Petroleum Science 21 (2024) 4025–4033

Contents lists available at ScienceDirect

Petroleum Science

journal homepage: www.keaipublishing.com/en/journals/petroleum-science

Original Paper

Generalizable data driven full waveform inversion for complex structures and severe topographies

Mahdi Saadat, Hosein Hashemi *, Majid Nabi-Bidhendi

University of Tehran, Institute of Geophysics, Tehran, 1435944411, Iran

ARTICLE INFO

Article history:
Received 10 January 2024
Received in revised form
3 April 2024
Accepted 6 May 2024
Available online 7 May 2024

Edited by Meng-Jiao Zhou

Keywords:
Deep learning
Generalization
Full waveform inversion
Data-driven inversion
Complex structure

ABSTRACT

Traditionally, simplification has been used in scientific modeling practices. However, recent advancements in deep learning techniques have provided a means to represent complex models. As a result, deep neural networks should be able to approximate the complex models, with a high degree of generalization. To achieve generalization, it is necessary to have a diverse range of examples in the training of the neural network, for example in data-driven FWI, training data should cover the expected subsurface models. To meet this requirement, we porposed a method to create geologically meaningful velocity models with complex structures and severe topography. However, it is important to note that generalization comes with its own set of challenges.

Because of significant variation in topography of the generated velocity models, we need to include this information as an additional input data in training of the network. Therefore, we have transformed the seismic data to a fixed datum to incorporate geometric information. Additionally, we have enhanced the network's performance by introducing a term in the network loss function. Multiple metrics have been employed to evaluate the performance of the network. The results indicate that by providing the necessary information to the network and employing computational techniques to refine the model's accuracy, deep neural networks are capable of accurately estimating velocity models in complex environments characterized by extreme topography.

1. Introduction

Data-driven methods have gained significant attention in the past decade for solving various problems. Deep learning has emerged as a focal point among these methods due to its remarkable ability to handle complex problems. The field of geophysics has also witnessed the application of these innovative techniques to address a wide range of challenges. Notably, deep learning has been employed for tasks such as random noise suppression (Yang et al., 2021; Zhao et al., 2019; Birnie et al., 2021), demultipling (Zhang et al., 2021), surface wave elimination (Kaur et al., 2020; Jia et al., 2018), super-resolution (Li et al., 2020), interpolation (de Groot et al., 2022), seismic data storage and processing workflow (Harsuko and Alkhalifah, 2022, 2023), facies classification (Zhao, 2018; Chevitarese et al., 2018; Liu et al., 2020), and object detection for the geological features like faults, channels, collapse features, and salt (Saadat et al., 2022; Di et al., 2018). Furthermore, deep learning techniques have been applied to inversion problems (Biswas et al., 2018), data assimilation (Moseley et al., 2020), and velocity model building or full waveform inversion (Aryapolo et al., 2018; Li et al., 2019; Song and Alkhalifah, 2020; Zhang and Lin, 2020; Sun and Alkhalifah, 2022). These applications highlight the significance of deep learning methods in addressing key challenges in geophysics.

Constructing a precise and reliable velocity model has consistently played a crucial role in various seismic data processing and interpretation tasks. Full waveform inversion (FWI) is a widely recognized methodology for generating high-resolution velocity models. To ensure a successful FWI processing flow, several key elements are required in the input data. These include preconditioned shot gathers with adequate coverage, an appropriate initial model, an accurate representation of the source signature, and precise acquisition geometry. Despite its promising and elegant theoretical foundation, an efficient implementation of FWI in practical applications faces several obstacles. One of the most

* Corresponding author.
E-mail address: Hashemy@ut.ac.ir (H. Hashemi).

https://doi.org/10.1016/j.petsci.2024.05.002

significant challenges is the strong dependency on the initial model. The initial model serves as the starting point for the FWI process, and for successful convergence, it must closely approximate the true model, especially capturing its low wavenumber components. When the initial model fails to meet this requirement, FWI can converge to a local minimum, resulting in cycle skipping. Thus, constructing an appropriate initial model is one of the most formidable aspects of FWI. Constraining the FWI process, by prior information through the regularization techniques, decreases null space of the inversion algorithm (Virieux and Operto, 2009), however, some regularization methods, depending on the associated parameters may introduce bias into the final results.

On the contrary, data-driven methods often do not require an initial model. Instead, they directly learn the mapping function or inverse operator between the input/output pairs provided. Numerous studies have been conducted in this area, resulting in the development of notable deep learning-based FWI methods. Examples of such methods include InversionNet (Wu and Lin, 2019), SeisInvNet (Li, et al., 2019), VelocityGAN (Zhang and Lin, 2020), and Physics-guided methods (Dhara and Sen, 2022), which have shown great promise on synthetic datasets.

Some other studies have applied their methodologies to real datasets, notably those conducted by Kazei et al. (2021) and Ovcharenko et al. (2022), but they tend to admit smooth velocity models. The propensity of data-driven models to produce smooth velocity models in real-world applications, as opposed to their efficacy in synthetic models, stems from the oversimplification of the velocity models used during training. To address this limitation, concerted efforts have been made to construct more realistic Earth models for the training of deep neural networks. Particularly noteworthy are the studies undertaken by Wu et al. (2020) and Ren et al. (2021).

Most of the data-driven methods are fed only shot-gather data as input, while physics-driven methods need multiple inputs. For an oversimplified model, it might be sufficient to estimate the true model by considering shot gather data as the only input. However, in the real world, the areas where we seek hydrocarbon traps are often characterized by highly complex structures. To successfully utilize data-driven methods, it is crucial to ensure the provision of essential input data for reconstructing a generalizable velocity model. This study aims to make data-driven methods one step closer to real-world applications by incorporating complex structures with significant topographic variations during the training phase. We apply a datum transformation on data to integrate both shot gather data and topography information effectively. Additionally, we employ several loss functions in the training process of our deep convolutional network. These approaches collectively enhance the performance and applicability of data-driven methods in addressing complex scenarios.

2. Methodology

A large number of geologically meaningful models have been generated, which were subsequently used to generate synthetic data. Here, the process of generating velocity models and synthetic data is explained. The model and shot-gathers pairs are then used to train deep neural networks.

2.1. Synthetic models

The synthetic two dimensional models used in this study consist of 200 × 200 (in height and width) grids with a grid size of 10 m. Each model contains a variable number of randomly chosen events, ranging from 10 to 60. Once the number of events is determined, we establish a sequential timeline to represent the nature of each event. These events are selected from a predefined set, which includes deposition, deformation, and erosion. Notably, the likelihood of selecting a specific event from this set varies depending on its position in the timeline. For instance, erosion is the most probable event after a deformation phase, while deposition holds a higher chance following an erosion phase.

The foundational assumption is based on Steno's principles during the depositional events, suggesting that the layers are initially deposited horizontally. To determine the thickness of each layer, a random number is drawn from a normal distribution, in which the minimum is 20 m. The average of this distribution is calculated as the model's height divided by the number of layers plus one. The standard deviation of the distribution is set to half of the average thickness for each model. Thus,

$$Thickness = \max\left(\mathrm{rand}\left(\mathrm{N}\left(\mu = \frac{Model\,height}{Number\,of\,layers + 1}, \sigma = \frac{\mu}{2}\right)\right), 20\right). \quad (1)$$

During deformation events, the first step is to select randomly a number between 2 and 20, representing the number of control points for deformation. Subsequently, the positions of these points are chosen along the x-axis in a random manner, ensuring a minimum grid difference of 3 between them. The amount of deformation at these control points is determined by sampling from a normal distribution. The distribution has an average of 0 and a standard deviation equal to the average thickness of the layers. Deformation surface is then calculated by interpolating the deformation values at the control points. This resulting surface influences the layers starting from a base level up to the youngest layer in the model. The base level also could be either an erosion surface or a strata surface. When it comes to erosion events, constructing the erosion surface closely resembles the costruction of the deformation surface. However, there is one fundamental difference: the resulting surface in erosion events must invariably move downwards.

The construction of the models can be generally divided into two main parts: sketching and shading. During the sketching phase, the structural framework of the model is established. In the shading phase, however, the velocity of each layer is determined.

To determine the velocity of the layers, an increasing velocity trend with depth is initially defined for each model. The average velocity value of each layer is then calculated by adding a random perturbation within the interval [−2000 m/s, +2000 m/s] to the trend. However, the lower and upper bound trends constrain the final velocity value, ensuring it remains within an acceptable range. Next, a random number of control points within the layer are selected, and the velocity value at these points is allowed to vary within a range of ±500 m/s from the average velocity of the layer. Subsequently, the velocity distribution within the layer is interpolated based on the velocities at the control points using a 2D interpolation method. The layer boundaries can exhibit either abrupt changes or gradual transitions, achieved through smoothing techniques applied between the adjacent layers.

This approach for generating velocity models demonstrates our efforts to create geologically significant models that exhibit a high level of randomness and flexibility. This approach allows us to generate diverse models encompassing various complex structures. Although faults are not explicitly represented in the models, the deformation phase captures abrupt structural changes corresponding to faulted regions. The topographic surface in the proposed algorithm is formed by the interaction of sedimentation, deformation and erosion, which is capable of generating flat to severe topographic surfaces. Fig. 1 shows the flowchart of the

velocity model generation process using the porposed algorithm.

2.2. Synthetic data

In total, we generated over 13,000 models. Each model consists of 10 source points distributed evenly along the horizontal direction. A fixed spread of 100 receivers was used for each source, with a regular spacing of 20 m between receivers along the horizontal direction. Regardless of the topography of the models, which varies from flat surfaces to severe topography, the vertical coordinate of the sources and receivers follows the topography surface.

Ricker wavelets were employed in the simulations, with a central frequency ranging from 15 to 25 Hz. A constant phase rotation within the interval of [−180, +180] was also applied on the wavelet. The central frequency and phase rotation values were randomly selected from uniform distributions. In order to simulate the seismic experiment on the synthesized models, the acoustic wave equation was utilized. For the numerical approximation of the wave equation, an explicit 8th-order finite difference stencil was used for spatial derivatives, while a second-order finite difference stencil was used for temporal derivatives (Eq. (3); Dablain, 1986).

$$\frac{\partial^2 \mathbf{P}}{\partial t^2} = \mathbf{V}^2 \nabla^2 \mathbf{P} + \mathbf{w}(t, \mathbf{x}), \tag{2}$$

$$\mathbf{P}_{i,j}^{n+1} - \left(2 + \frac{\alpha^2 \nabla^2}{\mathbf{S}_{i,j}^2}\right) \mathbf{P}_{i,j}^{n} + \mathbf{P}_{i,j}^{n-1} = \mathbf{w}_{i,j}^{n}, \alpha^2 = \frac{\Delta t^2}{\Delta x^2}, \tag{3}$$

where **P** is the pressure field, **S** is the slowness model, **w** is the source signature distrusted in time and space, Δt is the time step, Δx is grid size, ∇^2 is a discrete Laplacian operator, and n, i, and j are indices in time, depth and horizontal axes, respectively.

Each shot was recorded for 3 s, with a sampling rate of 3 ms. Table 1 summarizes the parameters used in the forward modeling process. To integrate acquisition geometry data as an input for the deep learning network, we shifted the shot gathers to a fixed datum. This datum was chosen so that the height of all models reachs to 2000 m. Additionally, a replacement velocity of 1500 m/s was applied to transfer the traces to the fix datum level. Consequently, the static shift amount for each trace was calculated as follows:

$$t_{\text{shift}} = \frac{(h - z_s) + (h - z_r)}{v_{\text{rep}}}, \tag{4}$$

where h is the elevation of the supposed datum, z_s and z_r are source and receiver elevations, and v_{rep} is replacement velocity.

Part (A) of Fig. 2, illustrates the total static time shift, represented by vertical black lines, which is a combination of shot static and receiver static for each trace. The depicted velocity model is generated by the proposed method in previous section. The solid black lines show the structural framework of the model while velocity values correspond to the colormap. Part (B) displays the acquired shot gathers, showcasing the recordings of the receivers for each shot. Finally, part (C) exhibits the shot gathers that have been transferred to the fixed datum, serving as the main input for the deep networks.

2.3. Conventional FWI

In a conventional FWI process we need following items as input data: pre-conditioned shot gathers, initial model, source wavelets, and the acquisition geometry. The synthetic shot gathers will be calculated based on the initial model, which subsequently will be compared to the real shot gathers in a loss function. The gradient of the loss function is utilized to update the model iteratively. Additionally, the wavelet and acquisition geometry are employed in the simulation phase.

It is important to note that if the initial model deviates significantly from the true model, the convergence of the process is compromised. Hence, we encounter the challenge of constructing reliable initial models to avoid the cycle-skipping issue. Apart from the strong dependence of the resulting model on the initial model and the complexity associated with building an accurate initial model, the computational time required for the FWI process poses another obstacle to its efficient implementation. Numerous promising approaches have been proposed to address these challenges in FWI, however, most of these methods make certain assumptions about the model and rely on approximations that significantly impact the resulting model, introducing a certain level of bias to the problem.

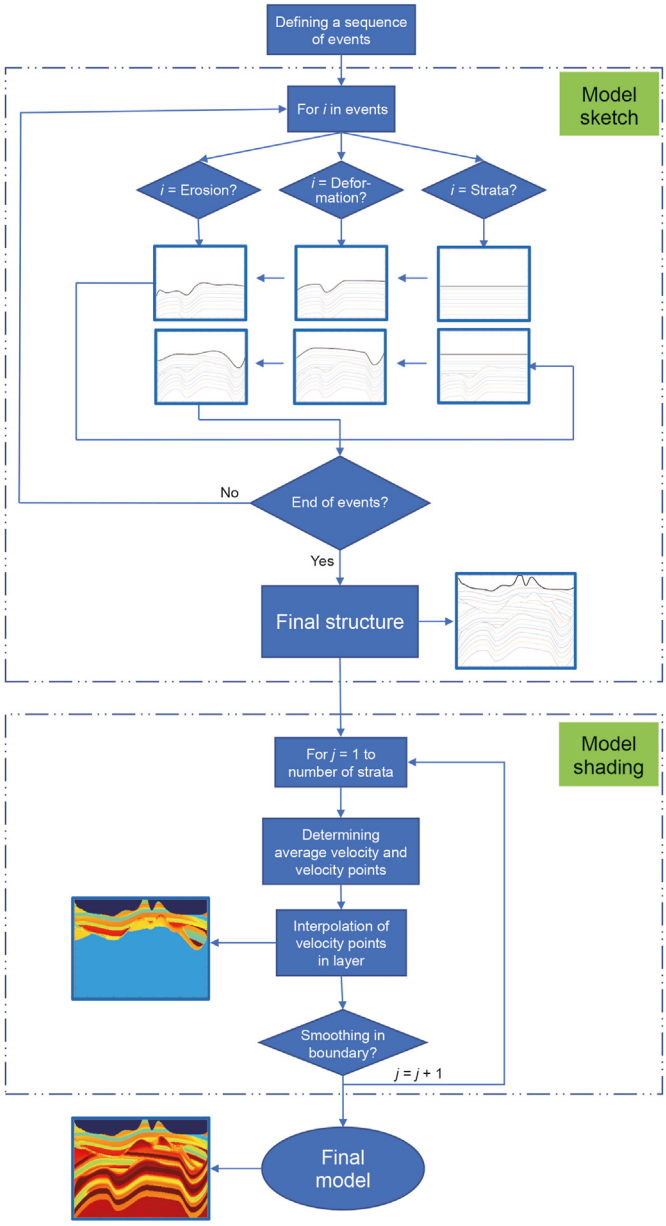

Fig. 1. Flow-chart of generating geologically meaningful velocity models, which consists of two parts: sketching and shading.

Table 1
Parameters used in forward modeling process.

Parameter	Number of shot-points	Shot interval, m	Number of channels, Traces	Reciever interval, m	Sample rate, s	Number of samples
Value	10	220	100	20	0.003	1000

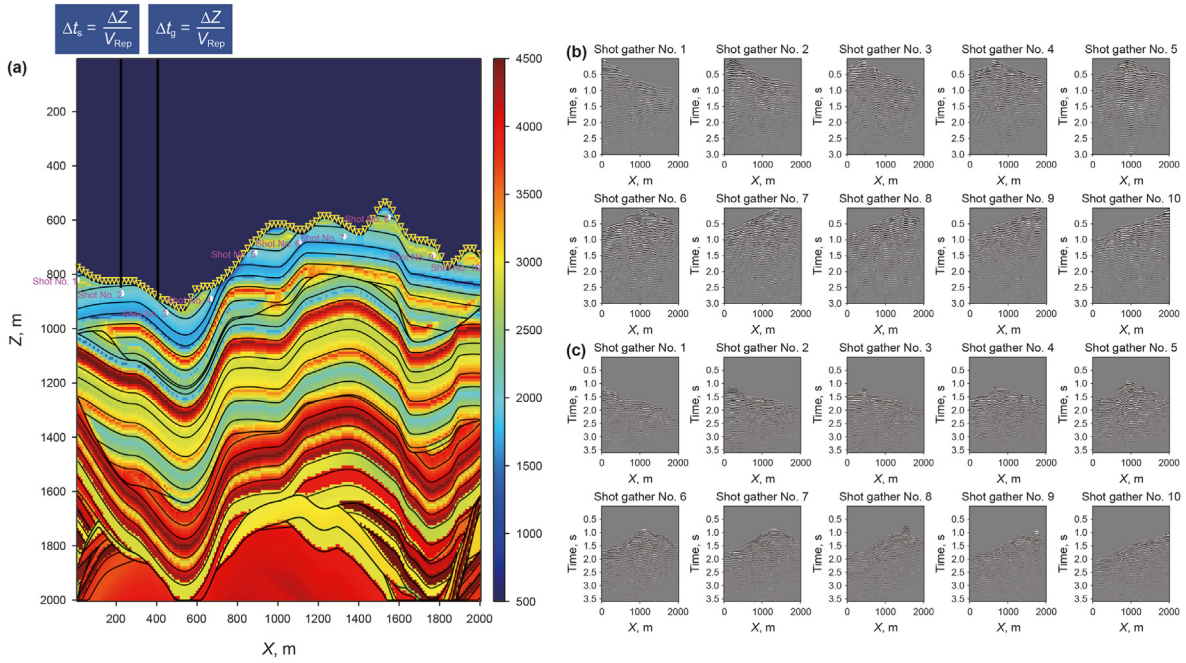

Fig. 2. **(a)** A model from the training set, solid black lines show the Sketch or framework of the model which is subsequently Shaded by determination of their velocity values. The location of the shots are shown by the white circles in model while the location of the receivers are shown by the yellow triangles on top of the model surface. **(b)** Synthesized shot-gathers which are computed by solving the acoustic wave equation for the model in part (A). **(c)** Synthesized shot-gathers after transferring data to fix datum on top of the model using a static shift for each trace which consists of shot static and receiver static (vertical black line in part (A)).

Table 2
Pseudo-code of conventional FWI.

Line number	**Variable definition:**	**Sub-routines:**
	$\mathbf{S}, \mathbf{S}_0$ are slowness model and initial model of slowness. α, α_0 are step length and initial value for step length. $\mathbf{P}, \mathbf{d}^{(\mathrm{pre})}$ and $\mathbf{d}^{(\mathrm{obs})}$ are pressure wavefield, predicted data and observed data. $\mathbf{w}, \mathbf{G}$ are source wavelet(s) and geometry of acquisition. $\mathbf{P}_{\mathrm{Inv}}$ is back propagated pressure wavefield. **grad**, **dir** are gradient and update direction. k is index. **M** is absorbing boundary condition operator.	**Forward**: Acoustic wave propagation operator. **StepLength**: Calculates the numerical step length. **abc**: Absorbing boundary condition. **MSE:** Mean square error. **CG**: Conjugate gradient direction.
	FWI:	$[\mathbf{P}, \mathbf{d}] = \mathbf{Forward}(\mathbf{S}, \mathbf{w}, \mathbf{G}, \mathrm{start}, \mathrm{end}, \mathrm{inc})$:
1	$\mathbf{S} = \mathbf{S}_0$	$\mathbf{M} = \mathrm{abc}(\mathbf{S})$
2	$\mathrm{k} = 0$	**for** $\mathrm{u} = 1 : \mathrm{nshots}$
3	$\alpha = \alpha_0$	for $\mathrm{i} = \mathrm{start} : \mathrm{inc} : \mathrm{end}$
4	**while** $\mathrm{Error} > \varepsilon$	$\mathbf{P}(\mathrm{i}, \mathrm{u}) = \left(2 + \frac{\alpha^2 \nabla^2}{\mathbf{S}^2}\right)\mathbf{P}(i - \mathrm{inc}, \mathrm{u})*\mathbf{M} -$
5	$[\mathbf{P}, \mathbf{d}^{(\mathrm{pre})}] = \mathrm{Forward}(\mathbf{S}, \mathbf{w}, \mathbf{G}, 2, \mathrm{nt} - 1, 1)$	$\mathbf{P}(\mathrm{i} - 2 \times \mathrm{inc}, \mathrm{u})*\mathbf{M} + \mathbf{w}(\mathrm{i}, \mathrm{u})$
6	$\mathbf{res} = \mathbf{d}^{(\mathrm{pre})} - \mathbf{d}^{(\mathrm{obs})}$	
7	$\mathrm{Error} = \mathrm{MSE}(\mathbf{res})$	$\mathbf{d}(\mathrm{i}, \mathrm{u}) = \mathbf{P}(\mathrm{i}, \mathrm{u})*\mathbf{G}$
8	$[\mathbf{P}_{\mathrm{Inv}}, \sim] = \mathrm{Forward}(\mathbf{S}, \mathbf{res}, \mathbf{G}, \mathrm{nt} - 1, 2, -1)$	
9	$\mathbf{grad} = \frac{\mathrm{sum}(\mathrm{xcorr}_{\mathrm{lag}=0}(\mathbf{P}, \mathbf{P}_{\mathrm{Inv}}))}{\mathrm{sum}(\mathbf{P}_{\mathrm{inv}}^2)}$	$\alpha = \mathrm{StepLength}(\mathbf{S}, \mathbf{w}, \mathbf{G}, \mathbf{d}^{(\mathrm{obs})}, \mathrm{dir}, \mathrm{Error}, \alpha)$
10	$\mathbf{dir} = \mathrm{CG}(\mathbf{grad}, \mathbf{dir}, \mathrm{k})$	$\mathrm{e}(1) = \mathrm{Error}$
11	$\alpha = \mathrm{StepLength}(\mathbf{S}, \mathbf{w}, \mathbf{G}, \mathbf{d}^{(\mathrm{obs})}, \mathbf{dir}, \mathrm{Error}, \alpha)$	**for** $n = 1 : 2$
12	$\mathbf{S} = \mathbf{S} + \alpha \times \mathbf{dir}$	$[\sim, \mathbf{d}^{(\mathrm{pre})}] = \mathrm{Forward}(\mathbf{S} + \alpha \times \mathrm{n} \times \mathbf{dir}, \mathbf{w}, \mathbf{G}, 2, \mathrm{nt} - 1, 1)$
13	$\mathrm{k} = \mathrm{k} + 1$	$\mathrm{res} = \mathbf{d}^{(\mathrm{pre})} - \mathbf{d}^{(\mathrm{obs})}$
14		$\mathrm{e}(\mathrm{n}+1) = \mathrm{MSE}(\mathbf{res})$
		$\mathrm{a} = \frac{\mathrm{e}(1) + \mathrm{e}(3)}{2\mathrm{a}^2}$
		$\mathrm{b} = \frac{(\mathrm{e}(2) - \mathrm{e}(3) + 3\mathrm{a}\alpha^2)}{-\alpha}$
		$\alpha = -\frac{\mathrm{b}}{2\mathrm{a}}$

In this study, we employed the conventional FWI methodology to reconstruct velocity model corresponding to three models in our test dataset, aiming to compare the results with those obtained from our data-driven method. A flowchart and the pesodu-code of the used methodology are shared in Table 2 and Fig. 3, respectively. Rest of the parameters are available in Table 3. The initial models for the inversion process were generated by applying a 2D Gaussian filter to smooth the original models. As illustrated in Fig. 4, all the resulting models encountered the issue of being trapped in local minima thus experiencing cycle skipping. To provide a quantitative comparison, Table 5 presents various metrics evaluating the outcomes of FWI and the data-driven methods.

2.4. Data-driven FWI

Data-driven inversion directly learns the mapping function for model reconstruction from the training data:

$$\mathbf{S} = F_{\boldsymbol{\theta}}(\mathbf{d_G}), \tag{5}$$

where $\mathbf{S}$, $\mathbf{d}$ and $\mathbf{G}$ are the slowness model, observed data, and acquisition geometry, respectively. F is the inverse operator, which directly will be learned in the training phase by optimizing the network weights ($\boldsymbol{\theta}$).

In the training phase of data-driven FWI, observed data ($\mathbf{d}_{\mathbf{G}}^{(\mathrm{obs})}$) that is transformed to a fix datum using acquisition geometry information ($\mathbf{G}$) is fed into the network. The predicted slowness model ($\mathbf{S}(\mathbf{d}_{\mathbf{G}}^{(\mathrm{obs})}, \boldsymbol{\theta})$) then will be compared with the true model ($\mathbf{S}^{(\mathrm{true})}$) in a loss function ($L$). The network parameters ($\boldsymbol{\theta}$) is updated iteratively in backpropagation to minimize the loss function:

$$\min_{\boldsymbol{\theta}} L\left(\mathbf{S}^{(\mathrm{true})}, \mathbf{S}\left(\mathbf{d}_{\mathbf{G}}^{(\mathrm{obs})}, \boldsymbol{\theta}\right)\right), \tag{6}$$

In this study, we employed three U-net models for velocity model reconstruction from shot gathers. The first network was trained using raw shot gathers as input. On the other hand, the main input of second and third networks is pre-conditioned shot gathers (shot gathers that were transferred to a fixed datum to incorporate topography information). The distinction between second and third networks lies in their loss functions. The second network employed a mean square error (*MSE*) loss, which measured the discrepancy between the true and predicted models. In contrast, the third network employed a combined loss function that incorporated both *MSE* and *SSIM* (Structural similarity index measure), which are given as follows,

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{x \times z}\left(\mathbf{S}_{\mathrm{pre}} - \mathbf{S}_{\mathrm{true}}\right)^2, \tag{7}$$

$$SSIM = \frac{1}{n}\sum_{i=1}^{n}\left(1 - \left(\frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)}\right)\right), \tag{8}$$

$$Loss = MSE + \lambda \times (1 - SSIM), \tag{9}$$

Eq. (7) represents the mean square error (*MSE*), where z, x, and i denote the height, width, and number of models, respectively. Eq. (9) shows the loss function of third network in which *MSE* is mean square error (Eq. (7)), *SSIM* is structural similarity index, λ is weight for the second term, and *SSIM* is expanded in Eq. (8) in which μ_x, μ_y, σ_x, σ_y, σ_{xy}, c_1, and c_2 represent the local mean of the true image, the local mean of the predicted image, the local variance of the true image, the local variance of the predicted image, the covariance of the local matrices of the true and predicted images, and two constants, respectively.

Fig. 5 illustrates the architecture of the U-Net used for Networks 2 and 3. The only difference in Network 1 is the length of the time axis of the image input layer, which will be set to 1000. This difference arises from the fact that the gathers were not moved to a fixed datum in this network.

The network is a fully convolutional encoder-decoder structure with a depth of 5. In the contraction path, the time axis is initially compressed in the first block, while the remaining convolutional blocks follow the CRCRP (C: Convolution layer, R: ReLU layer, P: Pooling layer), pattern. Each block doubles the number of feature maps, while the max pooling layer reduces the dimension of the feature maps by half through the stride property of this layer. In the expansion path, the blocks follow the TCRCR (T: Transposed convolution layer) pattern. The number of feature maps in each block is obtained by concatenating half of the feature maps from the previous layer plus the corresponding maps from the contraction path. The transposed convolution layers with a stride size of [2 2] then doubles the dimensions of the feature maps.

The hyper-parameter sets used to train the three networks are summarized in Table 4. Comparison of the loss curves of the networks for the training dataset in Fig. 6, reveals that all of the networks have a monotonously decreasing loss curve for this part of data, while, the validation curves show the superiority of Networks

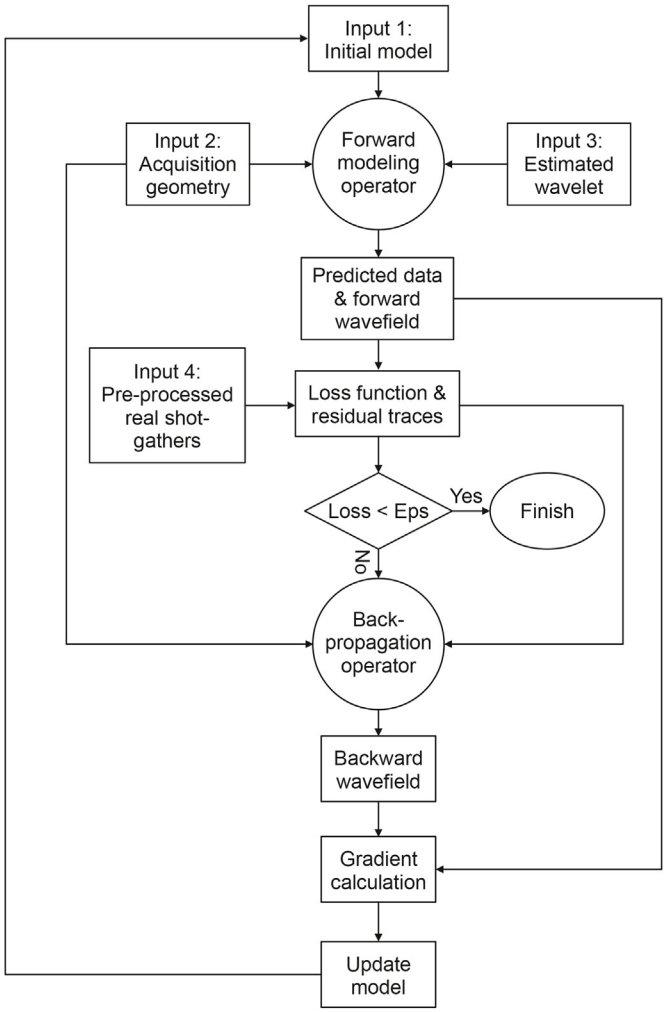

Fig. 3. Flowchart of conventional FWI.

Table 3
Associated parameters used for conventional FWI.

Parameter	Wavelet	Number of iterations	Frequency band of seismic data
Value	Ricker (15 Hz)	50	7–25 Hz

2 and 3 over Network 1. In other words, Network 1 overfits the training data due to a lack of essential information in the input data. The second column of Fig. 7, also shows that the first network fails to accurately reconstruct the velocity model or capture the topographical surface of the models from our test data set. In the third column, which illustrates the results of Network 2, the topographical surfaces and velocity models are well resolved, although there are still some deficiencies, particularly in sharp boundaries. However, these deficiencies are mostly addressed in the predictions of Network 3, as shown in the fourth column. Table 5 compares the results of different methods using well-known numerical metrics, including L1 norm, L2 norm, peak signal-to-noise ratio (*PSNR*), and structural similarity index. According to these metrics, it is evident that both the conventional FWI and Network 1 could not properly reconstruct the models from the test dataset for all three models. However, by incorporating geometrical information into Network 2, there was a significant decrease in the error metrics, and both PSNR and SSIM values increased. Furthermore, Network 3, which aims to recover the true velocity value of each pixel and capture the structural details, achieved even more accurate model reconstruction.

3. Discussion

Simplifying the velocity modelsfor training of a data-driven FWI method improves the robustness and performance of the deep network when using the shot gathers as the only input. In contrast, a physics-driven FWI requires multiple inputs to run the process, including shot gathers, initial model, source wavelet, and acquisition geometry. In the data-driven method, it is assumed that other inputs can be learned from the shot gathers or they are common across all models, and the algorithm will memorize them. The good news is that some of those inputs which are essential for a physics-driven method, could be learned from shot gathers and some others might be recognized as common features. For example, the initial model can be learned from the shot gathers through the kinematic of the seismic data, e.g. first breaks, and the source wavelet is generally consistent within a survey. However, the bad news is that topography and subsequently the acquisition geometry generally exhibit significant changes, especially in onshore

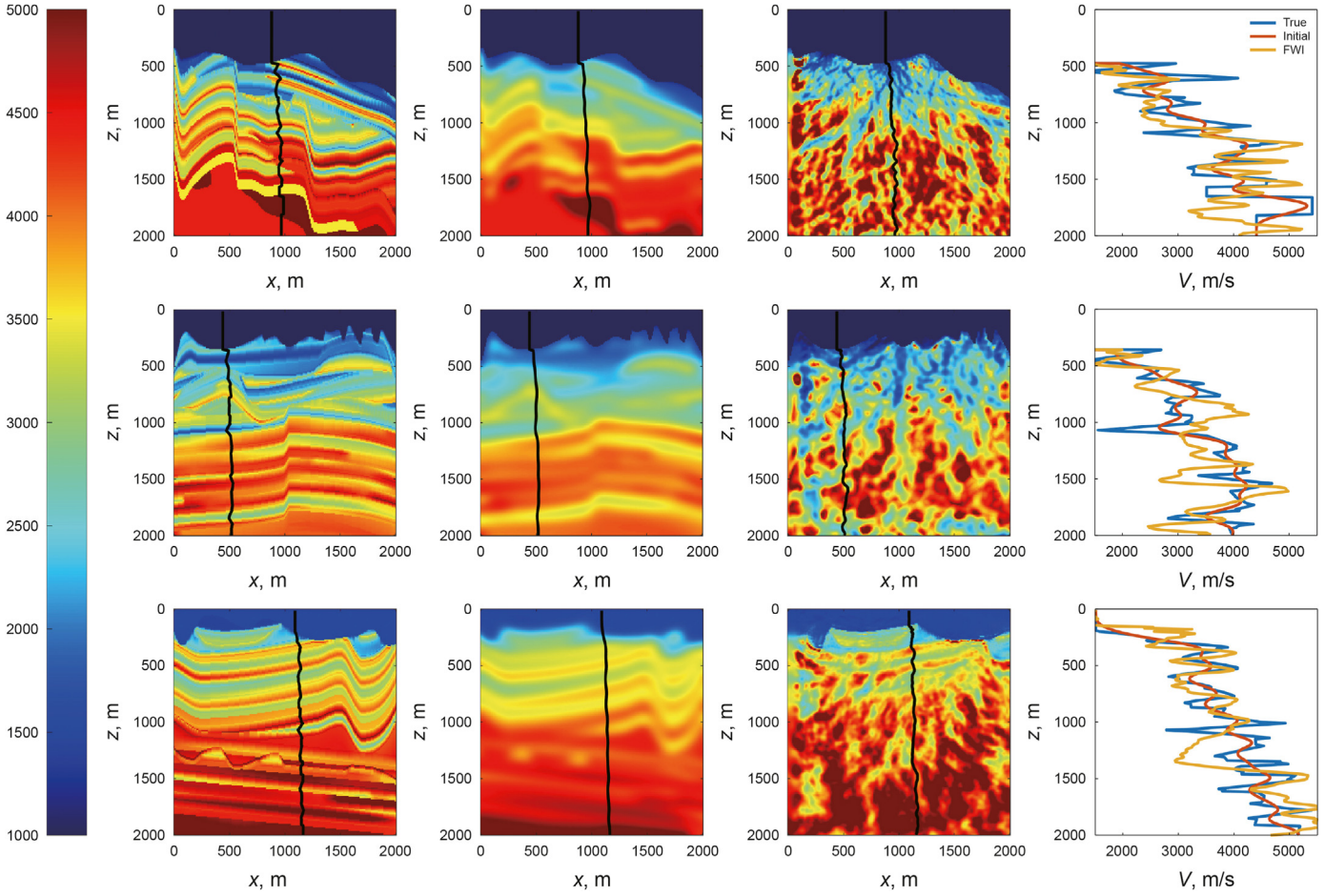

Fig. 4. The first column from left to right shows three true models from the test dataset; The second column shows the corresponding initial models for the conventional FWI process. The third column shows the results of the FWI for each model, in which all are cycle skipped, and the last column compares True, initial, and resulting models on a randomly selected profile (black vertical well-log).

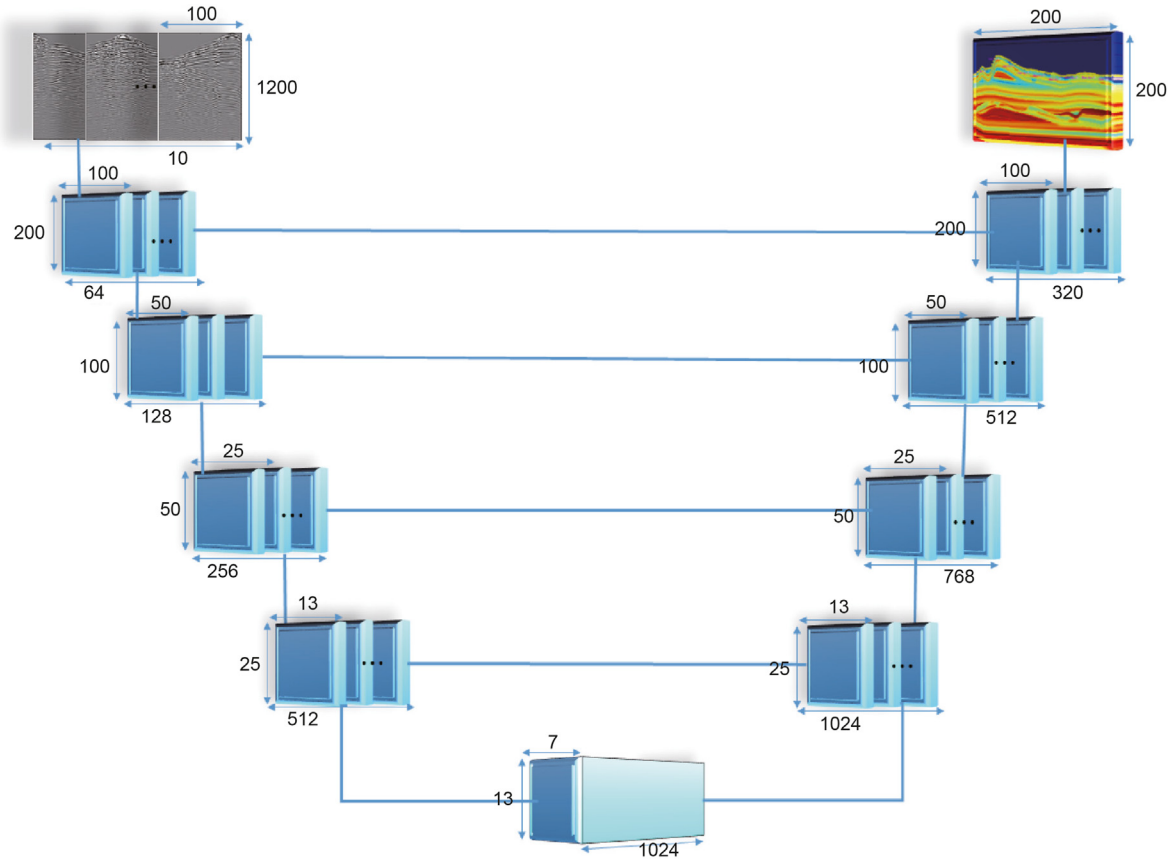

Fig. 5. The architecture of the U-Net and details on the layers used for the prediction of the velocity model from seismic data. The number on top of each layer is the horizontal dimension, while the number on the side is the vertical dimension, and the one in the bottom is the number of channels.

Table 4
Training hyper-parameters of the 3 networks.

	Number of epochs	Mini batch size	L2 regularization	Initial learning-rate	Learning-rate drop	Learning-rate period (epochs)	Optimizer	Loss function
Network 1	200	6	1e−3	1e−4	0.5	10	Adam	*MSE*
Network 2	200	6	1e−3	1e−4	0.5	10	Adam	*MSE*
Network 3	200	6	1e−3	1e−4	0.5	10	Adam	Eq. (2)

Table 5
Comparison of velocity model prediction performance of different methods for three models of test dataset in Figs. 2 and 4 based on different metrics.

	Metric	Conventional FWI	Network 1	Network 2	Network 3
Model 1	L2	1196.53	950.52	441.16	248.76
	L1	579.38	458.36	199.18	104.28
	PSNR	17.01	19.01	25.68	30.65
	SSIM	0.7416	0.8281	0.9302	0.9693
Model 2	L2	1051.41	895.88	275.47	152.64
	L1	555.15	455.31	131.36	69.77
	PSNR	18.13	19.52	29.77	34.89
	SSIM	0.8100	0.8626	0.9665	0.9867
Model 3	L2	991.48	1030.44	396.97	202.22
	L1	532.18	584.24	203.69	97.26
	PSNR	18.64	18.31	26.59	32.45
	SSIM	0.8043	0.8425	0.9395	0.9756

surveys. Therefore, the topography should not be considered as a common feature, so we need to incorporate topography information into the input data to ensure the network's generalizability. In case of a specific type of topography, such as a flat surface, the network could memorize it as a common feature and there would be no need to include topography information in the network's input data. Nevertheless, the resulting model would only be applicable to the same type of topography. Generally, the learned mapping function retains its generalizability as long as the common features remain valid across the validation and test datasets, otherwise overfitiing to the training dataset is unavoidable.

MSE term in the loss function aims to contribute to recovering the true velocity value at each pixel independently from other pixels. On the other hand, the structural similarity index (*SSIM*) is a measure that assesses the combined effect of three characteristics: luminance, contrast, and structure, providing an overall evaluation of image similarity (Zhou et al., 2004). The third network that uses the combination of the *MSE* and *SSIM* as loss function demonstrated superior performance over other networks due to the combined

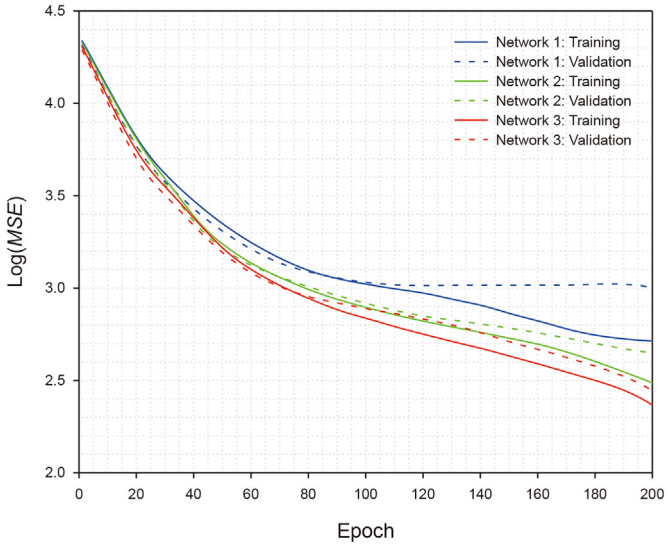

Fig. 6. Comparison of loss function of training and validation datasets for different networks versus training-epochs. Note that logarithm of loss functions are shown to be more distinctive. Loss function of Network 3 contains an additional term but here only logarithm of MSE part is shown to be comparable with other curves.

effects of *MSE* and *SSIM*. This combination improved performance, particularly in the deeper parts of the velocity models.

4. Conclusion

We conducted an extensive model generation process encompassing a wide range of topography, from flat to severe. The generated dataset also includes a range of models from simple to complex structures. Subsequently, we synthesized the shot gathers, which serve as the primary input for our data-driven method. To incorporate surface topography information into the input data, we performed a pre-conditioning step by transferring the shot gathers to a fixed datum using a constant replacement velocity, which acts as a common feature. In the next step, we trained three networks using the generated dataset. The first network utilized the shot gathers without any pre-conditioning as main input. On the other hand, for input of second and third networks, we used the shot gathers that are transferred to the fixed datum.

The second and third networks differ in terms of their loss functions. Loss function of the second network is mean square error (*MSE*), while for third network a combination of *MSE* and structural similarity index (*SSIM*) is assumed to be loss function. Additionally, we conducted experiments using the conventional physics-based FWI method on the selected models from our test dataset to compare the results with the outputs of our data-driven method. To assess the performance of different algorithms in reconstructing velocity models, we employed various metrics such as the first and second norm of model residuals, peak signal-to-noise ratio (*PSNR*), and *SSIM*. The first network failed to recover the velocity model on the test dataset, however, the results obtained from the second and third networks closely resembled true models, with the difference

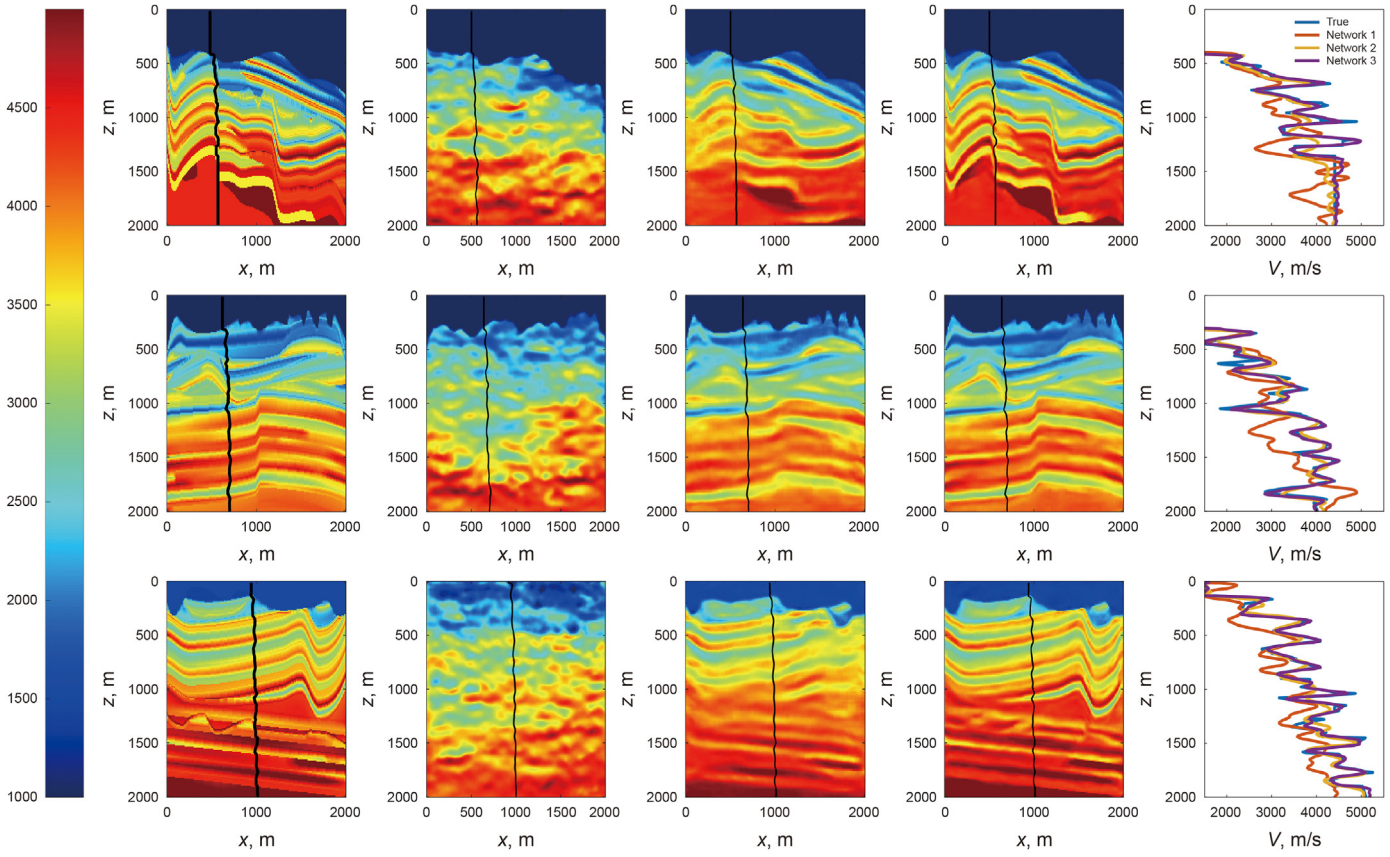

Fig. 7. The first column from left to right shows three true models from the test dataset, The second column shows the prediction of Network 1, The third column shows the prediction of Network2, the fourth column shows the prediction of Network 3, and the last column compares True, and predicted models by different Networks on a randomly selected profile, given by the black vertical line.

that the outputs of the third network performed better in retrieving local structures and greater depths. The physics-driven FWI failed to converge to the true models due to its dependency to the initial model and the occurrence of cycle-skipping problem.

CRediT authorship contribution statement

Mahdi Saadat: Writing – original draft, Methodology, Formal analysis, Conceptualization. **Hosein Hashemi:** Supervision. **Majid Nabi-Bidhendi:** Supervision.

Declaration of competing interest

The authors whose names are listed below, have no conflicts of interest to declare. All co-authors have seen and agree with the contents of the manuscript and there is no financial interest to report. We certify that the submission is original work and is not under review at any other publication.

References

Arayapolo, M., Jennings, J., Alder, A., Dahlke, T., 2018. Deep learning tomography. Lead. Edge. https://doi.org/10.1190/tle37010058.1.

Birnie, C., Ravasi, M., Li, S., Alkhalifah, T., 2021. The potential of selfsupervised networks for random noise suppression in seismic data. Artif. Intell.Geosci. 2, 47–59. https://doi.org/10.1016/j.aiig.2021.11.001.

Biswas, R., Sen, M.K., Das, V., Mukerji, T., 2018. Prestack and poststack inversion using a physics-guided convolutional neural network. Interpretation 7 (3). https://doi.org/10.1190/INT-2018-0236.1.

Chevitarese, D., Szwarcman, D., Mozart, R., 2018. Deep learning applied to seismic facies classification: a methodology for training. EAGE, Saint Petersburg. https://doi.org/10.3997/2214-4609.201800237.

Dablain, M.A., 1986. The application of high-order differencing to the scalar wave equation. Geophysics 51, 54–66. https://doi.org/10.1190/1.1442040.

De-Groot, P., Huck, A., van Hout, M., 2022. Reconstructing seismic images and creating pseudo-3D volumes: a machine learning approach. First Break 40 (2), 57–62. https://doi.org/10.3997/1365-2397.fb2022013.

Dhara, A., Sen, M.K., 2022. Physics-guided deep autoencoder to overcome the need for a starting model for full-waveform inversion. Lead. Edge 41 (6), 375–381, 10.1190./tle41060375.1.

Di, H., Wang, Z., Alregib, G., 2018. Why using CNN for seismic interpretation: an investigation. SEG Int. Expo. 88th Annu. Meet. https://doi.org/10.1190/segam2018-2997155.1.

Harsuko, R., Alkhalifah, T., 2022. StorSeismic: a new paradigm in deep learning for seismic processing. IEEE Trans. Geosci. Rem. Sens. 60, 1–15. https://doi.org/10.1109/TGRS.2022.3216660.

Harsuko, R., Alkhalifah, T., 2023. Optimizing a transformer-based network for a deep learning seismic processing workflow arXiv2308.04739.

Jia, Z., Lu, W., Zhang, M., Miao, Y., 2018. Separating ground-roll from land seismic record via convolutional neural network. In: SEG 2018 Workshop: SEG Maximizing Asset Value through Artificial Intelligence and Machine Learning. https://doi.org/10.1190/AIML2018-16.1. Beijing, China, 17–19 September 2018.

Kaur, H., Fomel, S., Pham, N., 2020. Seismic ground-roll noise attenuation using deep learning. Geophys. Prospect. 68 (7), 2064–2077. https://doi.org/10.1111/1365-2478.12985.

Kazei, V., Ovcharenko, O., Plotnitskii, P., Peter, D., Zhang, X., Alkhalifah, T., 2021. Mapping seismic data cubes to vertical velocity profiles by deep learning: new full-waveform inversion paradigm? Geophysics 86 (5), 1–50. https://doi.org/10.1190/geo2019-0473.1.

Li, J., Wu, X., Hu, Z., 2020. Deep Learning for Simultaneous Seismic Image Super-resolution and Denoising. SEG Technical Program Expanded Abstracts 2020. https://doi.org/10.1190/segam2020-3426412.1

Li, S., Liu, B., Ren, Y., Chen, Y., Yang, S., Wang, Y., 2019. Deep learning inversion of seismic data. IEEE Trans. Image Process. 58 (3), 2135–2149. https://doi.org/10.48550/arXiv.1901.07733.

Liu, M., Jevris, M., Li, W., Nivlet, P., 2020. Seismic facies classification using supervised convolutional neural networks and semi-supervised generative adversarial networks. Geophysics. https://doi.org/10.1190/geo2019-0627.1.

Moseley, B., Nissen-Meyer, T., Markham, A., 2020. Deep learning for fast simulation of seismic waves in complex media. Solid Earth 11, 1527–1549. https://doi.org/10.5194/se-11-1527-2020.

Ovcharenko, O., Kazei, V., Alkhalifah, T., Peter, D., 2022. Multi-task learning for low-frequency extrapolation and elastic model building from seismic data. IEEE Trans. Geosci. Rem. Sens. 60, 1–17. https://doi.org/10.1109/TGRS.2022.3185794.

Ren, Y., Nie, L., Yang, S., Jiang, P., Chen, Y., 2021. Building complex seismic velocity models for deep learning inversion. IEEE Access 9, 63767–63778. https://doi.org/10.1109/ACCESS.2021.3051159.

Saadat, M., Salehi, E., Etminan, M., Yousefzadeh, A., Nezamoleslami, H., 2022. Enhanced collapse feature extraction from high-resolution seismic data using convolutional neural network. Second EAGE Digitalization Conference and Exhibition, Vienna, Austria. https://doi.org/10.3997/2214-4609.202239084.

Song, C., Alkhalifah, T., 2020. Wavefield reconstruction inversion via machine learned functions. SEG 2020, Huston. https://doi.org/10.1190/segam2020-3427351.1.

Sun, B., Alkhalifah, T., 2022. ML-misfit: a neural network formulation of the misfit function for full-waveform inversion. Front. Earth Sci. 10. https://doi.org/10.3389/feart.2022.1011825.

Virieux, J., Operto, S., 2009. An overview of full waveform inversion in exploration geophysics. Geophysics 74 (6), WCC1–WCC26. https://doi.org/10.1190/1.3238367.

Wu, X., Geng, Z., Shi, Y., Pham, N., Fomel, S., 2020. Building realistic structure models to train convolutional neural networks for seismic structural interpretation. Geophysics 85 (4), WA27–WA39. https://doi.org/10.1190/geo2019-0375.1.

Wu, Y., Lin, Y., 2019. InversionNet: A realtime and accurate full waveform inversion with CNNs and continuous CRFs. IEEE Transcriptions of computational imaging. https://doi.org/10.48550/arXiv.1811.07875 arXiv: 1811.0775vol. 2.

Yang, L., Chen, W., Wang, H., Chen, Y., 2021. Deep learning seismic random noise attenuation via improved residual convolutional neural network. IEEE Trans. Geosci. Rem. Sens. 59 (9), 7968–7981. https://doi.org/10.1109/TGRS.2021.3053399.

Zhang, D., de Leeuw, M., Verschuur, E., 2021. Deep learning-based seismic surface-related multiple adaptive subtraction with synthetic primary labels. First International Meeting for Applied Geoscience & Energy Expanded Abstracts. https://doi.org/10.1190/segam2021-3584041.1.

Zhang, Z., Lin, Y., 2020. Data-driven seismic waveform inversion: a study on robustness and generalization. IEEE Trans. Geosci. Rem. Sens. 58 (10), 6900–6913. https://doi.org/10.1109/TGRS.2020.2977635.

Zhao, T., 2018. Seismic facies classification using different deep convolutional neural networks. SEG Int. Expo. 88th Annu. Meet. https://doi.org/10.1190/segam2018-2997085.1.

Zhao, X., Lu, P., Zhang, Y., Chen, J., Li, X., 2019. Attenuating random noise in seismic data by a deep learning approach. arXiv. https://doi.org/10.48550/arXiv.1910.12800.

Zhou, W., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13 (4), 600–612. https://doi.org/10.1109/TIP.2003.819861.