectra for the results of LSRTM, deep LSRTM before transfer learning, and deep LSRTM after transfer learning in Figure 21a–21c, respectively, where it can be notice that the retrained approach produces a wavenumber that is closer to the LSRTM result.

In addition, Figure 22 shows the demigrated shot gathers for a source located in the central part of the model at 15.6 km. Although the adjoint method cannot match the data due to amplitude and phase mismatches, all the inversion results yield a better correlation with the observed recorded data. Figure 23 shows this behavior in more detail by comparing three contiguous near-offset traces extracted from the shot mentioned previously. Comparing the demigrated shots obtained using the inverted results (Figure 22c–22e), we notice that while the change is not dramatic for near-offset traces, the waveform fit has been improved in the mid-to-long offsets after the use of transfer learning.

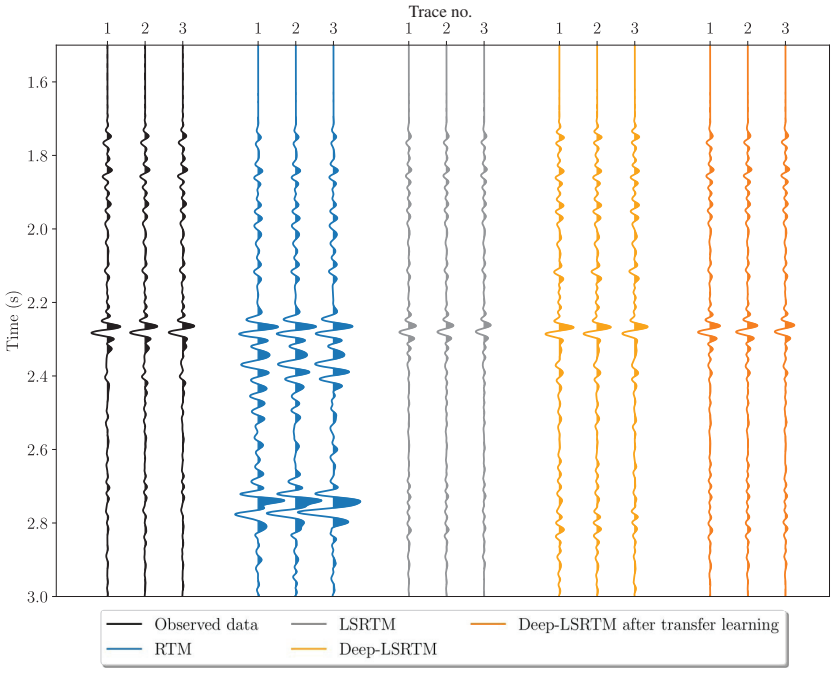


Figure 23. Amplitude comparison of three contiguous near-offset traces extracted from the shot gathers shown in Figure 22.

CONCLUSION

Based on the recent advances in deep learning, we have built an LSRTM framework that leverages the universal approximation capabilities of CNNs to predict reflectivity updates by mimicking a PGD algorithm. Tests on synthetic data show that, despite using a reasonably small training set, the iterative deep-LSRTM approach yields superior results than conventional LSRTM baselines for the same number of iterations. Once trained, the computational cost per iteration of the learned projection method is similar to one iteration of the CGLS algorithm, but it requires fewer iterations to deliver high-resolution results. When evaluated on the testing data, deep LSRTM also outperforms a two-step residual U-net postmigration application according to the PS/N and SSIM scores, highlighting the value of including the forward and adjoint wave operators in the inference process. Satisfactory results over the central part of the Marmousi model and the Sigsbee2a model (out-of-distribution samples) confirm that the deep-LSRTM network is not severely influenced by model overfitting. However, the modified residual U-net method stood as an efficient alternative, filtering RTM images to obtain enhanced reflectivity approximations. However, this comes at the trade-off of being more susceptible to overfitting to the training data. For the Marmousi test, both learned methods proved to be stable to different degrees of smoothing of the migration velocity model, whereas the multiscale architecture of the U-net provides more robust performance for seismic data with higher levels of Gaussian noise. Additional iterations of the warm-started CGLS algorithm using either of the results given by the learned approaches as initial guesses led to higher PS/N and SSIM scores. A direct implementation of deep LSRTM on the Mississippi Canyon data set shows that the technique is sensitive to the different characteristics of the seismic data. Transfer learning helps to adapt the trained weights to a new distribution at a fraction of conventional LSRTM using only a reduced portion of the observed data. However, future research is needed to find the optimal number of shots for transfer learning and analyze how it affects the final prediction. It is significant for us also to study more efficient ways of adapting our method to real data sets. This will be the core of our future investigation.

The proposed approach is not limited to the 2D acoustic RTM engine and can potentially be extended to other migration techniques and 3D models. Future research will explore the performance of deep LSRTM in 3D large-scale acquisitions. Moreover, we expect that incorporating more information into the networks using, for example, extended-domain imaging conditions can help improve reconstruction performance in scenarios with incorrect velocity models. Identifying uncertainty in DNN solutions to inverse problems is still in its early stage. Still, studies that consider uncertainty quantification of the proposed network architecture also are further required.

ACKNOWLEDGMENTS

We are grateful to A. Guitton for providing the velocity model for testing the proposed algorithm

on the GOM data set. The authors also thank the editors J. Etgen and F. Liu, along with four anonymous reviewers for their constructive comments and suggestions that helped to improve the manuscript. Finally, we thank the sponsors of the SAIG group for their financial support.

APPENDIX A

PARAMETERIZATION OF THE PROJECTION OPERATORS

Conceptually, each updating operator block $\mathcal{P}_{\theta_k}$, corresponding to the k iteration of the deep-LSRTM framework, can be characterized as an N stack of 2D convolutional layers with trainable weights and nonlinear activation functions of the form

$$\mathcal{P}_{\theta_k} = ((\phi_N \circ W_{w_N,b_N}) \circ \ldots \circ (\phi_1 \circ W_{w_1,b_1}))_k, \quad n = 1, \ldots, N, \tag{A-1}$$

where ϕ is the ReLU function:

$$\mathrm{ReLU}(\mathbf{m}) = \max(0, \mathbf{m}), \tag{A-2}$$

which is applied to each convolutional layer for easier and faster training, and

$$W^q_{w_n,b_n} = \left(b^q_n + \sum_{p\in P} w^{q,p}_n * g_p\right), \quad q \in Q, \tag{A-3}$$

where P and Q denote the number of channels for the input data g and the output feature maps W_{w_n,b_n} of the nth convolutional layer, respectively; b^q_n is a bias term; and $w^{q,p}_n$ is the 2D convolutional kernel (or filter).

APPENDIX B

FIGURES OF MERIT

We provide two types of measurements to assess the performances of different techniques. The PS/N function (Huynh-Thu and Ghanbari, 2008) computes the PS/N between the reconstructed image $\mathbf{m}$ and the ground-truth image $\mathbf{m}_{\mathrm{true}}$ and is calculated by

$$\mathrm{PS/N}(\mathbf{m}, \mathbf{m}_{\mathrm{true}}) = 20\log_{10}\left(\frac{\max(\mathbf{m}_{\mathrm{true}})}{\sqrt{\mathrm{MSE}}}\right), \tag{B-1}$$

where MSE is the mean-squared error and $\max(\cdot)$ is the maximum possible intensity value. The SSIM (Wang et al., 2004) is used to indicate the perceived similarity in the reconstructions. It is locally computed between two windows x and y of equal size $N \times N$, which move pixel-by-pixel over the entire reconstructed and ground-truth images. The SSIM is given by the following expression:

$$\mathrm{SSIM(x, y)} = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{B-2}$$

where μ_x and σ_x indicate the average and variance of x, respectively; μ_y and σ_y denote the average and variance of y, respectively; σ_{xy} represents the covariance of x and y; and c_1 and c_2 are two constants to avoid instability.

REFERENCES

Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, 2015, TensorFlow: Large-scale machine learning on heterogeneous systems: Software available from tensorflow.org, accessed 14 September 2020.

Adler, A., M. Araya-Polo, and T. Poggio, 2021, Deep learning for seismic inverse problems: Toward the acceleration of geophysical analysis workflows: IEEE Signal Processing Magazine, **38**, 89–119, doi: 10.1109/MSP.2020.3037429.

Adler, J., and O. Öktem, 2017, Solving ill-posed inverse problems using iterative deep neural networks: Inverse Problems, **33**, 124007, doi: 10.1088/1361-6420/aa9581.

Alfarraj, M., and G. AlRegib, 2019, Semi-supervised learning for acoustic impedance inversion: 89th Annual International Meeting, SEG, Expanded Abstracts, 2298–2302, doi: 10.1190/segam2019-3215902.1.

Antun, V., F. Renna, C. Poon, B. Adcock, and A. C. Hansen, 2020, On instabilities of deep learning in image reconstruction and the potential costs of AI: Proceedings of the National Academy of Sciences, **117**, 30088–30095, doi: 10.1073/pnas.1907377117.

Araya-Polo, M., S. Farris, and M. Florez, 2019, Deep learning-driven velocity model building workflow: The Leading Edge, **38**, 872a1–872a9, doi: 10.1190/tle38110872a1.1.

Araya-Polo, M., J. Jennings, A. Adler, and T. Dahlke, 2018, Deep-learning tomography: The Leading Edge, **37**, 58–66, doi: 10.1190/tle37010058.1.

Baysal, E., D. D. Kosloff, and J. W. Sherwood, 1983, Reverse time migration: Geophysics, **48**, 1514–1524, doi: 10.1190/1.1441434.

Becker, S., L. Horesh, A. Aravkin, and S. Zhuk, 2015, General optimization framework for robust and regularized 3D full waveform inversion: 77th Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: 10.3997/2214-4609.201412589.

Biswas, R., M. K. Sen, V. Das, and T. Mukerji, 2019, Prestack and poststack inversion using a physics-guided convolutional neural network: Interpretation, **7**, no. 3, SE161–SE174, doi: 10.1190/INT-2018-0236.1.

Boink, Y. E., C. Brune, and S. Manohar, 2019, Robustness of a partially learned photoacoustic reconstruction algorithm: Photons Plus Ultrasound: Imaging and Sensing, International Society for Optics and Photonics, SPIE, 88–94.

Buur, J., and T. Kühnel, 2008, Salt interpretation enabled by reverse-time migration: Geophysics, **73**, no. 5, VE211–VE216, doi: 10.1190/1.2968690.

Calvetti, D., S. Morigi, L. Reichel, and F. Sgallari, 2000, Tikhonov regularization and the L-curve for large discrete ill-posed problems: Journal of Computational and Applied Mathematics, **123**, 423–446, doi: 10.1016/S0377-0427(00)00414-3.

Chang, J. R., C.-L. Li, B. Póczos, B. Vijaya Kumar, and A. C. Sankaranarayanan, 2017, One network to solve them all — Solving linear inverse problems using deep projection models: IEEE International Conference on Computer Vision, 5889–5898.

Chen, H., J. Gao, X. Jiang, Z. Gao, and W. Zhang, 2021, Optimization-inspired deep learning high-resolution inversion for seismic data: Geophysics, **86**, no. 3, R265–R276, doi: 10.1190/geo2020-0034.1.

Cheng, C., Y. He, B. Wang, and Y. Huang, 2020, Structure enhanced least-squares migration by deep learning based structural preconditioning: 90th Annual International Meeting, SEG, Expanded Abstracts, 2908–2912, doi: 10.1190/segam2020-3426676.1.

Cheng, J., N. Kazemi, and M. Sacchi, 2016, Least-squares migration via a gradient projection method — Application to seismic data deblending: 78th Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: 10.3997/2214-4609.201601413.

Clapp, M. L., 2005, Imaging under salt: Illumination compensation by regularized inversion: Ph.D. thesis, Stanford University.

Dai, W., P. Fowler, and G. T. Schuster, 2012, Multi-source least-squares reverse time migration: Geophysical Prospecting, **60**, 681–695, doi: 10.1111/j.1365-2478.2012.01092.x.

Díaz, E., and P. Sava, 2016, Understanding the reverse time migration backscattering: Noise or signal?: Geophysical Prospecting, **64**, 581–594, doi: 10.1111/1365-2478.12232.

Dong, S., J. Cai, M. Guo, S. Suh, Z. Zhang, B. Wang, and E. Z. Li, 2012, Least-squares reverse time migration: Towards true amplitude imaging

and improving the resolution: 82nd Annual International Meeting, SEG, Expanded Abstracts, doi: 10.1190/segam2012-1488.1.
Dragoset, B., 1999, A practical approach to surface multiple attenuation: The Leading Edge, **18**, 104–108, doi: 10.1190/1.1438132.
Dutta, G., 2017, Sparse least-squares reverse time migration using seislets: Journal of Applied Geophysics, **136**, 142–155, doi: 10.1016/j.jappgeo.2016.10.027.
Etgen, J., S. H. Gray, and Y. Zhang, 2009, An overview of depth imaging in exploration geophysics: Geophysics, **74**, no. 6, WCA5–WCA17, doi: 10.1190/1.3223188.
Fletcher, R. P., D. Nichols, R. Bloor, and R. T. Coates, 2016, Least-squares migration — Data domain versus image domain using point spread functions: The Leading Edge, **35**, 157–162, doi: 10.1190/tle35020157.1.
Gao, W., G. Matharu, and M. D. Sacchi, 2020, Fast least-squares reverse time migration via a superposition of Kronecker products: Geophysics, **85**, no. 2, S115–S134, doi: 10.1190/geo2019-0254.1.
Gregor, K., and Y. LeCun, 2010, Learning fast approximations of sparse coding: Proceedings of the 27th International Conference on Machine Learning, 399–406.
Guitton, A., 2004, Amplitude and kinematic corrections of migrated images for nonunitary imaging operators: Geophysics, **69**, 1017–1024, doi: 10.1190/1.1778244.
Guitton, A., 2012, Blocky regularization schemes for full-waveform inversion: Geophysical Prospecting, **60**, 870–884, doi: 10.1111/j.1365-2478.2012.01025.x.
Guitton, A., G. Ayeni, and E. Díaz, 2012, Constrained full-waveform inversion by model reparameterization: Geophysics, **77**, no. 2, R117–R127, doi: 10.1190/geo2011-0196.1.
Hauptmann, A., F. Lucka, M. Betcke, N. Huynh, J. Adler, B. Cox, P. Beard, S. Ourselin, and S. Arridge, 2018, Model-based learning for accelerated, limited-view 3-D photoacoustic tomography: IEEE Transactions on Medical Imaging, **37**, 1382–1393, doi: 10.1109/TMI.2018.2820382.
He, K., X. Zhang, S. Ren, and J. Sun, 2015, Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification: Proceedings of the IEEE International Conference on Computer Vision, 1026–1034.
He, K., X. Zhang, S. Ren, and J. Sun, 2016, Deep residual learning for image recognition: IEEE Conference on Computer Vision and Pattern Recognition, 770–778.
Hestenes, M. R., and E. Stiefel, 1952, Methods of conjugate gradients for solving linear systems: Journal of Research of the National Bureau of Standards, **49**, 409–436, doi: 10.6028/jres.049.044.
Hornik, K., M. Stinchcombe, and H. White, 1989, Multilayer feedforward networks are universal approximators: Neural Networks, **2**, 359–366, doi: 10.1016/0893-6080(89)90020-8.
Huynh-Thu, Q., and M. Ghanbari, 2008, Scope of validity of PSNR in image/video quality assessment: Electronics Letters, **44**, 800–801, doi: 10.1049/el:20080522.
Ioffe, S., and C. Szegedy, 2015, Batch normalization: Accelerating deep network training by reducing internal covariate shift: Proceedings of the 32nd International Conference on Machine Learning, PMLR, 448–456.
Kaur, H., N. Pham, and S. Fomel, 2020, Improving the resolution of migrated images by approximating the inverse Hessian using deep learning: Geophysics, **85**, no. 4, WA173–WA183, doi: 10.1190/geo2019-0315.1.
Kingma, D. P., and J. Ba, 2014, Adam: A method for stochastic optimization: 3rd International Conference for Learning Representations.
Kühl, H., and M. D. Sacchi, 2003, Least-squares wave-equation migration for AVP/AVA inversion: Geophysics, **68**, 262–273, doi: 10.1190/1.1543212.
Lailly, P., and J. Bednar, 1983, The seismic inverse problem as a sequence of before stack migrations: Conference on Inverse Scattering: Theory and Application, 206–220.
Lambaré, G., J. Virieux, R. Madariaga, and S. Jin, 1992, Iterative asymptotic inversion in the acoustic approximation: Geophysics, **57**, 1138–1154, doi: 10.1190/1.1443328.
LeCun, Y., Y. Bengio, and G. Hinton, 2015, Deep learning: Nature, **521**, 436–444, doi: 10.1038/nature14539.
Li, F., J. Gao, Z. Gao, X. Jiang, and W. Sun, 2020a, Least-squares reverse time migration with sparse regularization in the 2D wavelet domain: Geophysics, **85**, no. 6, S313–S325, doi: 10.1190/geo2018-0763.1.
Li, S., B. Liu, Y. Ren, Y. Chen, S. Yang, Y. Wang, and P. Jiang, 2020b, Deep-learning inversion of seismic data: IEEE Transactions on Geoscience and Remote Sensing, **58**, 2135–2149, doi: 10.1109/TGRS.2019.2953473.
Liu, B., S. Yang, Y. Ren, X. Xu, P. Jiang, and Y. Chen, 2021a, Deep-learning seismic full-waveform inversion for realistic structural models: Geophysics, **86**, no. 1, R31–R44, doi: 10.1190/geo2019-0435.1.
Liu, X., Y. Chen, and L. Huang, 2020a, Least-squares reverse-time migration with a machine-learning-based denoising preconditioner: 90th Annual International Meeting, SEG, Expanded Abstracts, 2993–2997, doi: 10.1190/segam2020-3427177.1.
Liu, Y., Y. Du, and Y. Luo, 2021b, Sparsity-promoting least-squares interferometric migration for high-resolution passive source location: Geophysics, **86**, no. 1, KS1–KS9, doi: 10.1190/geo2020-0084.1.
Liu, Z., Y. Chen, and G. Schuster, 2020b, Deep convolutional neural network and sparse least-squares migration: Geophysics, **85**, no. 4, WA241–WA253, doi: 10.1190/geo2019-0412.1.
Lu, Y., H. Sun, X. Wang, Q. Liu, and H. Zhang, 2020, Improving the image quality of elastic reverse-time migration in the dip-angle domain using deep learning: Geophysics, **85**, no. 5, S269–S283, doi: 10.1190/geo2019-0250.1.
Maier, A. K., C. Syben, B. Stimpel, T. Würfl, M. Hoffmann, F. Schebesch, W. Fu, L. Mill, L. Kling, and S. Christiansen, 2019, Learning with known operators reduces maximum error bounds: Nature Machine Intelligence, **1**, 373–380, doi: 10.1038/s42256-019-0077-5.
Martin, G. S., R. Wiley, and K. J. Marfurt, 2006, Marmousi2: An elastic upgrade for Marmousi: The Leading Edge, **25**, 156–166, doi: 10.1190/1.2172306.
Meinhardt, T., M. Möller, C. Hazirbas, and D. Cremers, 2017, Learning proximal operators: Using denoising networks for regularizing inverse imaging problems: CoRR, abs/1704.03488.
Monga, V., Y. Li, and Y. C. Eldar, 2021, Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing: IEEE Signal Processing Magazine, **38**, 18–44, doi: 10.1109/MSP.2020.3016905.
Nemeth, T., C. Wu, and G. T. Schuster, 1999, Least-squares migration of incomplete reflection data: Geophysics, **64**, 208–221, doi: 10.1190/1.1444517.
Oropeza, V., and M. Sacchi, 2011, Simultaneous seismic data denoising and reconstruction via multichannel singular spectrum analysis: Geophysics, **76**, no. 3, V25–V32, doi: 10.1190/1.3552706.
Paffenholz, J., B. McLain, J. Zaske, and P. J. Keliher, 2002, Subsalt multiple attenuation and imaging: Observations from the Sigsbee2B synthetic dataset: 72nd Annual International Meeting, SEG, Expanded Abstracts, 2122–2125, doi: 10.1190/1.1817123.
Pan, S. J., and Q. Yang, 2010, A survey on transfer learning: IEEE Transactions on Knowledge and Data Engineering, **22**, 1345–1359, doi: 10.1109/TKDE.2009.191.
Park, M. J., and M. D. Sacchi, 2020, Automatic velocity analysis using convolutional neural network and transfer learning: Geophysics, **85**, no. 1, V33–V43, doi: 10.1190/geo2018-0870.1.
Peters, B., B. R. Smithyman, and F. J. Herrmann, 2019, Projection methods and applications for seismic nonlinear inverse problems with multiple constraints: Geophysics, **84**, no. 2, R251–R269, doi: 10.1190/geo2018-0192.1.
Putzky, P., and M. Welling, 2017, Recurrent inference machines for solving inverse problems: CoRR, abs/1706.04008.
Richardson, A., 2018, Seismic full-waveform inversion using deep learning tools and techniques: arXiv preprint arXiv:1801.07232.
Rocha, D., P. Sava, and A. Guitton, 2018, 3D acoustic least-squares reverse time migration using the energy norm: Geophysics, **83**, no. 3, S261–S270, doi: 10.1190/geo2017-0466.1.
Ronneberger, O., P. Fischer, and T. Brox, 2015, U-Net: Convolutional networks for biomedical image segmentation: CoRR, abs/1505.04597.
Schuster, G., 2017, Seismic inversion: SEG.
Schuster, G., and Z. Liu, 2019, Least squares migration: Current and future directions: 81st Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: 10.3997/2214-4609.201901270.
Sun, J., K. A. Innanen, and C. Huang, 2021, Physics-guided deep learning for seismic inversion with hybrid training and uncertainty analysis: Geophysics, **86**, no. 3, R303–R317, doi: 10.1190/geo2020-0312.1.
Sun, J., Z. Niu, K. A. Innanen, J. Li, and D. O. Trad, 2020, A theory-guided deep-learning formulation and optimization of seismic waveform inversion: Geophysics, **85**, no. 2, R87–R99, doi: 10.1190/geo2019-0138.1.
Tarantola, A., 1984, Linearized inversion of seismic reflection data: Geophysical Prospecting, **32**, 998–1015, doi: 10.1111/j.1365-2478.1984.tb00751.x.
Vamaraju, J., J. Vila, M. Araya-Polo, D. Datta, M. Sidahmed, and M. K. Sen, 2021, Minibatch least-squares reverse time migration in a deep-learning framework: Geophysics, **86**, no. 2, S125–S142, doi: 10.1190/geo2019-0707.1.
Vantassel, J. P., K. Kumar, and B. R. Cox, 2021, Using convolutional neural networks to develop starting models for 2D full waveform inversion: arXiv: 2104.01626.
Wang, J., H. Kuehl, and M. D. Sacchi, 2005, High-resolution wave-equation AVA imaging: Algorithm and tests with a data set from the Western Canadian Sedimentary Basin: Geophysics, **70**, no. 5, S91–S99, doi: 10.1190/1.2076748.
Wang, J., and M. D. Sacchi, 2007, High-resolution wave-equation amplitude-variation-with-ray-parameter (AVP) imaging with sparseness constraints: Geophysics, **72**, no. 1, S11–S18, doi: 10.1190/1.2387139.
Wang, J., and M. D. Sacchi, 2009, Structure constrained least-squares migration: 79th Annual International Meeting, SEG, Expanded Abstracts, 2763–2767, doi: 10.1190/1.3255423.
Wang, P., S. Huang, and M. Wang, 2017, Improved subsalt images with least-squares reverse time migration: Interpretation, **5**, no. 3, SN25–SN32, doi: 10.1190/INT-2016-0203.1.

Wang, Z., A. Bovik, H. Sheikh, and E. Simoncelli, 2004, Image quality assessment: From error visibility to structural similarity: IEEE Transactions on Image Processing, **13**, 600–612, doi: 10.1109/TIP.2003.819861.
Whitmore, N. D., 1983, Iterative depth migration by backward time propagation: 53rd Annual International Meeting, SEG, Expanded Abstracts, 382–385, doi: 10.1190/1.1893867.
Witte, P., M. Yang, and F. Herrmann, 2017, Sparsity-promoting least-squares migration with the linearized inverse scattering imaging condition: 79th Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: 10.3997/2214-4609.201701125.
Xiang, K., X. Chen, H. Chen, and Y. Chen, 2016, Least-squares reverse time migration for blended data with a local low-rank constraint: 78th Annual International Conference and Exhibition, EAGE, Extended Abstracts, doi: 10.3997/2214-4609.201601200.
Xu, L., and M. D. Sacchi, 2018, Preconditioned acoustic least-squares two-way wave-equation migration with exact adjoint operator: Geophysics, **83**, no. 1, S1–S13, doi: 10.1190/geo2017-0167.1.
Yang, F., and J. Ma, 2019, Deep-learning inversion: A next-generation seismic velocity model building method: Geophysics, **84**, no. 4, R583–R599, doi: 10.1190/geo2018-0249.1.
Yao, G., and H. Jakubowicz, 2016, Least-squares reverse-time migration in a matrix-based formulation: Geophysical Prospecting, **64**, 611–621, doi: 10.1111/1365-2478.12305.
Yosinski, J., J. Clune, Y. Bengio, and H. Lipson, 2014, How transferable are features in deep neural networks?: Advances in Neural Information Processing Systems.
Yu, S., and J. Ma, 2021, Deep learning for geophysics: Current and future trends: Reviews of Geophysics, **59**, e2021RG000742, doi: 10.1029/2021RG000742.
Zeng, C., S. Dong, and B. Wang, 2017, A guide to least-squares reverse time migration for subsalt imaging: Challenges and solutions: Interpretation, **5**, no. 3, SN1–SN11, doi: 10.1190/INT-2016-0196.1.
Zhang, J., and B. Ghanem, 2017, ISTA-Net: Iterative shrinkage-thresholding algorithm inspired deep network for image compressive sensing: CoRR, abs/1706.07929.
Zhang, Y., and J. Sun, 2009, Practical issues of reverse time migration: True amplitude gathers, noise removal and harmonic-source encoding: Beijing International Geophysical Conference and Exposition, 204–204.

Kristian Torres received a B.S. (2015) in geophysical engineering from Simon Bolivar University and an M.S. (2019) in computational geophysics from the Federal University of Rio de Janeiro. He is currently a Ph.D. student at the University of Alberta working on applications of deep learning to seismic imaging.

Mauricio D. Sacchi received a B.S. (1988) in geophysics from the University Nacional of La Plata, Argentina, and a Ph.D. (1996) from the University British Columbia, Canada. He is currently a professor at the University of Alberta, Canada. He joined the Department of Physics at the University of Alberta in 1997. He is the recipient of the 2012 Medal of the CSEG, the 2014 Central and South America Honorary Lecturer for the Society of Exploration Geophysicists, and 2016 CSEG Distinguished Lecturer. He was the editor-in-chief of the journal GEOPHYSICS from 2016 to 2018. He was the recipient of the 2019 Virgil Kauffman Gold Medal. He is currently associate editor for the journal *IEEE Transactions on Geoscience and Remote Sensing*.