IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 14, 2020, accepted May 24, 2020, date of publication May 27, 2020, date of current version June 26, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2997921*

# A Physics-Based Neural-Network Way to Perform Seismic Full Waveform Inversion

**YUXIAO REN[1], XINJI XU[2], SENLIN YANG[1], LICHAO NIE[2], AND YANGKANG CHEN[3]**
[1]School of Qilu Transportation, Shandong University, Jinan 250061, China
[2]Geotechnical and Structural Engineering Research Center, Shandong University, Jinan 250061, China
[3]School of Earth Sciences, Zhejiang University, Hangzhou 310027, China

Corresponding authors: Xinji Xu (xuxinji1990@163.com) and Lichao Nie (lichaonie@163.com)

This work was supported in part by the National Key Research and Development Plan under Grant 2016YFC0401801, in part by the National Natural Science Foundation of China under Grant 51739007 and Grant U1806226, in part by the Fundamental Research Funds of Shandong University under Grant 2017JC002 and Grant 2018GN019. The work of Yangkang Chen was supported by the Starting Funds from Zhejiang University.

**ABSTRACT** Seismic full waveform inversion is a common technique that is used in the investigation of subsurface geology. Its classic implementation involves forward modeling of seismic wavefield based on a certain type of wave equation, which reflects the physics nature of subsurface seismic wavefield propagation. However, obtaining a good inversion result using traditional seismic waveform inversion methods usually comes with a high computational cost. Recently, with the emerging popularity of deep learning techniques in various computer vision tasks, deep neural network (DNN) has demonstrated an impressive ability in dealing with complex nonlinear problems, including seismic velocity inversion. Now, extensive efforts have been made in developing a DNN architecture to tackle the problem of seismic velocity inversion, and promising results have been achieved. However, due to the dependence of a labeled dataset, i.e., the barely accessible true velocity model corresponding to real seismic data, the current supervised deep learning inversion framework may suffer from limitations on generalization. One possible solution to mitigate this issue is to impose the governing physics into this kind of purely data-driven method. Thus, following the procedures of traditional seismic full waveform inversion, we propose a seismic waveform inversion network, namely SWINet, based on wave-equation-based forward modeling network cells. By treating the single-shot observation data and its corresponding shot position as training data pairs, the inverted velocity model can be obtained as the trainable network parameters. Moreover, since the proposed seismic waveform inversion method is performed in a neural-network way, its implementation and inversion effect could benefit from some built-in tools in Pytorch, such as automatic differentiation, Adam optimizer and mini-batch strategy, etc. Numerical examples indicate that the SWINet method may possess great potential in resulting a good velocity inversion effect with relatively fast convergence and lower computation cost.

**INDEX TERMS** Acoustic wavefield modeling, deep learning inversion, seismic waveform inversion.

## I. INTRODUCTION

Seismic full waveform inversion plays an important role in the estimation of subsurface properties, such as geology, lithology, rock mass quality, etc., and it has been of significant interest to exploration geophysicists for decades [1]–[5]. Waveform inversion is often achieved based on the forward modeling of a particular wave equation to generate the synthetic observation data, which will be further compared with the actual observation data to find the best description of the subsurface properties.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Liu.

Particularly, seismic full waveform inversion (FWI) in its conventional form is to reconstruct the velocity model that is capable of matching the actual recorded data by minimizing the data residual between synthetic data and actual data in a $L_2$ norm [6]–[8]. This inverse problem is usually ill-posed and sensitive to initial model. Thus, the conventional FWI often requires a good estimation of velocity as initial model and utilizes an iterative optimization algorithm for updating the model parameters. Moreover, due to the large amount of model parameters (usually between $10^4$ in 2D and $10^{10}$ in 3D) to invert, the computation cost is another constraining factor that requires consideration in the research and practical application of this method [9].


This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

In recent years, the successful applications of deep learning in various fields like computer vision (*e.g.* [10]–[12]) and natural language processing (*e.g.* [13]–[15]), have attracted attention from scholars in exploration geophysics community. Deep neural networks (DNNs) have demonstrated good application effects on almost all kinds of geophysical data (*e.g.* [16]–[20]). Particularly, some state-of-the-art deep learning techniques have been widely applied in various seismic data problems, such as seismic noise attenuation (*e.g.* [21]–[24]), automatic seismic event picking (*e.g.* [25]–[27]) and seismic structure interpretation (*e.g.* [28], [29]). All these progresses indicate a great potential of DNN techniques and collaboration between the deep learning community and the geophysics community could lead to more achievements.

Especially in the domain of seismic data inversion, the remarkable ability of DNN to accurately simulate a non-linear and complex mapping has demonstrated some advantages over traditional methods. For example, Araya-Polo *et al.* [30] proposed a velocity prediction method using a DNN with the calculated velocity feature semblance as an input and a k-means method to post-process the output. Wu *et al.* [31] proposed a CNN-based network called InversionNet to directly map the raw seismic data to the corresponding seismic velocity model and it achieved good inversion effect on simple fault models with flat or curved subsurface layers. More recently, Li *et al.* [32] deeply analyzed the features of mapping the time-series seismic data to a velocity image and then developed a novel DNN framework called SeisInvNet to perform the end-to-end velocity inversion mapping with enhanced single-trace seismic data as the input. In a word, various DNN frameworks have been adopted into the task of seismic velocity inversion and some of them have already outperformed the traditional FWI on simple velocity models.

However, like all data-driven methods, the performance of the aforementioned DNN-based seismic inversion methods largely depends on the training dataset. Currently, almost all of the DNN-based applications in seismic velocity inversion is to simplify the velocity models into some classes and then design an algorithm to generate enough velocity models for training the deep learning network. Thus, the generalization ability of the proposed deep learning inversion frameworks to a more complex velocity model would be an issue. One way to address this issue is to build or otherwise acquire a set of realistic velocity models which is of a larger scale and contain more complex structures such as salt bodies, faults, several different curved layers with different velocity distributions, etc. However, these kinds of improvements in the aspects of velocity model size and complexity are usually difficult, due to limitations of computation power such as memories, CPU, GPU and etc. Thus, the extension of current deep learning inversion philosophy to practical application may require extensive research work in the future.

On the other hand, the promising inversion performance of traditional FWI in synthetic and actual complex models, indicates the great power of considering the governing wave equation in seismic inversion. Thus, another possible solution to generalize the seismic deep learning inversion to large complex models is to incorporate the relevant physical nature rules into the purely data-driven methods and form a physics-based deep learning concept [33], [34]. In this concept, a seismic forward modeling operator needs to be designed and imposed into the deep learning inversion architecture, which will relate the basic generation rules and features of seismic observation data with the learning process, and as a result, may lead the optimization of network parameters to a more stable and reasonable direction. In fact, since the traditional method achieves seismic waveform inversion based entirely on the physics nature of wavefield propagation, i.e., the wave equation, its implementation from the perspective of deep learning can be considered as a special case of the physics-based deep learning scheme. Thus, the physics-based seismic deep learning inversion may not only have the ability of inverting a complex velocity model like traditional FWI, but also maintain the potential of adopting techniques from the deep learning community. For example, the advanced tools like mini-batch strategy, various optimization algorithms and an easy access to GPU parallel computing techniques will provide novel perspectives and convenience for the study of seismic waveform inversion.

One key factor of the physics-based deep-learning inversion is to build a seismic forward-modeling operator that can be incorporated into the deep-learning inversion network. By convolving with an angle-dependent wavelet, Biswas *et al.* [33] proposed a forward modeling data generation method for updating the classic convolutional neural network (CNN) in the task of elastic pre-stack inversion. Richardson [35] achieved the simulation of acoustic wave equation via a recurrent neural network (RNN) based on TensorFlow platform [36], and undertook an initial attempt on seismic FWI based on some useful deep learning tools.

In this work, we simulate seismic wave equation based on a general DNN architecture using Pytorch [37], and the implementation of a sponge absorbing boundary condition in this framework is quite straightforward. Then for seismic inversion, a comparison of the gradient calculation between the adjoint state method that is widely used in FWI and automatic differentiation in deep learning is conducted for the discrete form of acoustic wave equation. Moreover, seismic waveform inversion is achieved in a neural network optimization way, so that normal deep learning tools can be adopted. As shown in the numerical examples, the proposed method demonstrates satisfactory inversion effect with fast convergence and controllable computation cost. In the end, we discuss the potential of the physics-based seismic deep learning inversion and remark on some further possible research directions along the line.

## II. NETWORK FORMULATION FOR SEISMIC WAVEFIELD EXTRAPOLATION

As stated in the INTRODUCTION section, the seismic forward modeling operator simulated via a neural network is

an essential component for the physics-based seismic deep learning inversion. Currently, one of the most popular and accurate seismic wavefield extrapolation methods is based on wave equation, which can be modified according to the physical property of the wave propagation medium. Usually, for a homogeneous medium with constant density, acoustic wave equation in time domain is used based on a finite-difference discretization approach. Thus, for the convenience of demonstrating our work, we will start with the basic one-dimensional acoustic wave equation and its corresponding network formulation. Similar implementations can be extended to higher dimensions and other types of wave equation.

### A. PROBLEM DEFINITION AND BACKGROUND THEORY

The acoustic wave equation in 1D time-space domain with a source term $s$ can be described as

$$\frac{\partial^2 p}{\partial t^2} = v^2(z)\frac{\partial^2 p}{\partial z^2} + s(t,z), \tag{1}$$

where $t$ and $z$ denote time and depth, respectively, $p(t,z)$ usually denotes the pressure field and $v(z)$ represents the acoustic wave velocity. The finite-difference discretization in the second order gives

$$p_i^{n+1} - (2+\alpha v_i^2\nabla^2)p_i^n + p_i^{n-1} = s_i^{n+1}. \tag{2}$$

Here, $\alpha = \frac{\Delta t^2}{\Delta z^2}$, $v_i$ denotes the acoustic velocity at all $nz$ model gird points and $\nabla^2$ denotes the discrete Laplace operator that conducts spatial discretization of the acoustic wavefield $p(t,z)$. By letting $\mathbf{p}^n = [p_1^n, p_2^n, \cdots, p_{nz}^n]^T$ and $\mathbf{s}^n = [s_1^n, s_2^n, \cdots, s_{nz}^n]^T$ represent the wavefield and source at the time step $n$, respectively, the discrete acoustic wave equation can be further rewritten in matrices as

$$\mathbf{p}^{n+1} = \mathbf{G}\mathbf{p}^n - \mathbf{p}^{n-1} + \mathbf{s}^{n+1}, \tag{3}$$

where $\mathbf{G} = 2\mathbf{I} + \mathbf{A}\mathbf{L}$ and $\mathbf{A} = diag\{\alpha v_i^2\}$, $\mathbf{L}$ is the Laplacian matrix of the form

$$\mathbf{L} = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}_{nz\times nz}. \tag{4}$$

Notice that both matrix $\mathbf{A}$ and $\mathbf{L}$ are symmetric, while the matrix $\mathbf{G}$ is not, which means that this discretization form fails to maintain the self-adjoint property of acoustic wave equation in the continuous form and extra attention is required when calculating the inversion gradient based on the corresponding adjoint wave equation.

Furthermore, by denoting $\mathbf{p} = [\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^{nt}]^T$, $\mathbf{s} = [\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^{nt}]^T$, we can obtain the acoustic equation (3) in a more compact form as $\mathbf{B}\mathbf{p} = \mathbf{s}$. Here

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & & & & \\ -\mathbf{G} & \mathbf{I} & & & \\ \mathbf{I} & -\mathbf{G} & \mathbf{I} & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{I} & -\mathbf{G} & \mathbf{I} \end{bmatrix}. \tag{5}$$

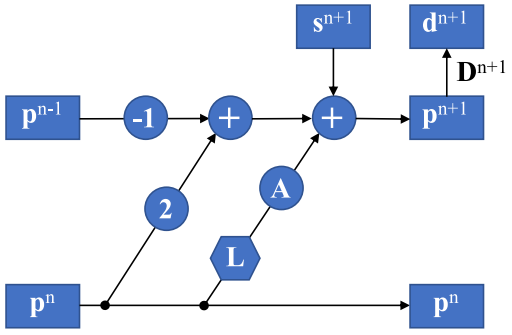



**FIGURE 1.** The network architecture of seismic forward modeling (FM) cells. The hexagon denotes a convolution operation with the Laplacian operator L and the circles represent element-wise operations like addition or multiplication with a number or matrix. Especially, matrix A(v) contains the trainable parameter v of the network.

As for the seismic acquisition geometry, we introduce the matrix $\mathbf{D}_s$ to distribute the source wavelet $\mathbf{w}$ on every wavefield grids for all time steps, that is, $\mathbf{s} = \mathbf{D}_s\mathbf{w}$. Moreover, data collection process is denoted by the observation matrix $\mathbf{D}_r$, which will extract the wavefield value at the predetermined receiver position to an abstract data vector, i.e., $\mathbf{d} = \mathbf{D}_r\mathbf{p} = \mathbf{D}_r\mathbf{B}^{-1}\mathbf{s}$. Especially, when the seismic sources and receivers are set on the same grids, we will have $\mathbf{D}_s = \mathbf{D}_r^T$. Actually, for seismic data recorded on ground surface, the corresponding observation matrix is $\mathbf{D}_r = \begin{bmatrix}\mathbf{D}^1 & \mathbf{D}^2 & \cdots & \mathbf{D}^{nt}\end{bmatrix}$ with submatrix $\mathbf{D}^n$ of the size $nt \times nz$ in the following form

$$\mathbf{D}^1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix},$$
$$\mathbf{D}^2 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \cdots. \tag{6}$$

### B. FORWARD MODELING NETWORK FORMULATION

Generally speaking, seismic wavefiled extrapolation is an iterative process in time domain based on forward modeling operators. In our work, considering the acoustic wave equation denoted in (3), we design a neural network with $nt$ cells and in each cell, wave extrapolation within one time step is performed. That is, seismic wavefield extrapolation for $nt$ time steps can be achieved via the forward calculation of the network. Moreover, the network parameters can be regarded as the velocity values at every grid points and they can be trained to arrive at their optimal points when provided with suitable training setup and dataset.

More specially, the detailed network architecture in each cell is designed by following the operations in the wave equation (3). As shown in Fig. 1, the future wavefield $\mathbf{p}^{n+1}$ is calculated based on the wavefield on the previous two time steps $\mathbf{p}^{n-1}$ and $\mathbf{p}^n$ as well as the future source $\mathbf{s}^{n+1}$. Here, the

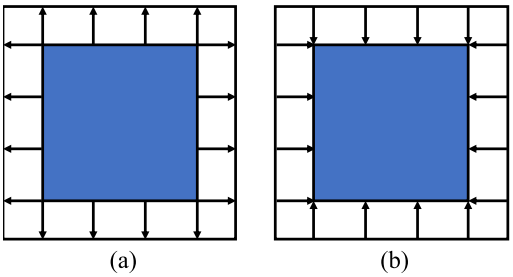



**FIGURE 2.** Illustration map for the (a) padding and (b) stacking operations in absorbing boundary layers. The middle blue area denotes the original velocity model.

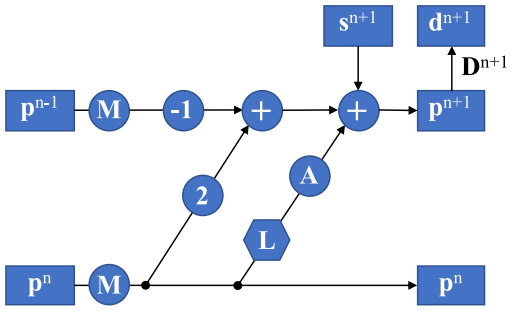



**FIGURE 3.** The network architecture of seismic forward modeling (FM) cells with a sponge absorbing boundary condition. The circle M denotes the sponge absorbing boundary condition, which indicates the "mask" matrix is element-wise multiplying with the corresponding wavefield.

circles represent element-wise operations. For example, the circle with a number in it means element-wise multiplication with that number and the circle **A** means element-wise multiplication between wavefield $\mathbf{p}^n$ and the velocity model $\alpha\mathbf{v}$. The circle with a "plus" sign denotes element-wise addition. Moreover, the hexagon denotes the convolution operation of wavefield $\mathbf{p}^n$ with the discrete Laplace kernel of the corresponding dimension. For example, the vector $[1, -2, 1]$ is for 1D problems and 2D problems will adopt the matrix filter

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \tag{7}$$

The calculation within each cell can be accelerated by parallel computing of the Laplace transform, which can be easily achieved by calling a ready-made function in the Pytorch platform.

### C. ABSORBING BOUNDARY CONDITIONS

Usually, an absorbing boundary condition is considered in seismic wavefield simulation to improve the calculation efficiency by reducing the wavefield reflection from the numerical boundaries. On the other hand, the addition of an absorbing boundary layers should avoid changing the basic architecture of the proposed forward modeling network and maintain its differentiability, which is essential for accurately computing the gradient when updating network parameters. One of the most efficient and convenient ways to mitigate the boundary reflection is to add several sponge layers outside the original velocity models [38], [39].

In fact, adding sponge absorbing boundary layers includes the padding of the outer boundary of the original velocity model (see Fig. 2a) and then element-wise times with an exponentially decaying factor to reduce the wavefield amplitudes in the extended layers. Thus, adding sponge layers can be considered as putting a "mask" over the whole wavefield, and the "mask" is a matrix whose element is one in the original velocity area and the decaying factor in the extended sponge area. With the absorbing boundary matrix **M**, the governing matrix **B** for the acoustic forward modeling can be expressed as

$$\mathbf{B} = \begin{bmatrix} \mathbf{I} & & & & \\ -\mathbf{GM} & \mathbf{I} & & & \\ \mathbf{M}^2 & -\mathbf{GM} & \mathbf{I} & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{M}^2 & -\mathbf{GM} & \mathbf{I} \end{bmatrix}. \tag{8}$$

Fortunately, the realization of the sponge absorbing boundary condition is quite straightforward. As shown in Fig. 3, the two input previous wavefields are multiplied by the "mask" **M** in an element-wise form, which easily achieves the goal of reducing wavefield amplitude without compromising the differentiability of the proposed forward modeling network.

## III. SEISMIC WAVEFORM INVERSION NETWORK

In this section, we build a neural network based on the proposed forward modeling (FM) cells and the velocity model can be updated during the training process. It will be evident to see that the way of optimizing velocity model is similar to the traditional seismic inversion except for some adapted concepts and techniques from the deep learning community.

### A. NETWORK ARCHITECTURE

The aim of training a deep neural network is to find a set of optimal DNN parameters that best fits the corresponding true observation data, and it is usually measured via a loss function. In our case, based on the proposed forward modeling cells, we can build a seismic waveform inversion network, namely SWINet, whose trainable parameters are the subsurface velocity distribution. The SWINet output is the synthetic observation data that will be further compared with the actual observation data to determine the velocity update. Thus for a certain set of seismic data receivers, with the shot positions acting as the network input, we can train the SWINet to fit the actual observation data and obtain a best guess of the subsurface velocity model.

Moreover, as shown in the network architecture illustration map in Fig. 4, the input information about shot position is transformed into seismic wavefield via the matrix $\mathbf{D}_s$. In addition, *nt* forward modeling cells are adopted sequentially to output the synthetic observation data $d_{syn}$ for *nt* time steps.

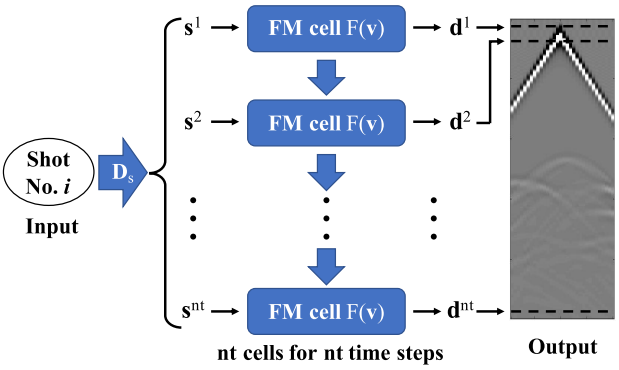



**FIGURE 4.** The network architecture of the seismic waveform inversion network SWINet based on *nt* forward modeling cells.

Considering the fact that seismic data and its corresponding shot position naturally come in pairs, it is not difficult to acquire a training dataset containing seismic data generated by thousands of or even millions of seismic shots in practice.

Similarly to the traditional seismic waveform inversion, for every input data (i.e., single-shot seismic data), the loss function can be defined as

$$\mathcal{L} = \frac{1}{2}\sum_{nt}\sum_{nr}(d_{obs} - d_{syn})^2. \tag{9}$$

Here, $d_{obs}$ and $d_{syn}$ are the observed data and the synthetic data modeled by the neural network, respectively. $nr$ and $nt$ denote the number of receivers and time steps, respectively. In this way, we can optimize the velocity model by training the SWINet using all sorts of deep learning advances, such as the Adam optimizer and mini-batch strategy in the numerical examples. It is worth noticing that the requirement of a relatively good initial velocity model in traditional FWI also applies to the training process of SWINet, which means a good network initialization. It will provide essential low-frequency information for the reconstruction of the velocity model and plays a vital role in mitigating the cycle-skipping problem in seismic waveform inversion [40]–[42].

### B. GRADIENT CALCULATION

In the standard training process of DNN, network parameters are usually optimized using gradient-based methods, which can be found very convenient to use in the deep learning platforms Pytorch and TensorFlow. The gradient can be calculated easily via the built-in automatic differentiation tools, which are based on the backward propagation (BP) theory. On the other hand, the gradient calculation in traditional seismic inversion is based on the adjoint state method and it has been theoretically proved to be coincident with BP gradient in continuous form. Considering the discrete nature of the finite-difference method, we demonstrate our proof in the Appendices VI and VI from the perspective of matrix calculation.

As shown in the Appendix A, the gradient calculation involves computing the adjoint wavefield $\mathbf{B}^{-T}\boldsymbol{\epsilon}$ with the data residual $\boldsymbol{\epsilon}$ as source. Normally, the adjoint wavefield is obtained via the operator $\mathbf{B}^{-T}$, and the adjointness between them should be numerically verified by the dot-product test [43]. When considering the absorbing boundary condition, in addition to adopting the forward modeling operator shown in (8), one should also consider the corresponding adjoint operation of padding the boundary of the original velocity model, which is the stacking of the absorbing layers onto the corresponding boundary of the original velocity model (see Fig. 2b for illustration). This can be understood via matrix transposition where padding corresponds to a vertical vector $\mathbf{a} = [1, 1, \cdots, 1]^T$ and stacking corresponds to $\mathbf{a}^T = [1, 1, \cdots, 1]$. For example, when padding a boundary velocity value $v_i$ for $m$ times, the extended layers will have a velocity value of $v_i\mathbf{a} = [v_i, v_i, \cdots, v_i]^T$ and the stacking operation means $\mathbf{a}^T v_i\mathbf{a} = \sum_m v_i = mv_i$. Thus, there will be abnormal values on the outmost layers of the gradient and the updated velocity model and this influence can be easily neglected by cutting these abnormal layers during the training process.

As mentioned before, using a handcraft code to calculate the gradient for seismic velocity inversion usually requires an adjoint test on the code [44], which is not easy to pass. However, automatic differentiation calculates the gradient based on the chain rule and its algorithm passes the adjoint test automatically. Thus, considering the convenience of implementing gradient calculation in deep learning platform, we choose to use the automatic differentiation way for the gradient computation in our work.

### C. GPU PARALLEL COMPUTATION

Another advantage of using a neural network way to perform seismic waveform inversion is the easy implementation of GPU parallelization based on modern deep learning platforms. In our case, the parallel computation is mainly included in two processes: (a) parallel computing of the Laplacian in acoustic extrapolation via convolution and (b) parallel computing of the wavefield with different sources on different GPUs. Both can be easily implemented by calling the built-in functions in Pytorch. However, using GPU parallel computation also poses restrictions on the size of velocity model due to the limited GPU memory. Thus, the effective utilization of multiple GPUs in large 2D or even 3D velocity models is worth further investigation.

In our work, we achieve the inversion of small 2D velocity models by implementing the SWINet on several GPUs with 24G memory. As for a larger velocity, SWINet can be implemented via CPU with larger memory. Moreover, experience from deep learning indicates that seismic inversion can benefit from the gradient oscillations of the mini-batch strategy and the adaptive gradient momentum of the Adam optimizer, thereby obtaining a good inversion convergence. Actually, the randomness in the gradient calculation is conducive for the gradient trajectory to escape from saddle points or local minima during the optimization process. Considering that Adam optimizer utilizes historical gradients and works

well with mini-batch strategy in various deep learning tasks, we would like to expend their application scope to the field of seismic inversion and impose them in the training process of SWINet. Thus, the basic algorithm for training the SWINet can be summarized in Alg. 1.

**Algorithm 1** Training Process of SWINet

**Inputs:**

$\{\hat{\mathbf{d}}_n\}_{n=1}^{ns}$: a set of real observation data; $\mathbf{v}_0$: initial velocity model; $\mathbf{s}$: seismic source wavefield.

**Notations:**

$\mathbf{p}$: seismic wavefield; $P(\cdot)$: boundary padding of the velocity model; $L(\cdot)$: Laplacian of wavefield; $\odot$: Hadamard product; $\mathbf{M}$: absorbing boundary condition mask.

**Training process:**

**Init:** Set trainable parameter $\mathbf{v} = \mathbf{v}_0$ and $\mathbf{p} = \mathbf{0}$;
**for** $j = 1 : 1 : batchsize$ in GPUs **do**
  $\mathbf{v} \leftarrow P(\mathbf{v})$
  **for** $t = 1 : 1 : nt$ **do**
    $\mathbf{a} \leftarrow \frac{dt^2}{dz^2} \cdot \mathbf{v}^2$
    $\mathbf{p}_{temp} \leftarrow \mathbf{M} \odot \mathbf{p}(t-1)$
    $\mathbf{p}(t) \leftarrow \mathbf{M} \odot \mathbf{p}(t)$
    $\mathbf{p}(t+1) \leftarrow 2\mathbf{p}(t) + \mathbf{a} \odot L(\mathbf{p}(t)) - \mathbf{p}_{temp}$
    $\mathbf{p}(t+1) \leftarrow \mathbf{p}(t+1) + \mathbf{s}(j, t+1)$
  **end for**
  $\mathbf{d}_j \leftarrow \mathbf{p}$(receiver position)
  $loss \leftarrow \mathcal{L}(\mathbf{d}_j, \hat{\mathbf{d}}_j)$
**end for**
$\mathbf{v} \leftarrow Adam(\mathbf{v}, loss, lr)$

## IV. NUMERICAL EXAMPLES

One major goal of building the SWINet is to introduce some popular strategies and techniques from the deep learning community which can be experimented conveniently on seismic waveform inversion via deep learning platforms. In this work, we focus on the approaches of Adam optimizer and mini-batch strategy, and evaluate their application effect with the proposed SWINet on the task of inverting the complex Marmousi model. This experiment is conducted on ten NVIDIA Titan RTX with 24G memory.

### A. DATASET AND TRAINING PARAMETER SETUP

The training dataset is constructed based on the synthetic data simulated from the Marmousi model shown in Fig. 5a. This model is down-sampled to a grid size of $112 \times 384$ in order to fit the GPU memory. The smoothed velocity model in Fig. 5b will be used for the initialization of the SWINet parameters. Moreover, the grid spacing in both horizontal $X$ and vertical $Z$ directions is 10m and time stepping interval is 1ms. We set 20 shots and 384 receivers evenly deployed on the ground surface. The dominant frequency of seismic source Ricker wavelet is 10Hz.

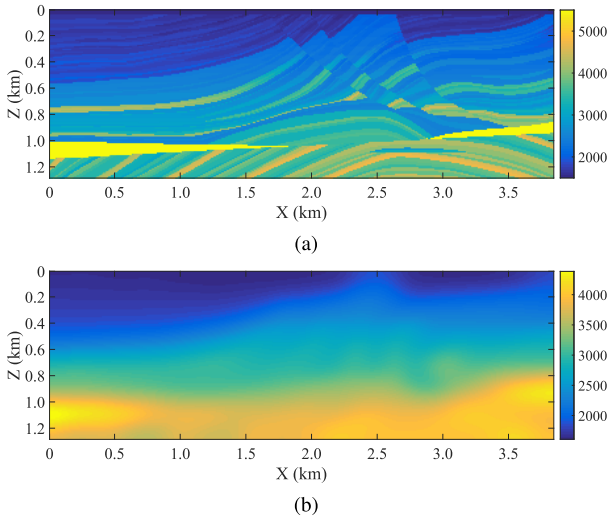



**FIGURE 5.** The Marmousi model. (a) True velocity and (b) smoothed velocity. The scale-bar indicates the velocity value in *m/s*.

As for the network hyper-parameters, we follow the discussion in [34] and set the learning rate $lr = 40$ for the Adam optimizer with the number of epochs equal to 50. Note that there is no activation function or normalization applied in the SWINet, a larger learning rate will benefit the convergence of the Adam optimizer. To test the influence of the mini-batch strategy in seismic waveform inversion, we design a series of comparison experiments with different batch sizes and batch iteration steps per epoch. Firstly, we fix the total computation costs similar to the traditional full waveform inversion and conduct the velocity inversion using SWINet with batch sizes set as 2, 3, 4, 5, 6, 7, 10 and 20. The total computation cost is defined as the number of gradient computations, which is evaluated as $batchsize \times step \times Nepoch$. Considering that in every batch iteration step, the traditional FWI usually utilizes all of the observation data to update the velocity parameters, its process is coincident with the trial with one batch iteration step per epoch and batch size of 20. In other neural network based waveform inversion, for batch size equal to 3, 6, 7, we set batch step be 6, 3 and 2, respectively. In addition, we remove the restriction regrading computation cost and set batch steps equal to 2, 3, 4 and 5 to explore the inversion effect of SWINet with more or limited computation power.

In order to obtain a reliable outcome of SWINet inversion on Marmousi model, we repeat the aforementioned experiments for four times and their mean losses are summarized in Table 1. Here, the data loss is computed via the L2 norm of the difference between the forward modeling data and the real observation data and the model loss is defined based on the L2 norm of the difference between the inverted velocity model and the truth velocity model corresponding to the real observation data. According to the data loss, we mark the the traditional FWI loss, the overall lowest loss, the lowest loss with the fixed computation cost and the similar loss with lowest computation cost, respectively with color blue, red, green and magenta. The model losses are also marked accordingly

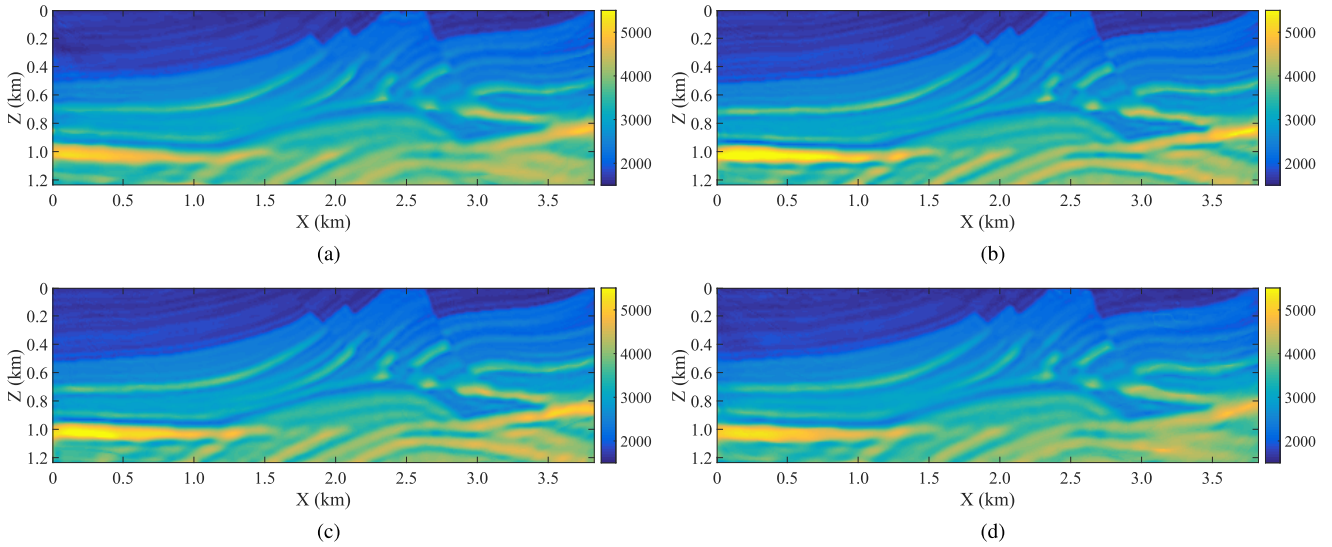



**FIGURE 6.** SWINet inversion results on Marmousi model corresponding to (a) traditional FWI (color blue in the table), (b) the overall lowest loss (color red in the table), (c) the lowest loss with the fixed computation cost (color green in the table) and (d) the similar loss with lowest computation cost (color magenta in the table).

**TABLE 1.** Statistics of the SWINet inversion loss with respect to different batch size (BS) and different batch iteration steps per epoch. According to the data loss, we mark the the traditional FWI loss, the overall lowest loss, the lowest loss with the fixed computation cost and the similar loss with lowest computation cost, respectively with color blue, red, green and magenta.

| Losses | Steps | BS=2 | BS=3 | BS=4 | BS=5 | BS=6 | BS=7 | BS=10 | BS=20 |
|---|---|---|---|---|---|---|---|---|---|
| Data loss ($\times 10^{-6}$) | full | 257.9005 | 165.3958 | 39.6733 | **9.6421** | 9.7075 | 32.8799 | 18.7311 | **347.0283** |
| | 2 | **555.0903** | 264.3514 | 95.4653 | 52.6833 | 39.7673 | 32.8801 | 18.7311 | 16.3929 |
| | 3 | 312.7122 | 112.6419 | 23.5527 | 11.7582 | 9.7075 | 9.3255 | 6.8733 | 6.9150 |
| | 4 | 221.6038 | 88.4615 | 32.7814 | **9.6421** | 6.4087 | 5.1219 | 3.8034 | 4.1269 |
| | 5 | 324.2509 | 88.6020 | 39.6688 | 8.0474 | 3.9161 | 3.1873 | **2.4755** | 2.7393 |
| Model loss ($\times 10^{4}$) | full | 9.6959 | 10.0557 | 9.7558 | **9.8072** | 10.0341 | 11.2089 | 10.7335 | **13.2180** |
| | 2 | **12.8754** | 12.3232 | 11.8651 | 10.9233 | 10.9862 | 11.2089 | 10.7335 | 11.0757 |
| | 3 | 11.5717 | 10.8416 | 10.4908 | 10.2157 | 10.0341 | 10.5329 | 9.8214 | 10.0376 |
| | 4 | 11.1437 | 10.4308 | 10.2332 | **9.8072** | 9.8076 | 9.6288 | 9.1222 | 9.2965 |
| | 5 | 11.0759 | 10.0932 | 9.7558 | 9.1771 | 8.9838 | 9.0077 | **8.5236** | 8.7563 |

and their corresponding inversion results randomly selected from the four repeated experiments are shown in Fig. 6.

### B. RESULT ANALYSIS

Here, we use the conventional FWI result as a benchmark to compare the SWINet inversion results in the view of data and model misfit as well as computation costs. By comparing the FWI result with the results of SWINet inversion based on other combinations of batch size and batch iteration step, we can see that the mini-batch strategy plays an essential role in obtaining a good inversion outcome with fast convergence and less computation power consumption.

As shown in Fig. 6, the best inversion result (Fig. 6b) comes from a batch size of ten and five batch iteration steps per epoch, which gives a total of 2500 times of gradient computations. Instead of 50 iterations, when conducting the conventional FWI for 250 iterations, which corresponds to a batch size of 20 and five batch iteration steps, one still cannot obtain an inversion result as good as Fig.6b. On the other hand, for a fixed computation power consumption (see the first row of data and model loss in Table 1), the best inversion result (Fig. 6c) comes from the case with batch size of five, which also demonstrates the advantage of using a mini-batch strategy. Moreover, it spends only 200 times of

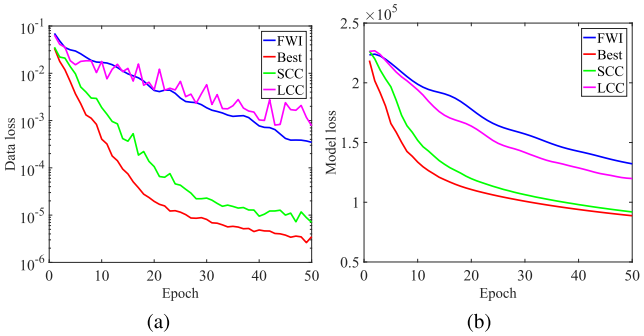



**FIGURE 7.** SWINet inversion losses corresponding to the four figures in Fig. 6, respectively. (a) data loss and (b) model loss. The "SCC" and "LCC" in legend correspond to the inversion results using a similar computation cost (SCC) to the conventional FWI and the lowest computation cost (LCC), respectively.

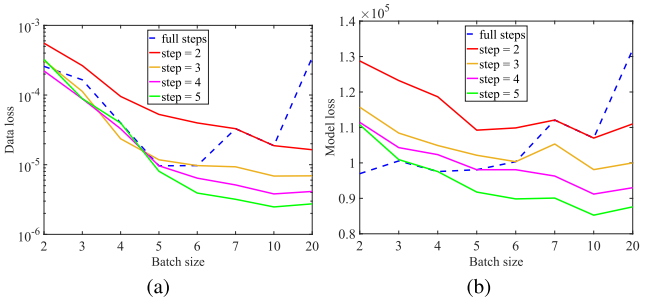



**FIGURE 8.** SWINet inversion losses with respect to different batch size. (a) data loss and (b) model loss.

gradient computations to get an inversion result (see Fig. 6d) whose inversion effect is similar to the conventional FWI. Noticing that the conventional FWI requires 1000 times gradient computations, which is another proof of the fast convergence when using mini-batch strategy in SWINet inversion. It is worth mentioning that the data loss curve for cases of setting batch size as two presents relatively severe oscillation in Fig. 7a. Although it is not ideal in the perspective of loss decaying, it implies a flexibility in updating the network parameters and demonstrates a great potential in refining the velocity model in the last stage of inversion. Thus, one may consider using a very small batch size to further invert the obtained inversion results in order to get an even better result.

Generally speaking, a smaller batch size leads to various gradient direction with smaller memory storage requirement. For a fixed batch size, larger batch iteration steps per epoch usually results in a better convergence, but requires larger computation cost. In our case, in order to determine an optimal batch size, we plot the losses in Fig. 8, where each curve represents each row in the loss Table 1. Close observation of these curves indicates that for a fixed batch step, both data loss and model loss have a decreasing trend. However, the downward trend seems to slow down at the batch size of 5. Moreover, for a fixed computation cost, i.e., when the batch size is defined as "full", the data loss curve started to rise after a turn point at batch size of 5. Thus, for a scenario with limited computation resources, especially a limited GPU memory, batch size set as 5 is potentially an efficient choice to receive a good inversion result and it is empirical to adopt four or five batch iteration steps in each epoch.

### C. COMPARISON WITH SeisInvNet

In order to demonstrate the potential of the physics-based deep learning scheme, we conduct a comparison experiment to show the inversion effect of the SeisInvNet and the proposed SWINet. A dataset containing 18000 pairs of velocity models and the corresponding observation data is numerically generated. 15000 of them will be used to train the SeisInvNet and we use another 1500 pairs of data for validation and 1500 for test. To make the comparison in a more realistic situation, we design the velocity models to have some complex geology structures like folding layers and faults, and 20 seismic sources and 32 receivers are evenly placed on the top layer to generate the corresponding observation data. By following the setup parameters shown in the reference [32], we train the SeisInvNet for 200 epochs and its inversion effect on the test dataset is compared with the SWINet, which is trained on the observation data of each test velocity model based on the same network setup as the color green indicates in Table 1.

An example of the comparison experiment result is shown in Fig. 9. By comparing Figs. 9b and 9c, one can see that SWINet seems to have a better inversion effect on the shallow layers and the faulting structure while the inverted velocity value within each layers seems more consistent in the SeisInvNet result than the SWINet result. However, we believe this drawback of SWINet can be addressed via incorporating a smooth regularization constraint in the network architecture. Generally speaking, both methods have certain advantages. Unlike the SeisInvNet, SWINet does not rely on the availability of the true velocity models as training labels. On the other hand, the training process of SWINet requires a good estimated velocity model for network parameter initialization, while SeisInvNet can be initialized randomly. As will be discussed later, a combination of both SWINet and SeisInvNet could be studied to weaken the requirement of labelled dataset (for SeisInvNet) and good network parameter initialization (for SWINet) in the future.

## V. DISCUSSION

The goal of this paper is to reformulate the traditional FWI from a different perspective, e.g., in the format of training a neural network, where the network parameters become the subsurface velocity distribution. Thus in this work, we demonstrate a physics-based neural network way to perform seismic waveform inversion by proposing the SWINet. After theoretically proving the equivalence of two gradient calculation methods, i.e., the adjoint state method in traditional seismic waveform inversion and the automatic differentiation widely used in deep learning, in a discrete matrix form, we choose the latter due to its guarantee of adjointness and the convenient implementation in Pytorch. Numerical experiments with the Adam optimizer and mini-batch strategy indicate that popular ideas from the deep learning

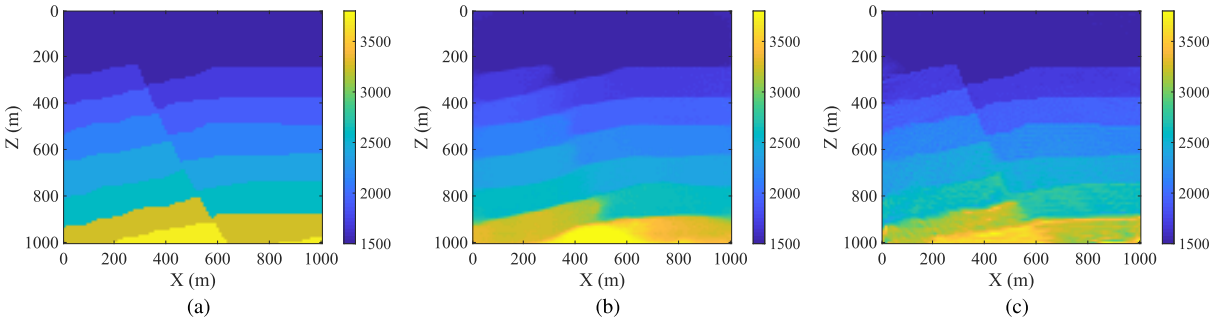



**FIGURE 9.** A comparison experiment example. (a) True velocity model, (b) inversion result of SeisInvNet and (c) inversion result of SWINet.

community can also be beneficial to seismic full waveform inversion and result in an improvement in the perspective of convergence speed and computation cost.

Moreover, the numerical experiment results could be further analyzed in terms of the possibility to obtain an uncertainty estimation [45] in the proposed framework. In a nutshell, the uncertainty estimation based on the Adam optimizer is highly worth investigating, but we haven't reached that step. From the perspective of a general inverse problem, the uncertainty estimation can be easily obtained by a bootstrapping method, i.e., we randomly select a fraction of the whole input data for the inversion, and then calculate the standard derivation of all the inverted result we have tested to be the uncertainty of the inversion process.

Compared with the traditional FWI, the innovation of this work lies in the sense of the implementation form and optimization strategy. This not only grants geophysicist an easy access to latest advancements arised in all sorts of deep learning tasks, but also have a potential in solving the deep learning problems with inspiring seismic approaches. For example, when facing a high memory demand problem like seismic inversion, where calculating the gradient using automatic differentiation method requires the storage of the wavefield from both forward modeling and backward propagation. One possible way to greatly reduce the memory cost is to incorporate a wavefield reconstruction method [46]–[48] into the automatic differentiation, which can rebuild the forward modeling wavefield and calculate the gradient while backward propagation. Therefore, this work can be seen as a stepping stone to deepen the cooperation among the researchers from both geophysics and deep learning.

In addition, comparing the proposed SWINet with the classic multi-layer perceptron (MLP) neural network, one can conclude with two major differences in the network architecture design: in SWINet, (a) there is no need for activation functions like the sigmoid function in accurately simulating the wave equation, which plays an essential role in seismic waveform inversion problem and (b) network parameters (i.e., velocity model) are the same for all FM cells. That is, there seems to be a very strong constraint on network parameters. In addition, instead of the single outcome of the last layers, the loss function in seismic inversion problem involves the outcome of every layers. Thus, differently from the classic deep learning network, where random initialization of network parameters works well [49], SWINet inversion depends on a good initial model. How to modify the network architecture and tackle this issue may potentially lead to a major breakthrough in seismic full waveform inversion.

Another potential of SWINet is to incorporate the FM cells into the current purely data-driven seismic inversion network such as the InversionNet [31] and the SeisInvNet [32]. In this way, the physics nature of wave propagation can provide guidance for the deep learning network to capture the true pattern between the training data pairs, thereby improve the generalization ability of seismic inversion network. Furthermore, since the forward modeling data can be generated within the network, how to construct an unsupervised seismic inversion network and ease the requirement of true velocity model as a label would be a hot topic worth further investigation.

## VI. CONCLUSION

In this work, we constructed a seismic forward modeling neural network and proposed the SWINet for seismic waveform inversion. With the velocity model acting as learnable parameters, a set of single-shot observation data will be used to train the SWINet. Numerical examples indicate that the novel way to perform FWI provides an easy access to the useful tools in deep learning society and the incorporation of deep learning approaches, such as the Adam optimizer and mini-batch strategy, can lead to an improvement on seismic inversion like fast convergence and lower computation costs. As illustrated in the discussion, there are some topics related to this line of work that may be worth investigating.

## APPENDIX A
## GRADIENT DERIVATION VIA MATRIX CALCULUS

In this appendix, we will derive the velocity gradient in a matrix calculus way, whose outcome is identical to the result of adjoint state method that is massively used in traditional seismic full waveform inversion. Following the notations in the background acoustic wave equation theory, we have the wavefield solver in the discretized form as

$$\mathbf{p} = \mathbf{B}^{-1}\mathbf{s}, \tag{10}$$

and the least-squares loss function is

$$\mathcal{L} = \frac{1}{2}\left(\mathbf{D}_r\mathbf{p} - \mathbf{D}_r\bar{\mathbf{p}}\right)^T\left(\mathbf{D}_r\mathbf{p} - \mathbf{D}_r\bar{\mathbf{p}}\right). \tag{11}$$

Here, $\mathbf{p}$ and $\bar{\mathbf{p}}$ denote the forward modeling wavefield and the real wavefield, respectively. The goal is to calculate the gradient of the loss function with respect to the velocity $\mathbf{v}$. Firstly, we can easily compute

$$\frac{d\mathcal{L}(\mathbf{p})}{d\mathbf{p}} = {\mathbf{D}_r}^T\mathbf{D}_r(\mathbf{p} - \bar{\mathbf{p}}) := \boldsymbol{\epsilon}, \tag{12}$$

which is in fact the data residual distributed at all receiver points. Then, based on the chain rule and considering that $\mathbf{B}$ is a function of the velocity model $\mathbf{v}$, for each $v_i$ we have

$$\begin{aligned}\frac{d\mathcal{L}}{dv_i} &= \frac{d\mathbf{p}}{dv_i}\frac{d\mathcal{L}}{d\mathbf{p}} = -\left(\mathbf{B}^{-1}\frac{d\mathbf{B}}{dv_i}\mathbf{B}^{-1}\mathbf{s}\right)^T\boldsymbol{\epsilon} \\ &= -\left(\frac{d\mathbf{B}}{dv_i}\mathbf{B}^{-1}\mathbf{s}\right)^T\mathbf{B}^{-T}\boldsymbol{\epsilon}.\end{aligned} \tag{13}$$

Here,

$$\frac{d\mathbf{B}}{dv_i} = -\begin{bmatrix} \mathbf{0} & & & & \\ \mathbf{G}_i' & \mathbf{0} & & & \\ & \mathbf{G}_i' & \mathbf{0} & & \\ & & \ddots & \ddots & \\ & & & \mathbf{G}_i' & \mathbf{0} \end{bmatrix}, \tag{14}$$

and

$$\mathbf{G}_i' = \frac{d\mathbf{G}}{dv_i} = 2\alpha v_i\begin{bmatrix} \ddots & \ddots & \ddots & & & & \\ & 0 & 0 & 0 & & & \\ & & 1 & -2 & 1 & & \\ & & & 0 & 0 & 0 & \\ & & & & \ddots & \ddots & \ddots \end{bmatrix}. \tag{15}$$

Notice that, the subdiagonal matrix in (14) corresponds to the one-time-step delay in forward modeling wavefield $\mathbf{B}^{-1}\mathbf{s}$ when it cross-correlates with the backward propagated wavefield $\mathbf{B}^{-T}\boldsymbol{\epsilon}$. The matrix $\mathbf{G}'$ relates to the $i$th row of the Laplacian matrix, which can be further proved to be coincident with the second-order time derivative of the forward model wavefield in the conventional Born approximation theory.

## APPENDIX B
## GRADIENT CALCULATION VIA AUTO-DIFFERENTIATION

Here, we will calculate the gradient of network parameters following the steps of automatic differentiation. In this case, the least-squares loss function (11) can be rewritten as

$$\mathcal{L} = \frac{1}{2}\sum_{n=1}^{nt}(\mathbf{D}_r^n\mathbf{p}^n - \mathbf{D}_r^n\bar{\mathbf{p}}^n)^2. \tag{16}$$

Here, $\mathbf{p}^n$ and $\bar{\mathbf{p}}^n$ are the forward modeling wavefield and the real wavefield at the $n$th time step. According to the back-propagation method, we need to firstly define a series of middle variables according to the forward modeling equation (3) as

$$\begin{aligned}\boldsymbol{\delta}^{nt} &= \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt}} = {\mathbf{D}^{nt}}^T(\mathbf{D}^{nt}\mathbf{p}^{nt} - \mathbf{D}^{nt}\bar{\mathbf{p}}^{nt}) = \boldsymbol{\epsilon}^{nt}, \\ \boldsymbol{\delta}^{nt-1} &= \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt}}\frac{\partial\mathbf{p}^{nt}}{\partial\mathbf{p}^{nt-1}} + \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt-1}} \\ &= \mathbf{G}^T\boldsymbol{\delta}^{nt} + \boldsymbol{\epsilon}^{nt-1}, \\ \boldsymbol{\delta}^{nt-2} &= \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt}}\frac{\partial\mathbf{p}^{nt}}{\partial\mathbf{p}^{nt-2}} + \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt-1}}\frac{\partial\mathbf{p}^{nt-1}}{\partial\mathbf{p}^{nt-2}} + \frac{\partial\mathcal{L}}{\partial\mathbf{p}^{nt-2}} \\ &= \mathbf{G}^T\boldsymbol{\delta}^{nt-1} - \boldsymbol{\delta}^{nt} + \boldsymbol{\epsilon}^{nt-2}, \\ &\cdots\cdots \\ \boldsymbol{\delta}^{n} &= \mathbf{G}^T\boldsymbol{\delta}^{n+1} - \boldsymbol{\delta}^{n+2} + \boldsymbol{\epsilon}^{n}.\end{aligned} \tag{17}$$

Similarly to Appendix A, we obtain the data residual at the $n$th time step by $\boldsymbol{\epsilon}^n = \frac{\partial\mathcal{L}}{\partial\mathbf{p}^n}$. $\boldsymbol{\delta}^n$ corresponds to the data residual propagation backward in time based on the future wavefield and it is in fact the adjoint wavefield $\mathbf{B}^{-T}\boldsymbol{\epsilon}$ at the $n$th time step.

Then based on the chain rule, the gradient of loss function with respect to the $i$th network parameter $v_i$ can be calculated as

$$\begin{aligned}\frac{d\mathcal{L}}{dv_i} &= \sum_{n=0}^{nt}\left(\frac{d\mathcal{L}}{d\mathbf{p}^n}\frac{d\mathbf{p}^n}{d\mathbf{G}}\frac{d\mathbf{G}}{dv_i}\right) \\ &= \sum_{n=1}^{nt}\left((\mathbf{G}_i'\mathbf{p}^{n-1})^T\boldsymbol{\delta}^n\right).\end{aligned} \tag{18}$$

Here $\mathbf{G}_i'$ is the same as the (15) in the Appendix A, $\mathbf{p}^{n-1}$ denotes the forwarding modeling wavefield $\mathbf{B}^{-1}\mathbf{s}$ at the $(n-1)$th time step, which agrees with the one-time-step delay caused by the subdiagonal matrix (14). In other words, forward-propagating through the SWINet gives $\mathbf{B}^{-1}\mathbf{s}$ and backward-propagating the data error gives $\mathbf{B}^{-T}\boldsymbol{\epsilon}$, then cross-correlating both wavefields according (18) is identical to its theorical value shown in (13). Therefore, adopting the automatic differentiation steps to calculate the gradient leads to the same results as in the traditional seismic waveform inversion using the adjoint state method.

## REFERENCES

[1] C. Bunks, F. M. Saleck, S. Zaleski, and G. Chavent, "Multiscale seismic waveform inversion," *Geophysics*, vol. 60, no. 5, pp. 1457–1473, Sep. 1995.

[2] R. G. Pratt, "Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model," *Geophysics*, vol. 64, no. 3, pp. 888–901, May 1999.

[3] Z. Pan, J. Wu, Z. Gao, and J. Gao, "Adaptive differential evolution by adjusting subcomponent crossover rate for high-dimensional waveform inversion," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 6, pp. 1327–1331, Jun. 2015.

[4] S. Qu, E. Verschuur, and Y. Chen, "Fwi/jmi with an automatic directional total variation constraint," *Geophysics*, vol. 84, no. 2, pp. 175–183, 2019.

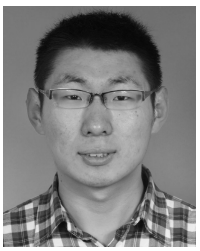
[5] L. Mosser, O. Dubrule, and M. J. Blunt, "Stochastic seismic waveform inversion using generative adversarial networks as a geological prior," *Math. Geosci.*, vol. 52, no. 1, pp. 53–79, Jan. 2020.

[6] P. Lailly, "The seismic inverse problem as a sequence of before stack migrations," in *Conference on Inverse Scattering: Theory and Application*. Philadelphia, PA, USA: SIAM, 1983, pp. 206–220.

[7] A. Tarantola, "Inversion of seismic reflection data in the acoustic approximation," *Geophysics*, vol. 49, no. 8, pp. 1259–1266, Aug. 1984.

[8] R.-E. Plessix, "A review of the adjoint-state method for computing the gradient of a functional with geophysical applications," *Geophys. J. Int.*, vol. 167, no. 2, pp. 495–503, Nov. 2006.

[9] Q. Zhang, W. Mao, H. Zhou, H. Zhang, and Y. Chen, "Hybrid-domain simultaneous-source full waveform inversion without crosstalk noise," *Geophys. J. Int.*, vol. 215, no. 3, pp. 1659–1681, Dec. 2018.
[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.
[11] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5574–5584.
[12] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–13, 2018.
[13] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 160–167.
[14] W. Yin, K. Kann, M. Yu, and H. Schätze, "Comparative study of CNN and RNN for natural language processing," 2017, *arXiv:1702.01923*. [Online]. Available: http://arxiv.org/abs/1702.01923
[15] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.
[16] L. Huang, X. Dong, and T. E. Clee, "A scalable deep learning platform for identifying geologic features from seismic attributes," *Lead. Edge*, vol. 36, no. 3, pp. 249–256, Mar. 2017.
[17] G. Zhang, Z. Wang, and Y. Chen, "Deep learning for seismic lithology prediction," *Geophys. J. Int.*, pp. 1368–1387, Aug. 2018.
[18] S. Lameri, F. Lombardi, P. Bestagini, M. Lualdi, and S. Tubaro, "Landmine detection from GPR data using convolutional neural networks," in *Proc. 25th Eur. Signal Process. Conf. (EUSIPCO)*, Aug. 2017, pp. 508–512.
[19] B. Liu, Q. Guo, S. Li, B. Liu, Y. Ren, Y. Pang, X. Guo, L. Liu, and P. Jiang, "Deep learning inversion of electrical resistivity data," *IEEE Trans. Geosci. Remote Sens.*, early access, Feb. 11, 2020, doi: 10.1109/TGRS.2020.2969040.
[20] F. Wang, S. Chen, and Y. Liu, "Deep learning for gravity and magnetic data interpolation," in *Proc. SEG Tech. Program Expanded Abstr.*, Aug. 2019, pp. 2533–2537.
[21] H. Li, W. Yang, and X. Yong, "Deep learning for ground-roll noise attenuation," in *Proc. SEG Tech. Program Expanded Abstr.*, Aug. 2018, pp. 1981–1985.
[22] D. Liu, W. Wang, W. Chen, X. Wang, Y. Zhou, and Z. Shi, "Random-noise suppression in seismic data: What can deep learning do?" in *Proc. SEG Tech. Program Expanded Abstr.*, Aug. 2018, pp. 2016–2020.
[23] S. Yu, J. Ma, and W. Wang, "Deep learning for denoising," *Geophysics*, vol. 84, no. 6, pp. V333–V350, Nov. 2019.
[24] O. Saad and Y. Chen, "Deep denoising autoencoder for seismic random noise attenuation," *Geophysics*, vol. 85, no. 4, doi: 10.1190/geo2019–0468.1.2020.
[25] Y. Chen, "Automatic microseismic event picking via unsupervised machine learning," *Geophys. J. Int.*, vol. 212, no. 1, pp. 88–102, Jan. 2018.
[26] H. Wu, B. Zhang, F. Li, and N. Liu, "Semiautomatic first-arrival picking of microseismic events by using the pixel-wise convolutional image segmentation method," *Geophysics*, vol. 84, no. 3, pp. V143–V155, May 2019.
[27] W. Zhu and G. C. Beroza, "PhaseNet: A Deep-Neural-Network-Based seismic arrival time picking method," *Geophys. J. Int.*, pp. 261–273, Oct. 2018.
[28] A. U. Waldeland, A. C. Jensen, L.-J. Gelius, and A. H. S. Solberg, "Convolutional neural networks for automated seismic interpretation," *Lead. Edge*, vol. 37, no. 7, pp. 529–537, Jul. 2018.
[29] H. Di, D. Gao, and G. AlRegib, "Developing a seismic texture analysis neural network for machine-aided seismic pattern recognition and classification," *Geophys. J. Int.*, vol. 218, no. 2, pp. 1262–1275, Aug. 2019.
[30] M. Araya-Polo, J. Jennings, A. Adler, and T. Dahlke, "Deep-learning tomography," *Lead. Edge*, vol. 37, no. 1, pp. 58–66, 2018.
[31] Y. Wu, Y. Lin, and Z. Zhou, "Inversionnet: Accurate and efficient seismic waveform inversion with convolutional neural networks," in *Proc. SEG Tech. Program Expanded Abstr.*, 2018, pp. 2096–2100.
[32] S. Li, B. Liu, Y. Ren, Y. Chen, S. Yang, Y. Wang, and P. Jiang, "Deep-learning inversion of seismic data," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 3, pp. 2135–2149, Oct. 2020.
[33] R. Biswas, M. K. Sen, V. Das, and T. Mukerji, "Pre-stack inversion using a physics-guided convolutional neural network," in *Proc. SEG Tech. Program Expanded Abstr.*, Aug. 2019, pp. 4967–4971.
[34] J. Sun, Z. Niu, K. A. Innanen, J. Li, and D. O. Trad, "A theory-guided deep-learning formulation and optimization of seismic waveform inversion," *Geophysics*, vol. 85, no. 2, pp. R87–R99, Mar. 2020.
[35] A. Richardson, "Seismic full-waveform inversion using deep learning tools and techniques," 2018, *arXiv:1801.07232*. [Online]. Available: http://arxiv.org/abs/1801.07232
[36] M. Abadi, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: http://arxiv.org/abs/1603.04467
[37] A. Paszke, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8024–8035.
[38] C. Cerjan, D. Kosloff, R. Kosloff, and M. Reshef, "A nonreflecting boundary condition for discrete acoustic and elastic wave equations," *Geophysics*, vol. 50, no. 4, pp. 705–708, Apr. 1985.
[39] C. Shin, "Sponge boundary condition for frequency–domain modeling," *Geophysics*, vol. 60, no. 6, pp. 1870–1874, Nov. 1995.
[40] N. Shah, M. Warner, T. Nangoo, A. Umpleby, I. Stekl, J. Morgan, and L. Guasch, "Quality assured full-waveform inversion: Ensuring starting model adequacy," in *Proc. SEG Tech. Program Expanded Abstr.*, Sep. 2012, pp. 1–5.
[41] K. Jiao, D. Sun, X. Cheng, and D. Vigh, "Adjustive full waveform inversion," in *Proc. SEG Tech. Program Expanded Abstr.*, Aug. 2015, pp. 1091–1095.
[42] Y. Chen, H. Chen, K. Xiang, and X. Chen, "Geological structure guided well log interpolation for high-fidelity full waveform inversion," *Geophys. J. Int.*, vol. 207, no. 2, pp. 1313–1331, Nov. 2016.
[43] J. F. Claerbout, *Earth Soundings Analysis: Processing Versus Inversion*, vol. 6. London, U.K.: Blackwell Scientific, 1992.
[44] M. Louboutin, P. Witte, M. Lange, N. Kukreja, F. Luporini, G. Gorman, and F. J. Herrmann, "Full-waveform inversion, part 2: Adjoint modeling," *Lead. Edge*, vol. 37, no. 1, pp. 69–72, Jan. 2018.
[45] Z.-D. Zhang and T. Alkhalifah, "Regularized elastic full-waveform inversion using deep learning," *Geophysics*, vol. 84, no. 5, pp. R741–R751, Sep. 2019.
[46] H. Liu, R. Ding, L. Liu, and H. Liu, "Wavefield reconstruction methods for reverse time migration," *J. Geophys. Eng.*, vol. 10, no. 1, Feb. 2013, Art. no. 015004.
[47] E. B. Raknes and W. Weibull, "Efficient 3D elastic full-waveform inversion using wavefield reconstruction methods," *Geophysics*, vol. 81, no. 2, pp. 45–55, Mar. 2016.
[48] M. Vasmel and J. O. A. Robertsson, "Exact wavefield reconstruction on finite-difference grids with minimal memory requirements," *Geophysics*, vol. 81, no. 6, pp. 303–309, Nov. 2016.
[49] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.



**YUXIAO REN** received the bachelor's degree in mathematics from Shandong University, China, in 2014, and the master's degree in mathematics from Loughborough University, U.K., in 2015. He is currently pursuing the Ph.D. degree in civil engineering with Shandong University. He is also a Visiting Scholar with the Georgia Institute of Technology under the supervision of Prof. Felix Herrmann. His research interests include seismic modeling and imaging, full-waveform inversion, and deep-learning-based geophysical inversion.

**XINJI XU** was born in Jining, Shandong, China, in 1990. He received the B.S. degree in city underground engineering and the Ph.D. degree from Shandong University, Jinan, Shandong, in 2012 and 2017, respectively. Since 2017, he has been working as an Assistant Research Fellow with the Geotechnical and Structural Engineering Research Center, Shandong University. His research interests include advance geological forecast of tunnel and underground engineering, seismic data processing, and seismic imaging and inversion.

**SENLIN YANG** received the bachelor's degree in engineering from Shandong University, China, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include deep-learning-based geophysical data processing and inversion.

**LICHAO NIE** received the Ph.D. degree from the Geotechnical and Structural Engineering Research Center, Shandong University, China, in 2014. He is currently an Associate Professor with the School of Civil Engineering, Shandong University. He is mainly engaged in geophysical forward and inversion theory and method, geological forward-prospecting method and technology in tunnels, and tunnel boring machine with forward-prospecting system as well as its engineering application, and so on.

**YANGKANG CHEN** received the B.S. degree in exploration geophysics from the China University of Petroleum, Beijing, in 2012, and the Ph.D. degree in geophysics from The University of Texas at Austin, in 2015, under the supervision of Prof. Sergey Fomel. He joined with the Oak Ridge National Laboratory, as a Distinguished Postdoctoral Research Associate, where he conducted research on global adjoint tomography. He is currently an Assistant Professor with Zhejiang University. He has published more than 100 internationally renowned journal articles and more than 80 international conference papers. His research interests include seismic signal analysis, seismic modeling and inversion, simultaneous-source data deblending and imaging, global adjoint tomography, and machine learning.