Soil Dynamics and Earthquake Engineering 162 (2022) 107510

Contents lists available at ScienceDirect

Soil Dynamics and Earthquake Engineering

journal homepage: www.elsevier.com/locate/soildyn

Deep and Convolutional Neural Networks for identifying vertically-propagating incoming seismic wave motion into a heterogeneous, damped soil column

Shashwat Maharjan [a], Bruno Guidio [a], Arash Fathi [b], Chanseok Jeong [a,c,*]

[a] School of Engineering and Technology, Central Michigan University, Mount Pleasant, 48859, MI, USA
[b] ExxonMobil Technology and Engineering, Annandale, 08801, NJ, USA
[c] Earth and Ecosystem Science Program, Central Michigan University, Mount Pleasant, 48859, MI, USA

ARTICLE INFO

Keywords:
Machine learning
Deep neural network
Convolutional neural network
Incident seismic-wave motion inversion
Inverse-source problem
Seismic wave propagation

ABSTRACT

Identification of the incoming seismic wave motion at a geotechnical site plays an integral role in the seismic analysis and design of the critical components of the civil infrastructure. Current practice relies on using one-dimensional models for the soil medium at the site, and a process called deconvolution, which identifies seismic input due to measurements made at the ground surface. Using multi-dimensional models for the underlying soil medium becomes important when a site exhibits considerable heterogeneity or change in topography. These situations necessitate using a gradient-based optimization method for inverting the multi-dimensional incoming seismic-wave motion. However, such methods are computationally expensive and time consuming.

We explore the effectiveness and robustness of a data-informed framework for the inverse-source problem, due to its potential in reducing the computational cost, compared to a gradient-based approach. We design deep and convolutional neural network architectures to predict the incoming wave motion based on measurements made at the ground surface. We demonstrate their effectiveness and robustness on blind test examples, where measured data are contaminated with noise, and when the incident signals in the training data set may or may not resemble a realistic seismic signal. Lastly, the presented artificial neural networks are shown to be effective in predicting incoming wave motion when the subsurface material properties lack accuracy, or are uncertain, which is likely the case in realistic situations. While only one-dimensional problems are considered here, generalization of our data-informed approach to handle multi-dimensional problems appears to be straightforward. Overall, our data-informed approach seems to be robust, fast, and promising for identifying the incoming seismic wave motion.

1. Introduction

Seismic design of critical components of the civil infrastructure, such as dams, bridges, hospitals, and power plants, relies on past seismic events [1]. Previously recorded seismic events, however, may not be available for certain geographical locations, or geological profiles. In these situations, seismic events corresponding to other locations or profiles are processed, in order to make them suitable for a new site. This process is known as site response analysis. Part of this process involves the identification of seismic wave motion on a bedrock, based on seismic wave motion measurements on the ground surface, in an area where the soil profile is known. The identified seismic wave motion on the bedrock, also known as input motion, can then be used at a new site with a possibly different soil profile. The input motion can be propagated through the site's local soil profile in a computer model, where the model may also account for local topography and the civil (infra)structure, in order to accurately compute the seismic response of the structure, which then may guide safe design options.

Profiling the site under investigation is often needed prior to conducting the site response analysis. This typically involves identifying shear and, sometimes, compressional wave velocities. We refer to Roesset et al. [2], Luke and Stokoe II [3], and Brown et al. [4], for using non-invasive techniques, such as the spectral analysis of surface waves (SASW) method, to Park et al. [5], and Rahimi et al. [6], for using the multi-channel analysis of surface waves (MASW) method, to Cox and Teague [7] for shear wave velocity profiling of sites via using surface wave dispersion data with limited *a priori* information and a robust parameterization scheme, to Teague et al. [8] for calculation of the transfer functions for a ground response by using the shear

* Corresponding author at: School of Engineering and Technology, Central Michigan University, Mount Pleasant, 48859, MI, USA.
E-mail address: jeong1c@cmich.edu (C. Jeong).

https://doi.org/10.1016/j.soildyn.2022.107510
Received 5 January 2022; Received in revised form 29 July 2022; Accepted 20 August 2022
Available online 2 September 2022

wave velocity profiles from surface wave testing at a site, to Kallivokas et al. [9], and Fathi et al. [10,11] for the full-waveform inversion (FWI) of the three-dimensional spatial distribution of the soil profile at a site, and to Askan et al. [12,13] for anelastic seismic inversion methods to characterize wave speeds and attenuation coefficients of basins from earthquake measurement data. We also refer to Poursartip et al. [14], Poursartip and Kallivokas [15], Asimaki and Mohammadi [16], and Ayoubi et al. [17] for accounting for topography effects, which form the remaining integral parts of this process.

In this paper, we focus on an element of the site response analysis workflow: identifying the input seismic wave motion on the bedrock, based on measurements at the ground surface, under a known soil profile. Current practice for identifying the input seismic wave motion on the bedrock, based on measurements on the ground surface, typically relies on using one-dimensional layered-soil models, along with a procedure known as deconvolution [18–21]. The deconvolution works as follows: (a) the measured waveform on the ground surface is transformed into the frequency domain; (b) the transfer function of the one-dimensional soil profile is computed; and (c) the measured waveform in the frequency domain is multiplied by the inverse of the transfer function, and this product is then transformed back into the time domain. This procedure yields the time-domain waveform of the incident wave on the bedrock.

The deconvolution procedure, however, has only been applied to one-dimensional domains that support vertically-propagating waves. To address shortcomings of the deconvolution process and identify incoherent incident waves (e.g., inclined waves) shaking a multi-dimensional domain, incoming seismic-motion estimation procedure is needed and was recently developed by Guidio and Jeong [22], Guidio et al. [23]. They used the apparatus of partial differential equation (PDE)-constrained optimization, using the full-waveforms of the recorded motions at receiver locations, to perform source inversion. Guidio et al. [23] estimated both the temporal signature and spatial distribution of the seismic input at the absorbing boundaries of a truncated computational model for two-dimensional anti-plane SH waves. Moreover, the work by Guidio and Jeong [22] estimated source body forces within a domain reduction method (DRM) layer [24], which then can be used to generate waveforms that match measured observations at receiver locations on the ground surface, for two-dimensional truncated domains. As Guidio and Jeong [25] argued, the relationship between the parameterized force vector and the wave response is linear, which results in a quadratic and convex objective functional, when an L_2-norm is used for the associated inverse-source problem. In addition, Wang and Alkhalifah [26] and Song and Alkhalifah [27] reported FWI studies to jointly invert for both the unknown source, as well as the unknown velocity profile, which is even more challenging due to the increased non-uniqueness of the joint inversion process than when only source-inversion is attempted.

Despite the robust performance of the PDE-constrained optimization for estimating seismic inputs in two-dimensional settings, serving as an alternative to the deconvolution, the PDE-constrained optimization procedure is computationally expensive and takes a significant amount of time for completion. Therefore, extensions of this procedure to the more realistic three-dimensional case will require significantly more computing power and time. In this work, we explore the effectiveness and robustness of using artificial neural networks (ANNs) for seismic-input inversion, and whether it is able to reduce the time to solution.

There is extensive literature on using ANNs for tackling inverse problems. Röth and Tarantola [28] applied a neural network method to reconstruct a one-dimensional wave speed from seismic data. They showed that neural networks are able to tackle velocity-model inverse problems. Vantassel et al. [29] investigated the use of convolutional neural networks (CNNs) to produce a two-dimensional material property profile by using surface measurements due to a surficial point source. The resulted material profile was then used as the starting point of the subsequent FWI process. Iturrarán-Viveros et al. [30] also utilized ANNs to estimate an initial velocity model for two-dimensional domains. Then, the approximated profile was used as initial input for FWI. Kazei et al. [31] used an ensemble of CNNs to invert seismic data, directly for high-resolution vertical velocity profiles, and showed its promise when applied to field data. Wu and Lin [32] explored a CNN model with an encoder–decoder, named InversionNet, to solve the seismic FWI directly. Later, they used a conditional random field to refine the reconstructed velocity profile. Ren et al. [33] proposed a technique for tackling FWI based on a physics-informed neural network (PINN). They also constructed a forward modeling neural network to generate training data for their FWI network. Song and Alkhalifah [34] also applied PINNs to FWI, when the physics is described by the Helmholtz equation. Moreover, Yang and Ma [35] utilized supervised deep fully convolutional neural networks (FCN)[1] to estimate the velocity profiles for the seismic inversion problem. Haile et al. [36] implemented two deep neural networks (DNNs) in order to exploit the acoustic emission signals to approximate the geometry and position of cracks in structures. The paper uses long short-term memory (LSTM) – a recurrent neural network (RNN) architecture – and two-dimensional CNNs, to carry out a classification task for detecting the presence, or absence, of damage in structures. Another paper by Li et al. [37] discussed a generative adversarial network (GAN) to develop an early earthquake warning system. The GAN is trained on 300,000 first-arrival earthquake P-waves, from southern California and Japan, and is trained to classify both real and false earthquake alerts in order to increase the robustness of the procedure. To predict the locations of microseismic events from waveform data, Wang and Alkhalifah [38], and Wang et al. [39], used a CNN network and demonstrated the success of their approach using field data. Furthermore, Zheng et al. [40] explored using synthetic data to train a CNN to predict features such as fault profiles, or P-wave velocities, in a domain of interest. The paper highlights the computational efficiency of using an ANN for seismic problems. Indeed, a trained ANN can almost instantaneously (e.g., within a second) produce the inversion outcome.

Inspired by the robust performance of ANNs for tackling inverse problems in seismology and the absence of using neural networks for inverse-source problems in the literature, we investigate the effectiveness and computational cost of using ANNs for seismic input inversion in a one-dimensional soil column. We discuss how training data for such problems may be generated. We then investigate how deep and convolutional neural networks perform for the incident seismic-wave motion inversion problem. Lastly, we explore how subsurface uncertainty, e.g., in wave speed, impacts the performance of our proposed data-informed framework.

2. Problem description

Fig. 1 shows the problem setting where the presented modeling can be utilized. We are interested in identifying an unknown incoming seismic wave through a site of interest, where the site is modeled with a one-dimensional soil column. In rare situations, this incoming wave can be directly measured from a neighboring outcrop rock, when such an outcrop rock exists. At the outcrop rock, the measured ground motion is the same as the incoming wave motion at the site of interest, but with a twice larger amplitude. However, several assumptions should be satisfied for the measured ground motion at a neighboring outcrop rock to be utilized as the incoming wave motion at the site of interest. First, material properties of the bedrock should be horizontally uniform, rendering the bedrock at site of interest, and at the outcrop rock, identical. Second, the incident waves should propagate vertically, in the same waveform and phase characteristics, at both the site of interest and

[1] Compared to CNN, FCN is composed of convolutional layers without any fully-connected layer.

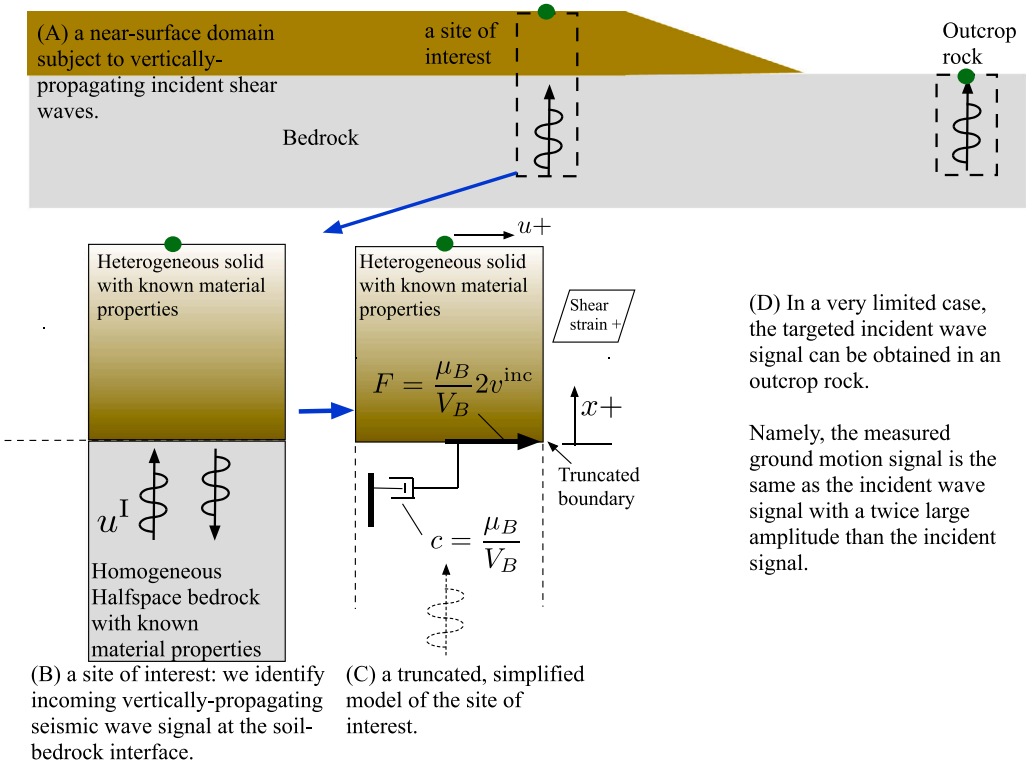


Fig. 1. A semi-infinite, one-dimensional soil column, subjected to a vertically propagating incident shear wave. The homogeneous halfspace rock can be replaced by a dashpot underneath the soil column, where a dynamic shear force substitutes the incident wave.

the outcrop rock. Since such a case is rare, methods[2] that estimate the incoming seismic wave motion based on measurements at the ground surface are widely used. This paper presents an alternative method for estimating the incoming seismic wave motion, based on ground-surface measurements, using ANNs.

Wave motion within a one-dimensional, semi-infinite soil column (Fig. 1) is described by the following partial differential equations [41]:

$$\frac{\partial}{\partial x}\left[\mu(x)\frac{\partial u(x,t)}{\partial x}\right] = \rho(x)\frac{\partial^2 u(x,t)}{\partial t^2}, \qquad 0<x<L, \quad 0<t<T, \tag{1a}$$

$$\mu(x)\frac{\partial u(x,t)}{\partial x} = \frac{\mu_B}{V_B}\left[\frac{\partial u(x,t)}{\partial t} - 2v^{\mathrm{inc}}(t)\right], \qquad x=0, \tag{1b}$$

$$\mu(x)\frac{\partial u(x,t)}{\partial x} = 0, \qquad x=L, \tag{1c}$$

where $v^{\mathrm{inc}}(t)$ is the particle velocity of the unknown incoming seismic wave that derives the wave motion in Eq. (1), which we aim to identify; $\mu(x)$ and $\rho(x)$ are shear modulus and mass density (material properties), respectively; $u(x,t)$ represents the particle displacement in the medium; and μ_B and V_B denote shear modulus and shear wave velocity of the bedrock, respectively. We use Rayleigh damping to model attenuation in the medium, described in discrete form by Eq. (4). The problem enjoys at-rest initial conditions: $u(x,0) = 0$, and $\frac{\partial u(x,0)}{\partial t} = 0$. The shear wave velocity within the soil column can be computed according to $V_s(x) = \sqrt{\mu(x)/\rho(x)}$. Due to the boundary condition described by Eq. (1b) (i.e., a dashpot, subjected to a dynamic shear force), prescribed incident waves enter the domain, and outgoing waves leave the domain, without causing unintended reflections. Namely, the soil column is truncated by an absorbing boundary condition, where the incoming seismic wave is applied as a shear force. We consider the wave response for the truncated domain of length L during an observation period of T.

[2] E.g., deconvolution or gradient-based inversion.

Typically, finite-difference or finite-element methods are used to obtain a numerical solution to Eq. (1). Here, we use the finite element method with linear elements for spatial discretization, along with a second-order Newmark scheme for temporal discretization, which solves the following ordinary differential equation in time:

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{C}\dot{\mathbf{u}}(t) + \mathbf{K}\,\mathbf{u}(t) = \mathbf{F}(t), \tag{2}$$

where $\ddot{(\,)}$ and $\dot{(\,)}$ stand for the second and first derivative of the subtended variable with respect to time, respectively, and the matrices and vector are defined as:

$$\mathbf{K} = \int_0^L \mu(x)\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x}\frac{\mathrm{d}\boldsymbol{\phi}^T}{\mathrm{d}x}\mathrm{d}x, \qquad \mathbf{C} = \frac{\mu_{\mathrm{B}}}{V_{\mathrm{B}}}\boldsymbol{\phi}(0)\boldsymbol{\phi}^T(0) + \mathbf{C}_{\mathrm{R}},$$
$$\mathbf{M} = \int_0^L \rho(x)\boldsymbol{\phi}\boldsymbol{\phi}^T\mathrm{d}x, \qquad \mathbf{F}(t) = \frac{\mu_{\mathrm{B}}}{V_{\mathrm{B}}}\boldsymbol{\phi}(0)2v^{\mathrm{inc}}(t), \tag{3}$$

where $\mathbf{K}$, $\mathbf{C}$, and $\mathbf{M}$ denote the global stiffness, damping, and mass matrices, respectively, $\mathbf{F}(t)$ denotes the global force vector, and $\boldsymbol{\phi}$ indicates basis functions. The Rayleigh damping matrix $\mathbf{C}_{\mathrm{R}}$, in Eq. (3), is defined as:

$$\mathbf{C}_{\mathrm{R}} = c_0\mathbf{M} + c_1\mathbf{K}, \tag{4}$$

where c_0 and c_1 are scalar values computed as:

$$c_0 = 2\xi\frac{\omega_i\omega_j}{\omega_i+\omega_j}, \quad c_1 = 2\xi\frac{1}{\omega_i+\omega_j}, \tag{5}$$

where ξ is the damping ratio of a given soil; ω_i and ω_j are control angular frequencies, typically corresponding to the first and fifth natural mode, and are commonly calculated through [42]:

$$\omega_n = 2\pi\,(2n-1)\frac{\bar{V}_s}{4L}, \tag{6}$$

where n is the mode number; $\bar{V}_s$ is the average shear wave velocity; and L is the total length of the soil column.

We use our finite element wave simulator to generate data for our ANNs. The correctness of our simulator is verified in Appendix.

3. Methods

We investigate the effectiveness of using machine learning (ML) algorithms for identifying an incoming seismic wave. We focus on one-dimensional models, where seismic shear waves travel vertically through a soil column, and the waveform is measured at the top of the soil column, which indicates the ground surface.

We use a two-stage procedure to generate data sets for training and testing our ANNs: (a) we use a two-dimensional wave simulator to generate complex and realistic waveforms based on randomized wave source and stratigraphy. We then extract these waveforms at multiple locations within the two-dimensional domain; and (b) we use these waveforms as incoming wave motion, $v^{\text{inc}}(t)$, within a one-dimensional wave simulator, where each incoming wave motion (at the bottom of the soil column) is associated with a corresponding *measured* waveform (at the ground surface). This way, we generate a data set that associates 12,000 incident waveforms, $v^{\text{inc}}(t)$, to their corresponding measured waveforms, $u(L,t)$. By using this data set, we train DNNs and CNNs to relate a measured waveform, $u(L,t)$, (input-layer features) to its associated incident waveform, $v^{\text{inc}}(t)$, (output-layer features).

3.1. Deep neural network (DNN)

A typical DNN consists of data-processing modules, called layers, stacked on top of one another. Each layer consists of a mathematical function, called neurons, through which data and calculations move forward onto the next layer. The basic structure of a DNN is comprised of an input layer, hidden layers, and an output layer [43].

The input layer of a DNN is responsible for receiving the input data and passing them through to the hidden layers. In this study, 9000 training samples out of 12,000 records in our data set are used to train the DNN. Each record has 1801 time-step features, chosen to agree with the 0.01 s temporal discretization points used to span 18 s records we considered in this study. Our DNN architecture has four fully connected hidden layers, consisting of 2048, 2048, 3072, and 4096 neurons. The DNN is shown in Fig. 2, where the relation between the input- and output-layer features is learned from different batches of training data set through an iterative process. We designed the DNN architecture such that a minimal number of hidden layers are employed. If too many hidden layers in a DNN are used, the back-propagation process[3] will likely fail to identify an appropriate neural network parameterization. The learnable parameters of the DNN, i.e., weights and biases, are applied according to:

$$a_i = \sum_{j=1}^{d} w_{ij} q_j + b_i, \tag{7}$$

where w_{ij} is the weight coefficient between the two successive neurons at neighboring layers (i.e., previous and current layers); q_j is the jth feature (or neuron) from a previous layer; d is the size of the data from a previous layer; b_i is the bias associated with each neuron; and a_i is the outcome resulting from the calculation corresponding to the ith neuron of the current layer.

The weights and biases are initialized using the commonly-used "Xavier" initialization and, then, are updated through back-propagation by virtue of automatic differentiation. The Xavier initialization is chosen over the other commonly-used "He" initialization due to its better performance on our validation data sets. We omit to show the detailed comparison of the accuracy of the presented ANNs, in terms of the type of initialization, and refer the interested reader to Glorot and Bengio [44] and He et al. [45] for details on the Xavier and He initializations, respectively.

[3] I.e., automatic differentiation, followed by determination of the unknown parameters associated with all the neurons in a DNN.

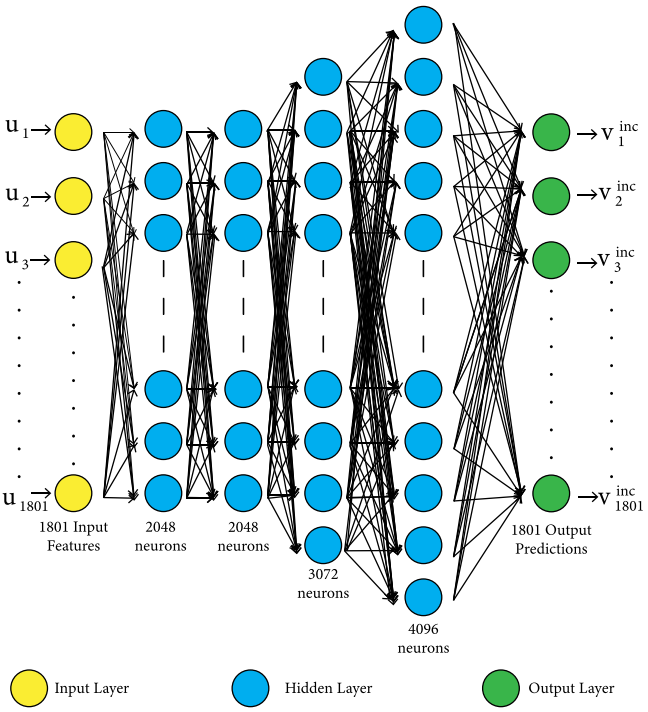


Fig. 2. Visual representation of our DNN architecture: u_j denotes the jth time-step of the measured signal, $u(L,t)$, and v_j^{inc} denotes the jth time-step of an unknown incident wave signal, $v^{\text{inc}}(t)$.

An activation function is added to each neuron so that the network is able to learn complex and possibly nonlinear patterns present in the data. We use Leaky Rectified Linear Unit (LReLU) as the activation function in our DNN model. The LReLU activation function is defined as:

$$r_i = f_{\text{LReLU}}(a_i) = \begin{cases} h \cdot a_i & \text{if } a_i < 0, \\ a_i & \text{if } a_i \geq 0, \end{cases} \tag{8}$$

where r_i is the outcome from the application of the activation function; $f_{\text{LReLU}}(a_i)$ is the application of the LReLu activation function on a_i; and h is a fixed parameter (0.3 in this case). The a_i values are passed through the LReLU activation function, and the r_i values are then passed as input (q_j) to the subsequent hidden layer, where the mathematical operation of weights and biases in Eq. (7) and the activation function in Eq. (8) are repeatedly applied. We note that employing the LReLu activation function is known to overcome the vanishing gradient problem, which typically arises when sigmoid and hyperbolic tangent (i.e., tanh) activation functions are used in multi-layered DNNs [46]. Robustness of the LReLu function against vanishing gradients typically allows the associated ANN models to learn faster and perform better.

The output values from the final hidden layer are then forwarded to the output layer, which consists of 1801 neurons. The graphical illustration of the DNN is presented in Fig. 2. The model is learned using the RMSprop optimizer with a learning rate of 0.01 and a momentum of 0.9, and it is allowed to train over 500 iterations (i.e. epochs) with a batch size of 20. We use Mean-Squared Error (MSE) as the loss function, which is defined as:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} (v_{ij}^{\text{inc}} - \hat{v}_{ij}^{\text{inc}})^2, \tag{9}$$

where n is the number of records in a batch; m is the number of time step features of the data; v_{ij}^{inc} is the ith record and jth time step feature of the targeted output-layer data; and $\hat{v}_{ij}^{\text{inc}}$ is the associated ANN-predicted output-layer data.

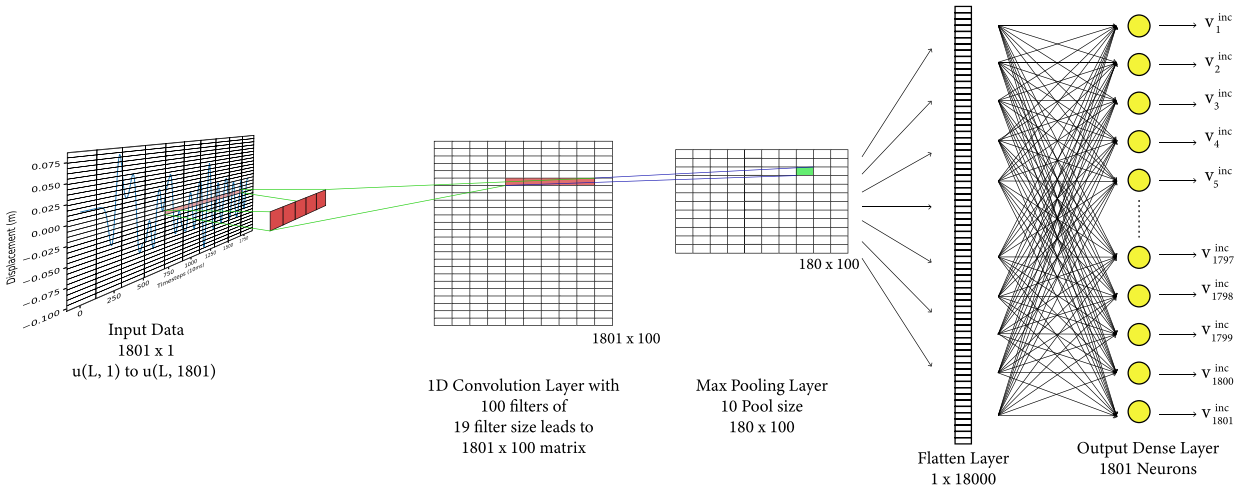


Fig. 3. Visual representation of our CNN architecture: u_j denotes the jth time-step of the measured signal, $u(L,t)$, and v_j^{inc} denotes the jth time-step of an unknown incident wave signal, $v^{\text{inc}}(t)$.

An error function is defined to judge the performance of the presented ANNs for an individual sample signal. Namely, the error function for the ith record (test sample) is defined as:

$$\epsilon_i = \sqrt{\frac{\sum_{j=1}^{m}(v_{ij}^{\text{inc}} - \hat{v}_{ij}^{\text{inc}})^2}{\sum_{j=1}^{m}(v_{ij}^{\text{inc}})^2}} \times 100\,[\%]. \tag{10}$$

In the first iteration of the training of the DNN, the initialized weights and biases are employed to predict an output. The loss is calculated by using Eq. (9), and the gradient of the loss function is computed with respect to each weight and bias in every layer by automated differentiation. The gradient is utilized for updating all the weight and bias parameters. These updated weight and bias values are then used to predict the new output values $\hat{v}_{ij}^{\text{inc}}$. The cycle of updating the weight and bias parameters continues for the specified number of iterations, or until the MSE reaches a minimum.

3.2. Convolutional neural network (CNN)

We also explore the effectiveness of using CNNs to solve the presented inverse source problem. Our CNN is composed of an input layer, one convolutional layer, a Max Pooling layer, a Flatten layer, and an output layer. A visual representation of the developed CNN architecture is shown in Fig. 3.

Within the convolutional layer, we use filters of a uniform size. A filter is a vector of a particular filter size with weight coefficients that are initialized using the Xavier initialization. The presented CNN includes a Max Pooling layer to reduce parameters, extract important features, and combat over-fitting. The Flatten Layer allows the feature map extracted from the Max Pooling layer to be used as input to the output layer. The generated data are fed into the input layer of the CNN in the form of a time-series signal.

Our convolution layer has 100 filters, each with a filter size of 19, and receives the data from the input layer for the convolution operation. This layer, after training, aims to capture feature patterns by convolving the filters over the input data in a sliding dot-product computation window. Namely, the 100 filters slide across each vector with a stride of 1, multiplying the filter weight by the overlapping feature-vector value and summing the dot product computation. This particular algebraic operation is illustrated in Fig. 4. Padding is added to maintain the output length dimension of the convolution layer based on the required time step features of 1801. The resulting computed values are then passed through the LReLu activation function, and this CNN layer results in an output feature-map matrix of (1801, 100) dimensions. The second dimension of the matrix, 100, is due to the 100 different filters convolving through the input data.

The output from the convolution layer passes through a Max Pooling layer, which takes the maximum value over a window length of 10 along the time-series dimension of the (1801, 100) matrix, reducing the dimensionality from (1801, 100) to (180, 100) and allowing the layer to learn feature-relations based on only the prominent features. The Flatten layer converts the matrix outcome from the Max Pooling layer into a vector of (18 000, 1) dimension. This layer is implemented to pass the flattened feature maps from the Max Pooling layer onto the final output layer. The neurons in the last output layer are fully connected to every neuron in the previous Flatten layer, through learnable parameters (i.e., weights and biases), and it has 1801 neurons to replicate an 18 s output waveform.

The parameters (i.e., filter weights and the weights and bias connecting the flatten layer and the output layer) in CNN are optimized using an Adam optimizer, with a learning rate of 0.001, where the MSE loss function is used. The model is trained over 350 epochs, with a batch size of 35. Similar to DNN, CNN optimizes both its filter's weights and dense layer weights through back-propagation by using automatic differentiation.

3.3. Selection of DNN and CNN hyper-parameters

The selection of the hyper-parameters in the presented neural network architectures is performed through an extensive manual search. To design the DNN architecture, we manually experimented with the following hyperparameters: different weights and bias initializers, activation functions, optimizers, loss functions, learning rates, number of hidden layers and neurons, number of iterations, and the batch size of the training data set. Our only initial reference point is the size of the input-layer waveform, which has 1801 time steps.

For our DNN, we initialize a single hidden layer, consisting of at least 1801 neurons, and monitor its performance. We are aware that using a hidden layer with neurons less than the number of time steps would result in a compression of feature representation, which could potentially hamper the performance of the DNN, resulting in a "bottleneck" condition. We continuously increased the number of neurons in the first hidden layer until the results in the validation data set predictions stopped improving. Using this as a guide, we moved towards defining a second hidden layer, with the same number of neurons considered in the first hidden layer. This iterative process was applied to multiple hyper-parameters mentioned above. We continued this process until the model had more than 4 hidden layers and higher neuron counts, which then caused the model to overfit,

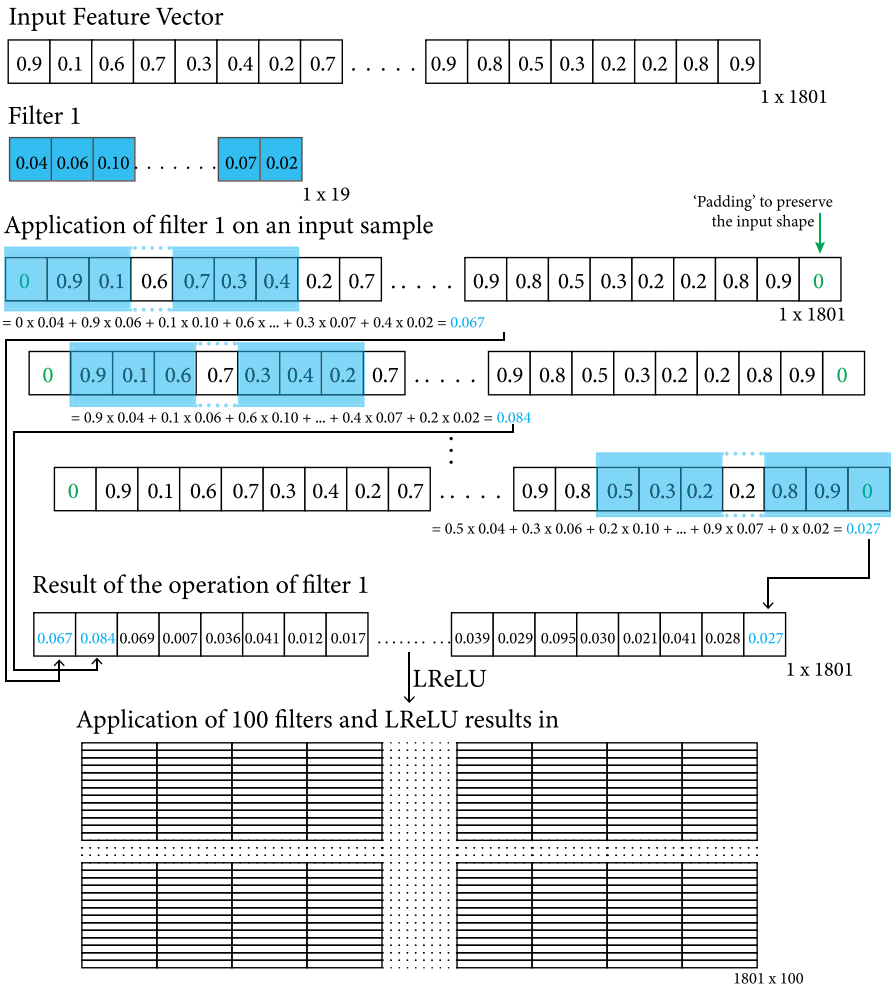


Fig. 4. Exemplary convolution operation on the input data set. Numbers are not representative of the actual feature input or filter weights.

resulting in poor results on the validation data set. Therefore, to combat overfitting, we decreased our learning rate, and implemented dropout and regularization penalty techniques. Through multiple iterations of this experimental process, we arrived at the conclusion that using the simpler architecture presented earlier (i.e., the one with 4 hidden layers) results in a shorter training time and a smaller error in the validation data set (as well as the test data set), compared to the latter case where a more elaborate model[4] was used.

The selection of hyper-parameters for the presented CNN follows a similar iterative process. The hyper-parameters experimented with for the CNNs are: number of convolutional layers, filters and filter sizes, max pooling layer and pool size, and different filter, weights, and bias initializers, activation functions, optimizers, loss functions, learning rate, number of iterations, and the batch size of the training data set.

3.4. Generation of training, validation, and test data sets

3.4.1. Waveform randomizer for generation of the output-layer feature data set: $v^{inc}(t)$

We design a waveform randomizer to generate training data for the incoming seismic waveforms $v^{inc}(t)$, which are used as the output-layer features of the presented DNN and CNN. To build our waveform randomizer, we consider anti-plane shear waves, propagating in a two-dimensional domain, which is appropriately truncated by using wave-absorbing boundary conditions (WABC). We take advantage of reflection and refraction in a heterogeneous domain in order to generate complex waveforms. Namely, the two-dimensional model only needs to enable the generation of complex randomized waveforms by virtue of reflection and refraction. This can be achieved through a series of randomized (over space and time) body-force sources within a simple, multi-layered two-dimensional domain.

Fig. 5 shows the geometry and material profile of the domain we used to generate randomized waveforms. The domain has four layers and three rectangular inclusions. Its dimension is 60 m × 60 m, and its mass density is 1500 kg/m^3. Eight body forces, with a Ricker wavelet temporal signature, consecutively generate wave motion in the domain. Each body force is applied at a different location and with a different central frequency and peak amplitude. The temporal interval between each Ricker wavelet is 1.5 s. Namely, the first Ricker wavelet is applied at $t = 0$ s, the second one at $t = 1.5$ s, the third one at $t = 3$ s, and so on. Then, by using a total observation time of 18 s, a time step of 0.01 s, and an element size of 1 m, the two-dimensional shear wave propagation is solved through the finite element method (FEM). Next, the particle velocity time-histories are extracted from the twelve pre-defined locations in the domain (Fig. 5). The randomizer is set to run 1000 iterations, in order to generate 12,000 particle velocity records. Algorithm 1 describes the details of the randomizer. Note, at each iteration of the Algorithm, parameters that characterize the wave motion inside the domain are changed randomly in order to generate a diverse data set. The extracted particle velocity records are then

[4] I.e., lowering the learning rate, implementing dropout, and applying regularization methods were utilized.

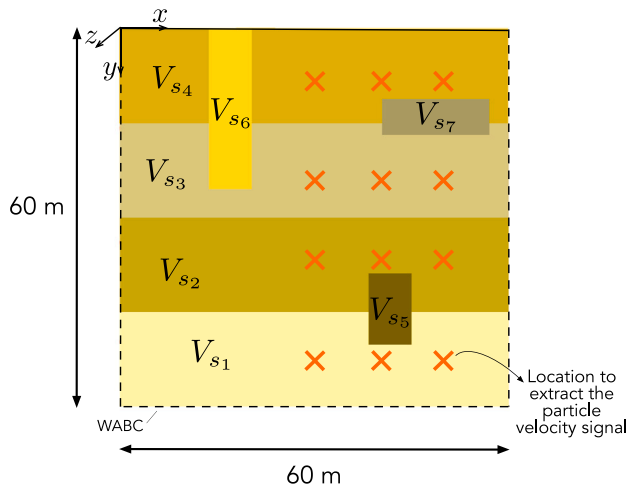


Fig. 5. Two-dimensional domain used to generate the randomized output-layer data set (i.e., incoming (incident) seismic wave, $v^{\text{inc}}(t)$, which are later used in a one-dimensional wave simulator). Complex wave patterns form due to reflection and refraction in the heterogeneous media. V_{s_1} to V_{s_7} are randomized as we generate the data set.

filtered, by using a low-pass filter, and then are used as the output-layer feature for training, validating, and testing our data-informed framework for incident seismic-wave motion identification.

Algorithm 1 Two-dimensional waveform randomizer to generate training, validation, and test data sets

1: Set the total number of iterations = 1000 and the number of body forces = 8.
2: **for** iteration ← 1 **to** (total number of iterations) **do**
3: Randomly set the values of shear wave speed (V_s) of each soil layer and inclusion in the domain by using the following ranges:
$280 \le V_{s_1} \le 500$ m/s,
$220 \le V_{s_2} \le V_{s_1}$,
$180 \le V_{s_3} \le V_{s_2}$,
$130 \le V_{s_4} \le V_{s_3}$,
$800 \le V_{s_5}, V_{s_6}, V_{s_7} \le 1200$ m/s.
4: Build mass, stiffness, and damping matrices.
5: **for** body force index ← 1 **to** (number of body forces) **do**
6: The amplitude (P), frequency (f), and horizontal and vertical coordinates of the location (s_x and s_y respectively) of each Ricker wavelet are randomly chosen within the following ranges:
$10 \le P \le 50$ N/m^2,
$0.1 \le f \le 0.8$ Hz,
$5 \le s_x \le 30$ m,
$5 \le s_y \le 55$ m.
7: (body force index) ← (body force index)+1.
8: **end for**
9: Solve the 2D wave propagation problem.
10: Extract the particle velocity signals from twelve different defined locations.
11: Use a low-pass filter to smooth each signal.
12: Save each filtered particle velocity signal to be used as an incoming (incident) seismic signal, $v^{\text{inc}}(t)$, in the one-dimensional wave solver, and as output-layer features in the presented DNN and CNN.
13: iteration ← iteration+1.
14: **end for**

3.4.2. Generation of input-layer feature data set: $u(L,t)$

The waveforms generated in the previous subsection are now used as the incident wave signal (i.e., $v^{\text{inc}}(t)$) in a one-dimensional wave simulator, described by Eq. (1), and shown in Fig. 1(c). The one-dimensional wave system appropriately accounts for the truncation boundary by using a viscous damper and uses a shear force proportional to the incoming incident wave to drive the system [41]. A sensor is placed at the top surface of the soil column to record the displacement field of the wave response induced by each $v^{\text{inc}}(t)$. Then, the measured seismic record ($u(L,t)$) is used as the input-layer feature in our ANN-based source inversion framework. In this study, we consider two different soil profiles in the one-dimensional wave system:

- **A soil column with an asymptotically-varying shear wave speed** with a length of 100 m, mass density of $\rho = 1500$ kg/m^3, damping ratio of $\xi = 0.02$, and target frequencies for the Rayleigh damping of $f_i = 0.67$ Hz and $f_j = 3.36$ Hz. As shown in Fig. 6(a), the shear wave speed varies asymptotically from 150 m/s at the ground surface to 300 m/s at the bottom.
- **A 4-layer soil column**, as shown in Fig. 6(b), where each layer is 25 m, with shear wave speeds of $V_{s_1} = 300$ m/s, $V_{s_2} = 250$ m/s, $V_{s_3} = 200$ m/s, and $V_{s_4} = 150$ m/s; the mass density of the entire column is $\rho = 1500$ kg/m^3; the damping ratios are $\xi_1 = 0.01$, $\xi_2 = 0.01$, $\xi_3 = 0.02$, and $\xi_4 = 0.03$; and the target frequencies are $f_i = 0.56$ Hz and $f_j = 2.81$ Hz.

The properties of the underlying bedrock are the same for both soil columns. Its shear wave speed is $V_B = 1500$ m/s, and its mass density is $\rho_B = 2000$ kg/m^3. The wave motion in the domain is calculated by using an element size of 0.01 m and a time step of 0.01 s. Per the two types of ANNs and the pair of soil profiles, we train a total of four ANNs: DNN and CNN for the asymptotically-varying soil profile and those corresponding to the 4-layered soil profile.

3.5. Data preparation

The data sets generated from the one-dimensional wave simulator are used to train two separate ANNs. The generated input-layer or output-layer waveforms have a matrix size of 12,000 samples and 1801 time steps of wave time-history data due to recording of 18 s signals at every 0.01 s. The input-layer feeds on measured displacement time-histories, $u(L,t)$, and the output-layer takes the incident waveform, $v^{\text{inc}}(t)$. Sample input-layer and output-layer waveforms are shown in Figs. 7(a) and 7(b).

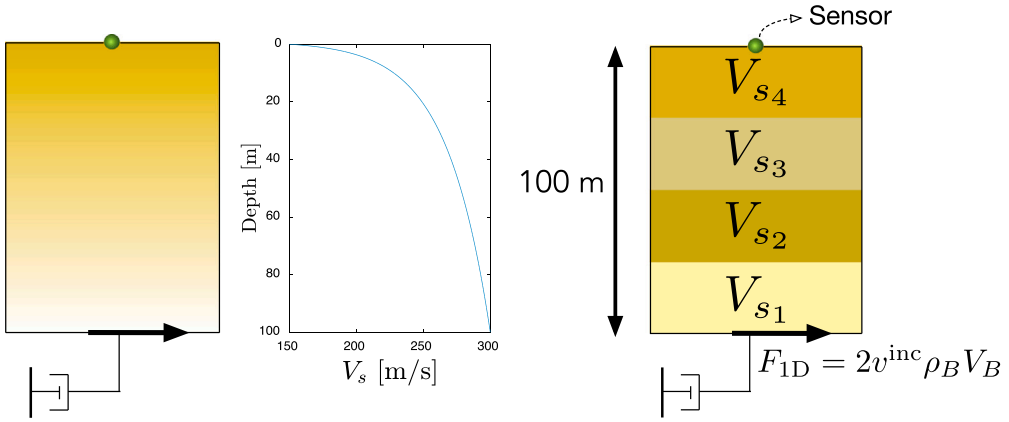
Figs. 7(a) and 7(b) show that the ordinates, both displacements and velocities, are in the range of 10^{-6} or 10^{-7}. These extremely small range of values make it difficult for the ANNs to learn critical feature-to-feature relations and patterns. To mitigate this issue, both the input- and the output-layer waveform values are feature-wise normalized such that their values lie between −1 to 1.

Before normalization, the entire data set is divided into three subsets of 9000, 1000, and 2000 samples of training, validation, and test data set, respectively. The 9000 samples are allocated for training and expose the ANNs to complex patterns between the input-layer and the output-layer data set. The validation subset is used to monitor the impartial performance of the ANN on a data set unseen by the model during the training phase in order to stop the training process if the model shows the signs of over-fitting. The test subset is used to provide an unbiased evaluation of the trained ANN by defining an error function.

The normalizing parameters – mean, maximum, and minimum – are used to feature-wise normalize both the input- and output-layer parts in the training, validation, and test data sets. The normalizing parameters are computed only from the training data set, in order to leave the validation and test data sets intact, to allow for a fair analysis of the ANNs [47]. Eq. (11) is used to normalize the data sets:

$$A_{ij}^{n} = \frac{A_{ij} - A_{\text{train}}^{\text{mean}}}{A_{\text{train}}^{\max} - A_{\text{train}}^{\min}}, \tag{11}$$

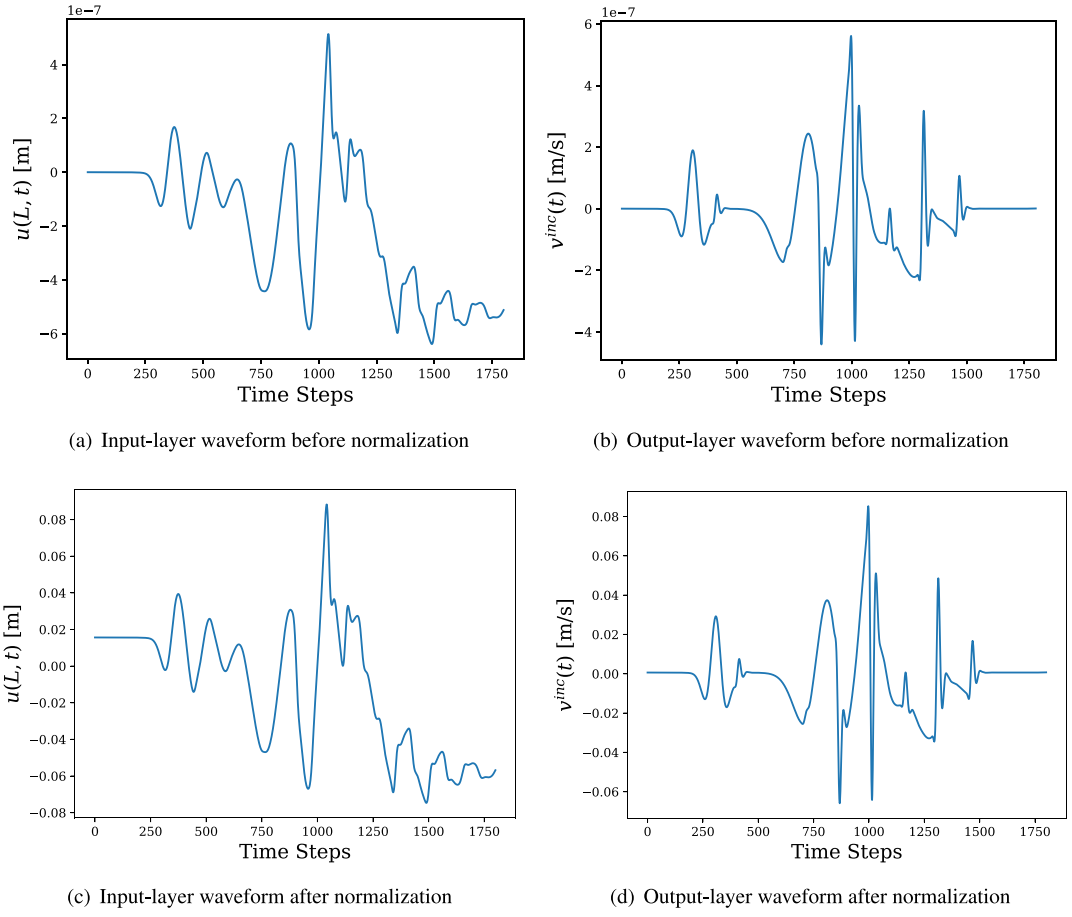
where A_{ij}^{n} is the normalized component of the data matrix – such as the training, validation, or test set (either input or output layer) – of which the ith row and jth column indicate the jth time-step of the ith sample; A_{ij} is the un-normalized component of the data matrix; $A_{\text{train}}^{\text{mean}}$ is the



(a) soil column where wave speed varies asymptotically over depth

(b) 4-layered soil column

Fig. 6. One-dimensional soil column configuration: the incident wave, $v^{inc}(t)$, is used in the shear force calculation, $F_{1D} = 2v^{inc}\rho_B V_B$, at the bottom of the soil column.



(a) Input-layer waveform before normalization

(b) Output-layer waveform before normalization

(c) Input-layer waveform after normalization

(d) Output-layer waveform after normalization

Fig. 7. Exemplary waveforms from the training data for the soil column with asymptotically varying shear modulus over depth.

mean of the training set of the data matrix; and A^{max}_{train} and A^{min}_{train} are the maximum and minimum values of the training set of the data matrix, respectively. The mean, maximum, and minimum values are retained for the eventual de-normalization of the predicted data, which are obtained from the application of ANNs. The vertical-axes of Figs. 7(c) and 7(d) show the normalized scales of data, i.e., from orders around 10^{-6} or 10^{-7} to values between −1 to 1. The shape of the waveform does not change due to the linear nature of the normalization. Lastly, the normalized data are fed into the constructed DNNs and CNNs, aiming to create a data-informed framework that is capable of seismic source inversion.

4. Numerical experiments and performance evaluation

The DNN and CNN architectures, trained to predict waveforms on the output layer, are evaluated on the basis of accuracy, which is determined by the evaluation metric described by Eq. (10). We use the TensorFlow 2 framework and take advantage of its training tools, such

as automated differentiation, in order to train the DNN and CNN. We train both the DNN and CNN for two different soil profiles, described in Section 3.4.2, by using data sets generated according to the procedure detailed in Section 3.4.

4.1. Example 1: soil column with an asymptotically-varying shear wave speed

Our trained DNN, when tested on the 2000 test data sets, predicts a targeted output-layer waveform, with an average error of 0.27%, a median error of 0.21%, a maximum error of 2.37%, and a minimum error of 0.07%. The decrease in MSE during the training process of our DNN over iterations (epochs) is presented in Fig. 8(a), where the MSE for the training data set and that for the validation data set are shown individually. The value of MSE for the training data set is smaller because the coefficients of the DNN are determined by using the training data set. The distribution of the error metric (Eq. (10)) of DNN-predicted waveforms over the 2000 test data samples is presented in Fig. 8(b). Furthermore, the two best and two worst waveforms, among all the samples, based on the prediction accuracy of the waveforms, are presented in Figs. 9 and 10, respectively. Fig. 10 (worst cases) shows that the DNN predicts a targeted waveform fairly accurately, except at regions with sharp spikes, which corresponds to high-frequency content. Because the FEM inherently performs as a low-pass filter, and the damped soil model enhances the low-pass filtering effects, it is hard for the ANNs to find the relationship between the high-frequency content (e.g., spike) in an incident signal and a measured motion at the ground surface. Accordingly, we found when a low-pass filter, with a high-cut of 10 Hz, is applied to the incident waveforms of the training set, the prediction error of ANNs decreases in the test data set. It is reasonable to assume that frequency contents above 10 Hz are negligible in typical particle velocity recordings, especially if the recording has occurred relatively away from a hypocenter because geomaterials serve as natural low-pass filters. Therefore, the frequency content of the measured seismic records in our data sets, for the low-pass filtered incident waveforms, is likely to be similar to that of a realistic signal.

Similar to the DNN, the CNN is also used to predict the 2000 output-layer waveforms for the test samples. This results in a mean error of 0.05%, a median error of 0.04%, the highest error of 0.83%, and the lowest error of 0.02%. Fig. 11(a) shows the reduction of MSE over iterations during the CNN training. Fig. 11(b) represents the error distribution of the CNN prediction on the 2000 test samples. Two of the best and worst predicted waveforms using our CNN are shown in Figs. 12 and 13, respectively. Fig. 13 demonstrates that even the worst CNN-predicted waveforms have very small errors (0.69% and 0.83%).

In addition to the successful prediction of the 2000 waveforms in the test data set, our trained DNN and CNN are also examined for the reconstruction of a real-world incident waveform. We use the waveform corresponding to the 1994 Northridge earthquake in California, which is considerably different from records we generated with our waveform randomizer. To this end, we feed the 1994 Northridge record to our one-dimensional wave simulator as an input, and then, measure the corresponding waveform at the top of the soil column. The measured waveform is then fed into our DNN and CNN to predict the incoming wave motion. The waveforms generated by the DNN and CNN, trained as mentioned earlier, are presented in Figs. 14(a) and 14(b), respectively. Overall, our data-informed framework has reconstructed the incoming Northridge waveform satisfactorily. The prediction error is 9.07% for the DNN and 3.81% for the CNN.

4.2. Example 2: 4-layer soil column

We examine the performance of our DNN and CNN for a different soil profile now. When the 4-layer soil column described in Section 3.4.2 is used, our DNN produced a mean average error of 0.26%, a median error of 0.19%, a maximum error of 2.73%, and a minimum error of 0.08% on the test samples. The decrease in MSE over multiple iterations during the training of the DNN and the error distribution associated with the test data set are shown in Figs. 15(a) and 15(b), respectively. Two of the best and worst DNN-predicted waveforms in the test data set are presented in Figs. 16 and 17, respectively.

Table 1
Performance summary of the ANNs for Example 1: asymptotically-varying soil column, and Example 2: 4-layered soil column.

Error type/data set	Example 1		Example 2	
	DNN	CNN	DNN	CNN
Mean error of the test data set	0.27%	0.05%	0.26%	0.06%
Median error of the test data set	0.21%	0.04%	0.19%	0.05%
Maximum error of the test data set	2.37%	0.83%	2.73%	0.80%
Minimum error of the test data set	0.07%	0.02%	0.08%	0.02%
Error for the Northridge earthquake record	9.07%	3.81%	7.68%	3.22%

The CNN architecture for the 4-layer soil column, produced an average error of 0.06%, with a median error of 0.05% on the test data set. The highest and lowest errors are 0.80%, and 0.02%, respectively. The reduction in MSE over multiple iterations of CNN training is presented in Fig. 18(a), and the prediction error associated with the test data set is shown in Fig. 18(b). Two of the best and the worst CNN-predicted waveforms are shown in Figs. 19 and 20, respectively.

Next, we use the DNN and CNN to reconstruct the 1994 Northridge earthquake waveform, similar to what we did in Section 4.1. The predicted results are presented in Figs. 21(a) and 21(b). The prediction error is 7.68% for the DNN and 3.22% for CNN.

The performance of the DNN and CNN for both soil profiles is summarized in Table 1.

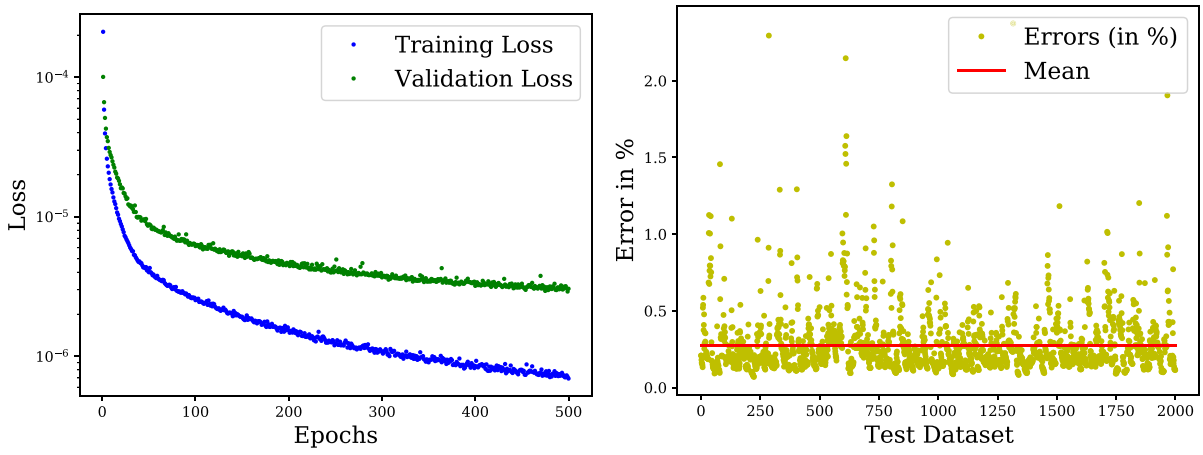
4.3. Effects of noise in measured data on the performance of the ANNs

To examine the robustness of the ANNs in realistic settings, where measured data are likely contaminated with noise, we consider different levels of noise, from 1% to 5%, during the training process. We generate the noisy waveform, $u^{\text{noisy}}(L,t)$, through:

$$u^{\text{noisy}}(L,t) = u(L,t) + B\ u^{\max}(L,t)\ \delta, \tag{12}$$

where B is an array of normally-distributed random numbers ranging $(-1,1)$ and contains m (i.e., the number of the time steps in the data) components; $u^{\max}(L,t)$ is the maximum value of the measured displacement at the receiver on a considered soil column; and δ is the noise level.

Next, we assess the performance of CNN and DNN predictions for the 1994 Northridge earthquake record, where noise has been added to the measured signal per Eq. (12). Table 2 shows the summary of our results: the first row of the table corresponds to a DNN that was trained with a noise-free data set. We then evaluate the performance of this DNN for the reconstruction of the Northridge earthquake waveform, where its associated input-layer data are contaminated with varying levels of noise. The performance of the DNN under these levels of noise seems satisfactory. In subsequent rows of Table 2, we add noise to the data set used to train the DNN, and evaluate the performance of the neural network for the Northridge earthquake waveform with varying levels of noise in measured signals. Overall, it appears that the DNN performs robustly when noise is present in the measured data. The corresponding results for CNN are presented in Table 3. All these results correspond to the soil profile with the asymptotically-varying shear wave speed. The corresponding results for DNN and CNN, for the case with the 4-layer soil profile, is presented in Tables 4, and 5, respectively. These results are also shown graphically in Figs. 22 and 23, for the two soil profiles, and all indicate the robust performance of the DNN and CNN when noise is present. Both Figs. 22(a) and



(a) MSE over 500 iterations (b) Error distribution across the test data set

Fig. 8. Example 1: The MSE history for training and validation samples, and performance of the DNN on test samples for a soil column with an asymptotically-varying shear wave speed.

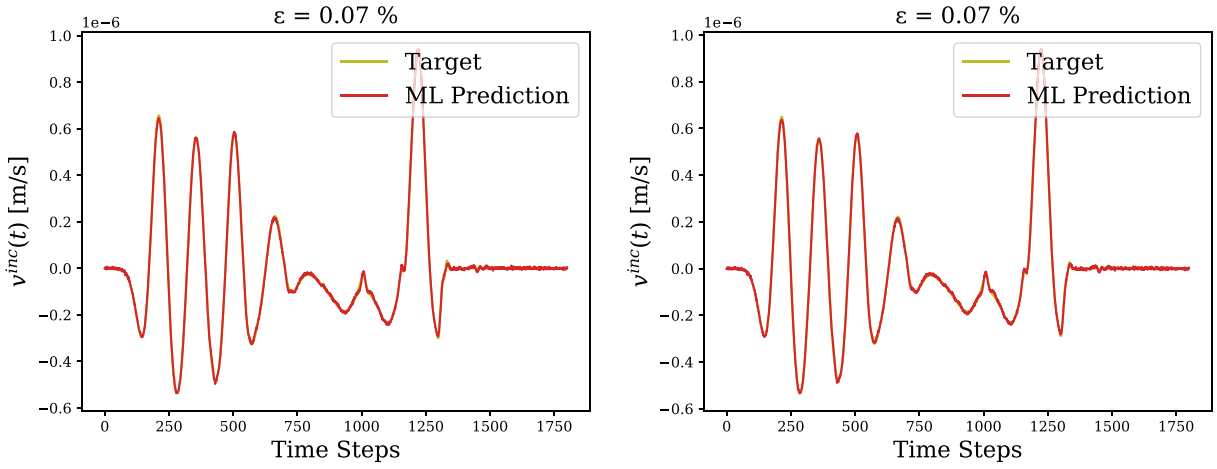


Fig. 9. Example 1: The two best waveform predictions, among the test samples, predicted by using the DNN for the soil column with an asymptotically-varying shear wave speed.

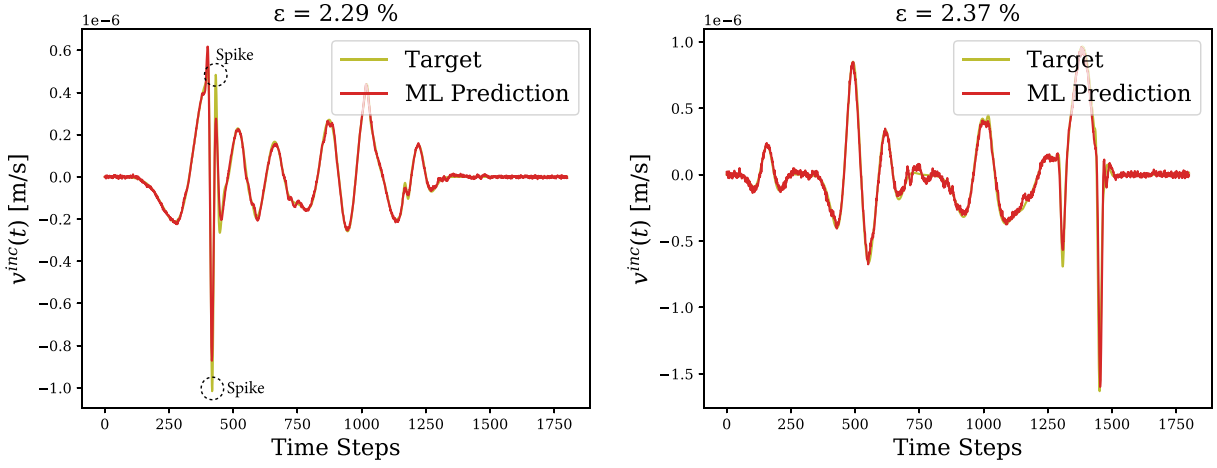
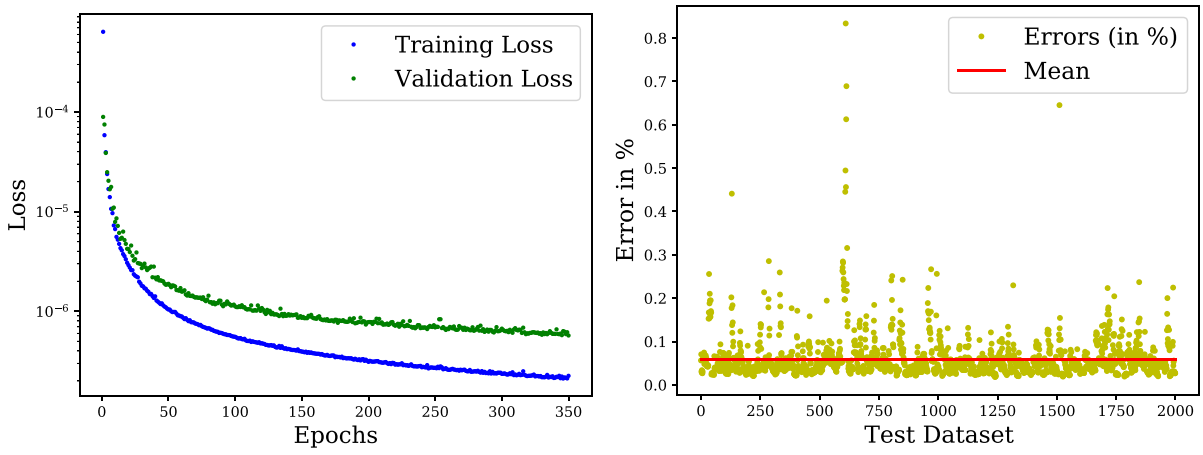


Fig. 10. Example 1: The two worst waveform predictions, among the test samples, predicted by using the DNN for the soil column with an asymptotically-varying shear wave speed.

23(a) show a similar trend where the performance of a DNN trained without noise worsens as higher levels of noise is introduced in test data sets. However, the performance of DNNs trained with noise levels between 1% to 5% shows better results with increasing levels of noise.



(a) MSE over 500 iterations

(b) Error distribution across the test data set.

Fig. 11. Example 1: The MSE history and performance of the CNN on the test samples for a soil column with an asymptotically-varying shear wave speed.

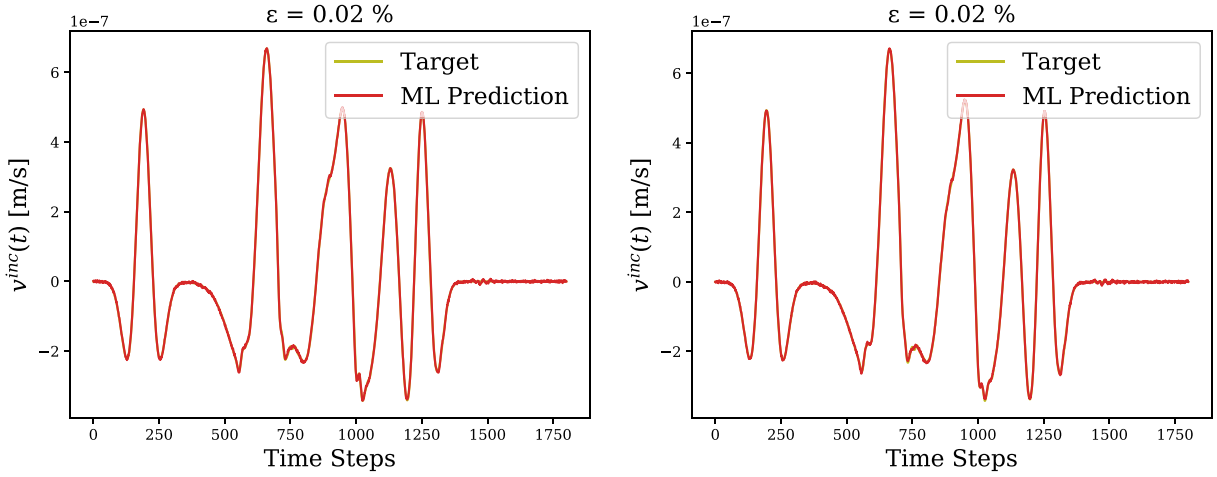


Fig. 12. Example 1: The two best waveform predictions, among the test samples, predicted by using the CNN for the soil column with an asymptotically-varying shear wave speed.

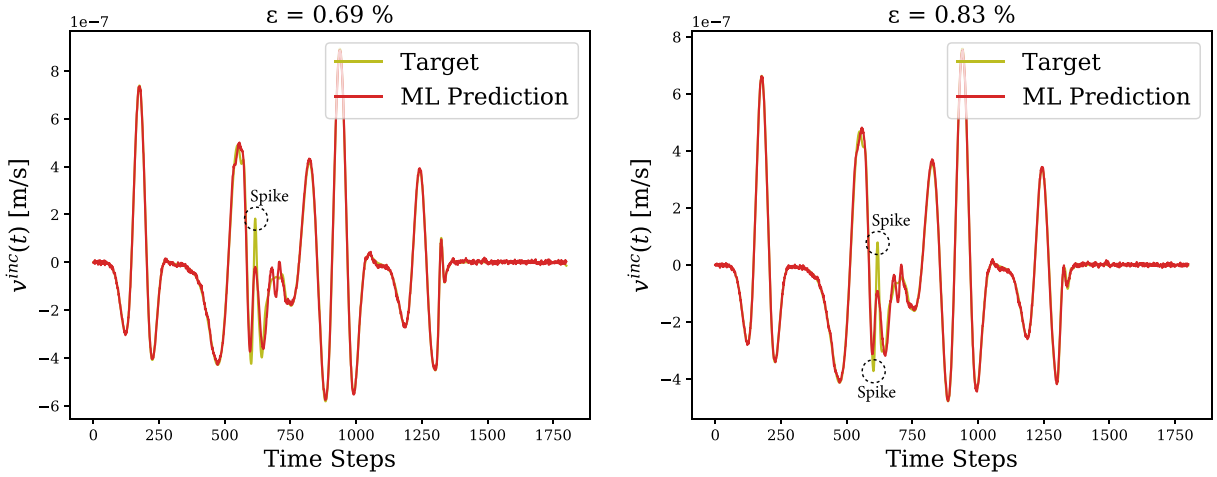
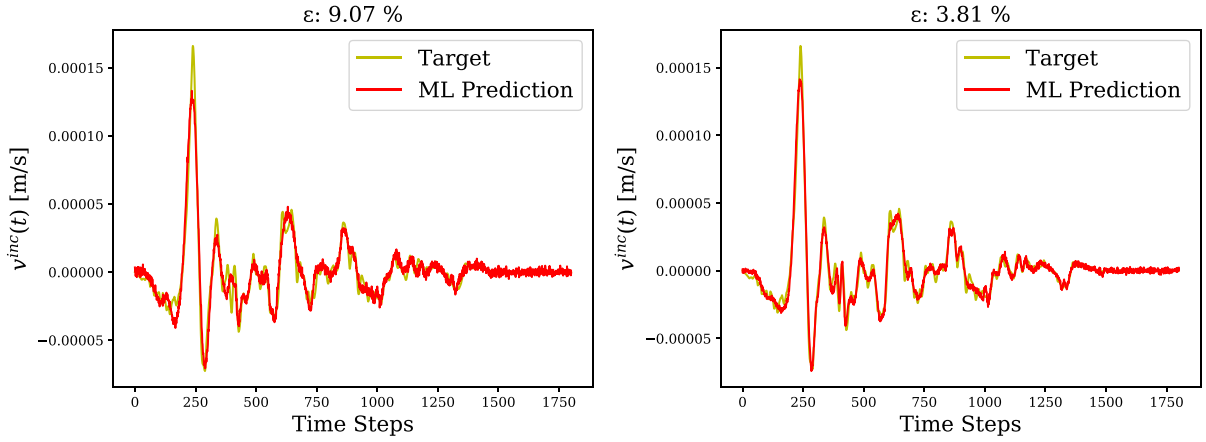


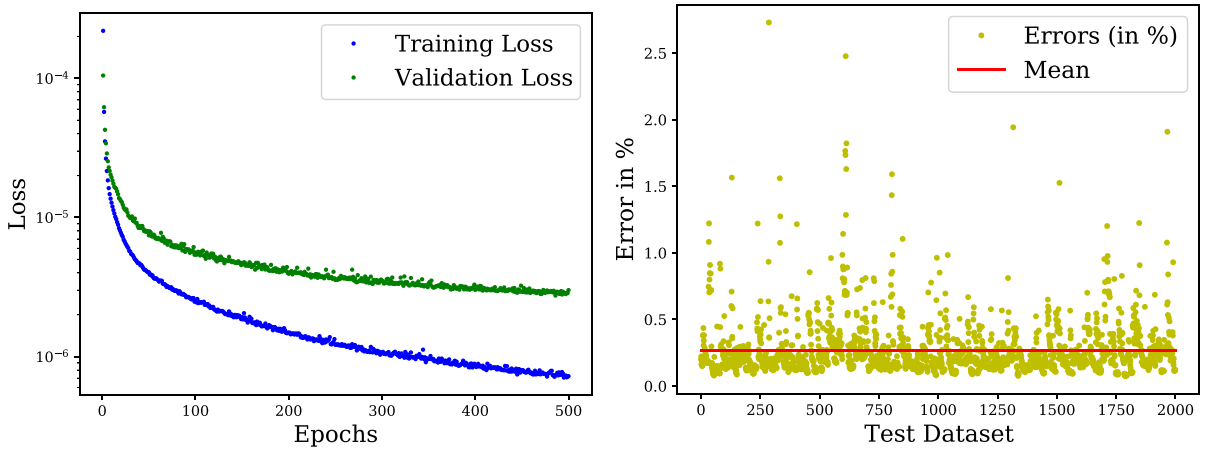
Fig. 13. Example 1: The two worst waveform predictions, among the test samples, predicted by using the CNN for the soil column with an asymptotically-varying shear wave speed.

Furthermore, a similar trend is observed in Figs. 22(b) and 23(b), but CNNs trained with noise levels between 1% to 5% perform a bit worse as the noise level increases in the test data sets. Overall, the performances of both DNN and CNN are satisfactory. However, the



(a) DNN (b) CNN

Fig. 14. Example 1: The reconstruction of the 1994 Northridge earthquake incident waveform, using the DNN and CNN, for the soil column with an asymptotically-varying shear wave speed.



(a) MSE over 500 iterations. (b) Error distribution across test data set.

Fig. 15. Example 2: The MSE history, and performance of the DNN for the 4-layer soil column.

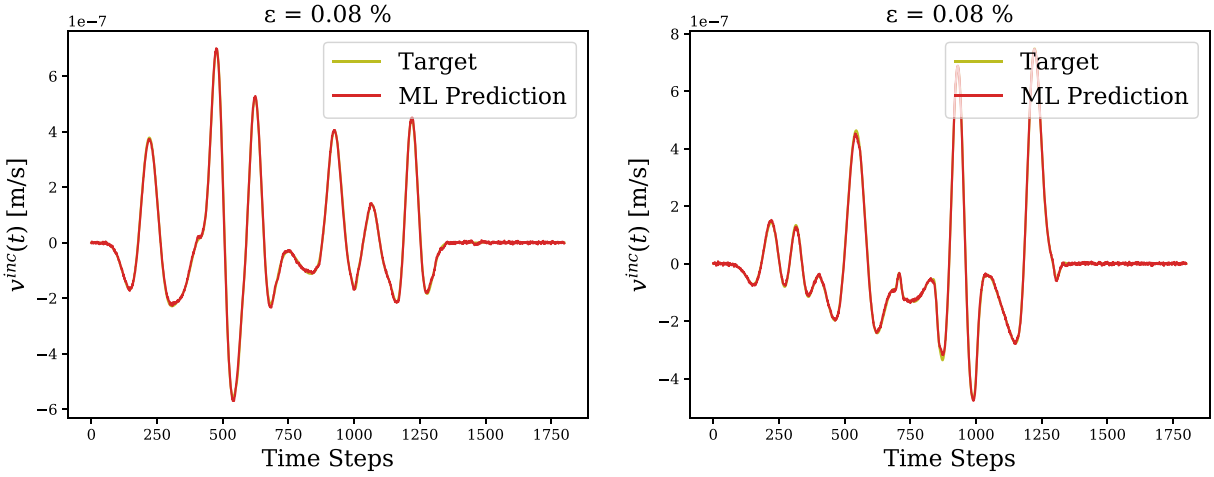


Fig. 16. Example 2: The two best waveforms, among the test samples, predicted by using the DNN for the 4-layer soil column.

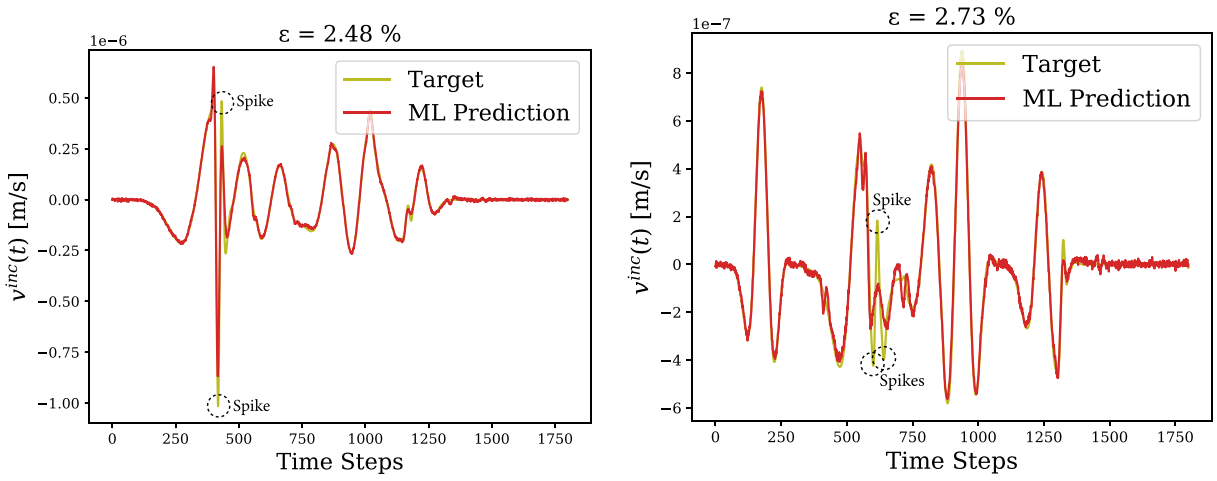
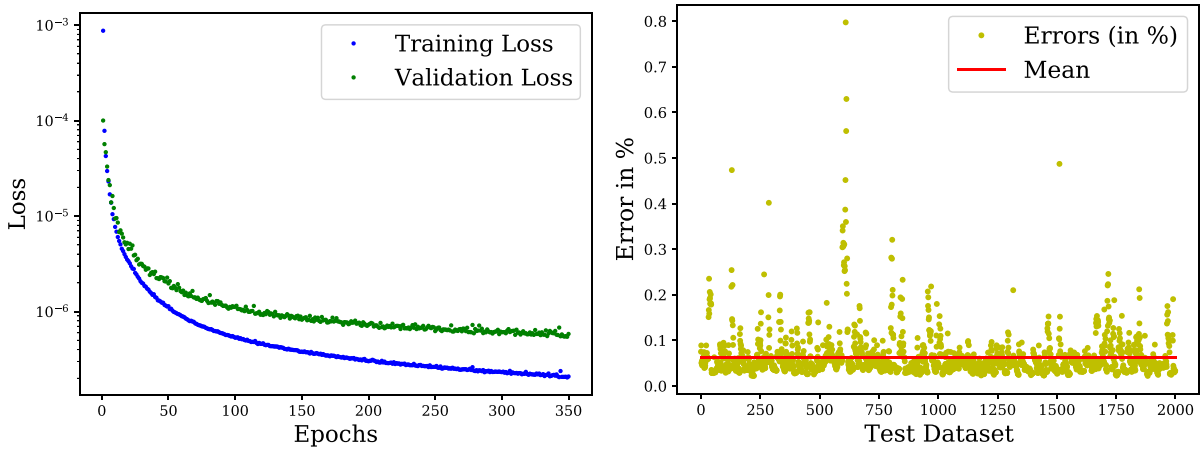


Fig. 17. Example 2: The two worst waveforms, among the test samples, predicted by using the DNN for the 4-layer soil column.



(a) MSE over 500 iterations

(b) Error distribution across test data set.

Fig. 18. Example 2: MSE history, and performance of the CNN for the 4-layer soil column.

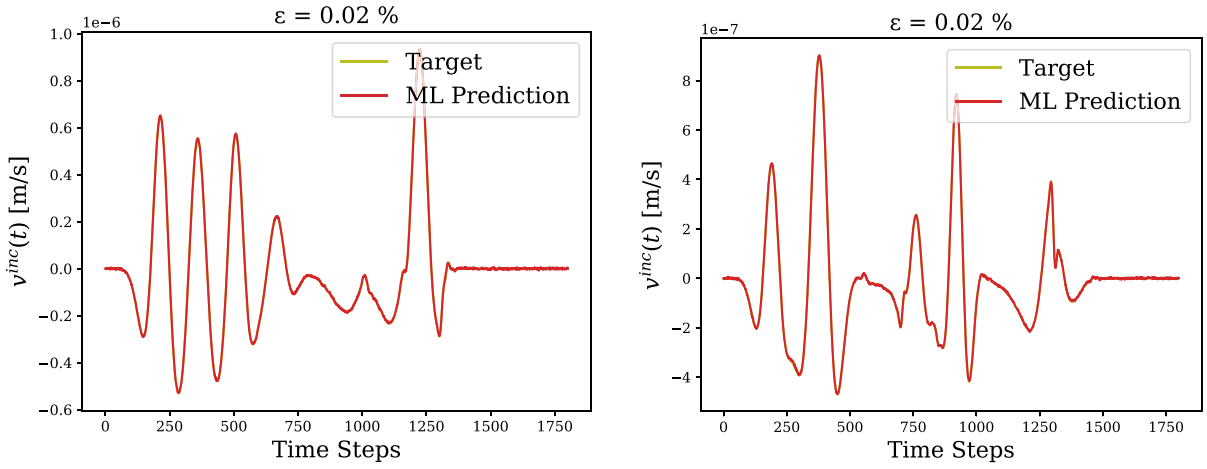


Fig. 19. Example 2: The two best waveforms, among the test samples, predicted by using the CNN for the 4-layer soil column.

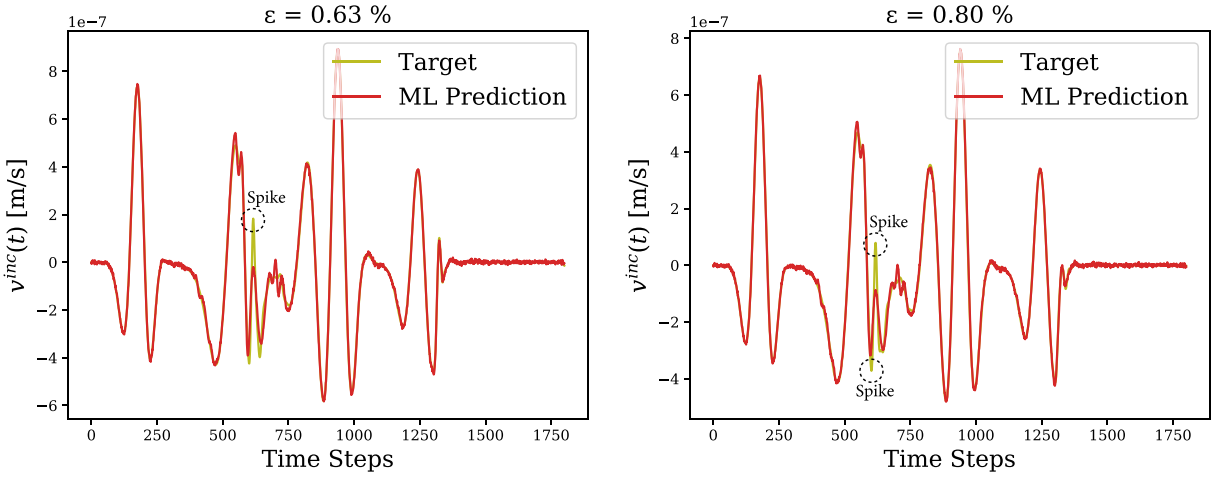
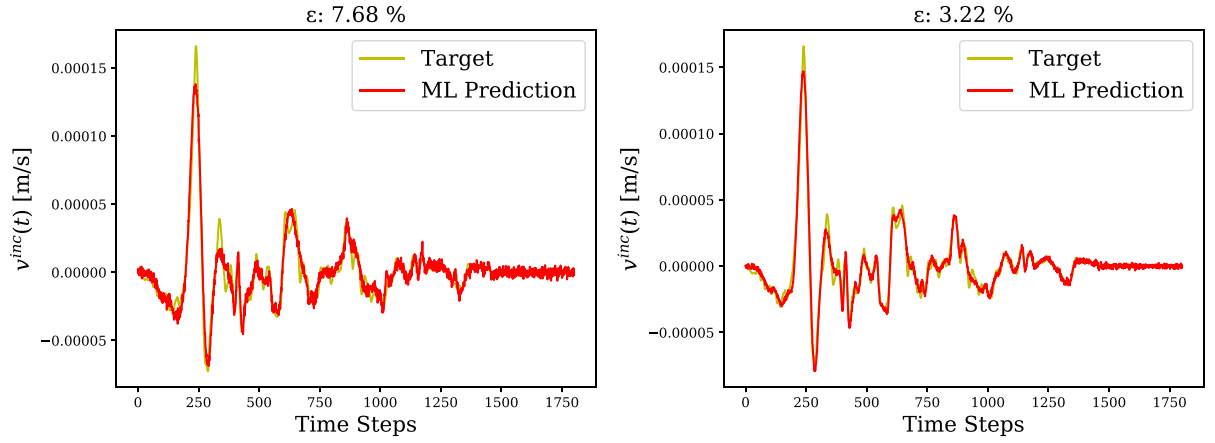
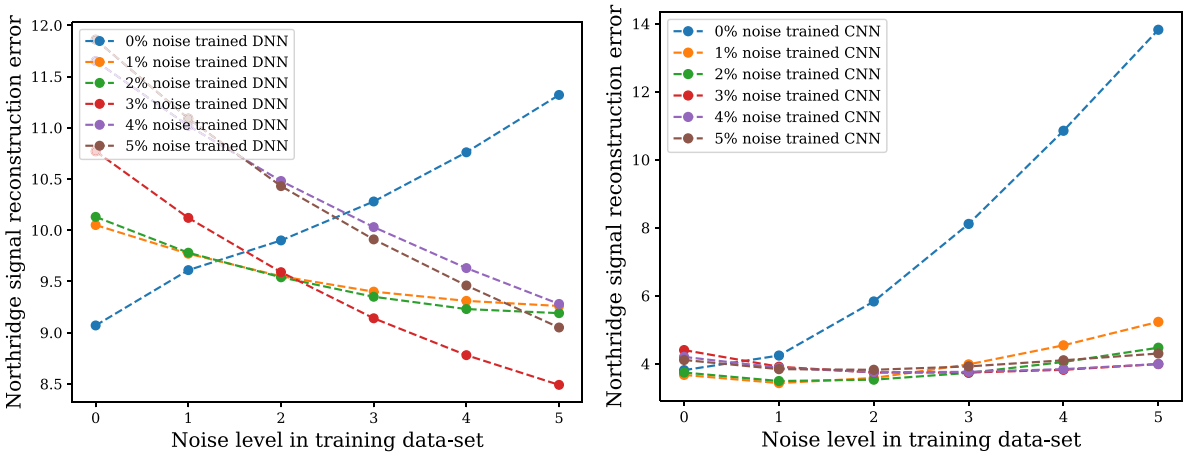


Fig. 20. Example 2: The two worst waveforms, among the test samples, predicted by using the CNN for the 4-layer soil column.



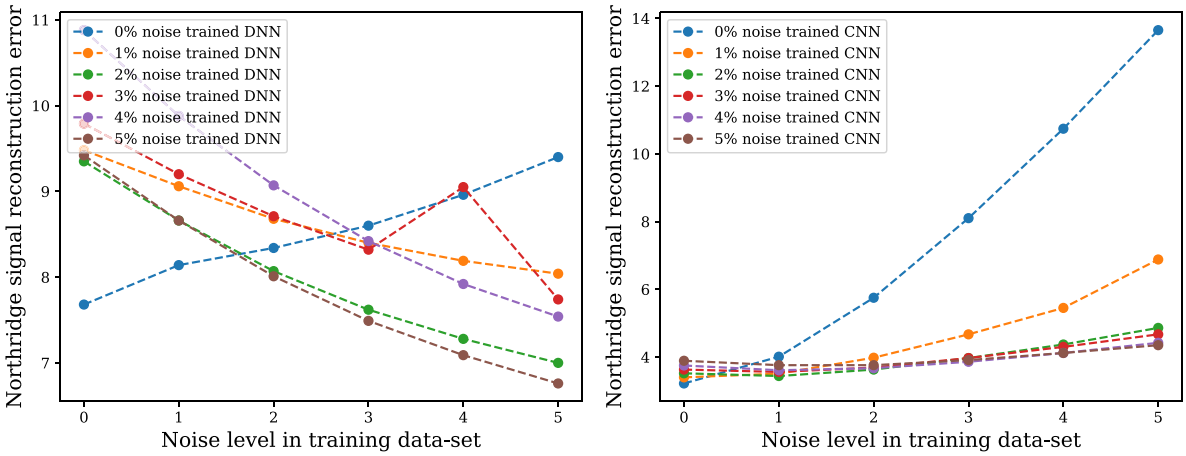
(a) DNN (b) CNN

Fig. 21. Example 2: The reconstruction of the 1994 Northridge earthquake incident waveform, using the DNN and CNN, for the 4-layer soil column.



(a) Noise analysis of a measured signal using DNN. (b) Noise analysis of a measured signal using CNN.

Fig. 22. Example 1: prediction error of the Northridge record, using DNN and CNN, and various noise levels, for the soil column with an asymptotically-varying wave speed.



(a) Noise analysis of a measured signal using DNN.

(b) Noise analysis of a measured signal using CNN.

Fig. 23. Example 2: prediction error of the Northridge record, using DNN and CNN, and various noise levels, for the 4-layer soil column.

Table 2
Prediction error of the Northridge record, using DNN and various noise levels, for the soil column with asymptotically-varying wave speed.

		Noise level in measured signal					
		0%	1%	2%	3%	4%	5%
Training noise	0%	9.07%	9.61%	9.90%	10.28%	10.76%	11.32%
	1%	10.05%	9.77%	9.55%	9.40%	9.31%	9.26%
	2%	10.13%	9.78%	9.54%	9.35%	9.23%	9.19%
	3%	10.77%	10.12%	9.59%	9.14%	8.78%	8.49%
	4%	11.65%	11.02%	10.48%	10.03%	9.63%	9.28%
	5%	11.86%	11.09%	10.43%	9.91%	9.46%	9.05%

Table 3
Prediction error of the Northridge record, using CNN and various noise levels, for the soil column with asymptotically-varying wave speed.

		Noise level in measured signal					
		0%	1%	2%	3%	4%	5%
Training noise	0%	3.81%	4.24%	5.83%	8.11%	10.85%	13.82%
	1%	3.67%	3.43%	3.58%	3.98%	4.54%	5.23%
	2%	3.74%	3.49%	3.53%	3.73%	4.05%	4.47%
	3%	4.40%	3.91%	3.74%	3.74%	3.82%	3.99%
	4%	4.20%	3.88%	3.75%	3.76%	3.84%	3.99%
	5%	4.11%	3.84%	3.82%	3.92%	4.10%	4.30%

Table 4
Prediction error of the Northridge record, using DNN and various noise levels, for a 4-layer soil column.

	Data set	Noise Level in a measured signal					
		0%	1%	2%	3%	4%	5%
Noise in training	0%	7.68%	8.14%	8.34%	8.60%	8.96%	9.40%
	1%	9.48%	9.06%	8.68%	8.40%	8.19%	8.04%
	2%	9.35%	8.66%	8.07%	7.62%	7.28%	7.00%
	3%	9.79%	9.20%	8.71%	8.32%	9.05%	7.74%
	4%	10.88%	9.88%	9.07%	8.42%	7.92%	7.54%
	5%	9.42%	8.66%	8.01%	7.49%	7.09%	6.76%

Table 5
Prediction error of the Northridge record, using CNN and various noise levels, for a 4-layer soil column.

	Data set	Noise Level in a measured signal					
		0%	1%	2%	3%	4%	5%
Noise in training	0%	3.22%	4.01%	5.75%	8.10%	10.74%	13.65%
	1%	3.40%	3.51%	3.98%	4.67%	5.45%	6.88%
	2%	3.52%	3.44%	3.63%	3.97%	4.37%	4.86%
	3%	3.63%	3.56%	3.69%	3.97%	4.30%	4.67%
	4%	3.75%	3.61%	3.67%	3.86%	4.12%	4.42%
	5%	3.89%	3.76%	3.76%	3.91%	4.12%	4.35%

DNN appears to be more tolerant to noise than CNN as the noise level in the training data set increases.

4.4. Performance of ANNs when actual seismic events are used to generate the training data set

In this example, we study the performance of our ANNs, when they are trained by using a training data set whose incident waveforms, $v^{\text{inc}}(t)$, are generated such that they resemble realistic seismic records. We compare the performance of the ANNs when trained in this manner, against the performance results reported in Sections 4.1 and 4.2, where the training data set did not resemble realistic seismic records.

To this end, we modified the two-dimensional waveform randomizer of Section 3.4, to use four different 18s-fragments of actually recorded earthquakes[5] as the first body force, to generate wave motion in the domain. This way, particle velocities, extracted from the domain, closely resembled realistic seismic records. To induce additional random reverberation, we modified the two-dimensional domain shown in Fig. 5 to have six layers, instead of four. Then, we applied eighteen body forces, consecutively, to generate wave motion in the domain. The first body force is the velocity field of one of the aforementioned actually recorded earthquakes, where its location in the domain is randomly chosen by the randomizer. Next, the remaining seventeen body forces, which have a Ricker wavelet temporal signature, are applied at various locations, with varying values for central frequency and peak amplitude. The time interval between each body force is 1 s, i.e., the first body force (the actual seismic record) is applied at $t = 0$ s, the second one (i.e., a Ricker wavelet) at $t = 1.0$ s, the third one at $t = 2.0$ s, and so on. Lastly, the particle velocity records are extracted from pre-defined locations. Algorithm 2 describes the details of the randomizer used in this section, and Fig. 24 shows an exemplary extracted particle velocity waveform that is used as incident waveform in a training data set.

[5] We used the 1979 Coyote Lake record, the 1922 Parkfield record, and two fragments of the 1995 Kobe record.

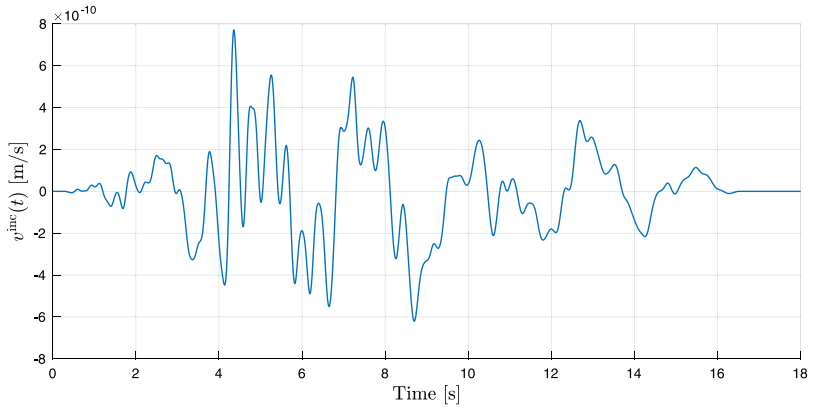


Fig. 24. A sample incident waveform in a training data set from Section 4.4.

Algorithm 2 Modified two-dimensional waveform randomizer to generate training, validation, and test data sets

1: Set the total number of iterations = 1000 and the number of body forces = 18.
2: **for** iteration ← 1 **to** (total number of iterations) **do**
3: Randomly set the values of shear wave speed (V_s) of each soil layer out of six layers and each inclusion out of 3 inclusions in the domain by using the following ranges:
$150 \leq V_{s_1}, V_{s_2}, V_{s_3}, V_{s_4}, V_{s_5}, V_{s_6} \leq 500$ m/s
$800 \leq V_{s_7}, V_{s_8}, V_{s_9} \leq 1200$ m/s.
4: Build mass, stiffness, and damping matrices.
5: **for** body force index ← 1 **to** (number of body forces) **do**
6: **if** body force index = 1 **then**
7: Realistic earthquake signal = Coyote Lake, Parfield, Kobe, or Kobe-2.
8: Horizontal and vertical coordinates of the location (s_x and s_y respectively) of the realistic earthquake signal are randomly chosen within the following ranges:
$5 \leq s_x \leq 30$ m,
$5 \leq s_y \leq 55$ m.
9: **else**
10: The amplitude (P), frequency (f), s_x, and s_y of each Ricker wavelet are randomly chosen within the following ranges:
-(50% of realistic earthquake signal's peak amplitude) $\leq P \leq$ (50% of realistic earthquake signal's peak amplitude),
$0.1 \leq f \leq 0.8$ Hz,
$5 \leq s_x \leq 30$ m,
$5 \leq s_y \leq 55$ m.
11: **end if**
12: (body force index) ← (body force index)+1.
13: **end for**
14: Solve the 2D wave propagation problem.
15: Extract the particle velocity signals from twelve different defined locations.
16: Use a low-pass filter to smooth each particle velocity signal.
17: Add zero-valued particle velocity at the last 150 time steps of each particle velocity signal.
18: Save each filtered particle velocity signal to be used as an incoming (incident) seismic signal, $v^{\text{inc}}(t)$, in the one-dimensional wave solver and as output-layer features in the presented DNN and CNN.
19: iteration ← iteration+1.
20: **end for**

As seen from Figs. 25 and 26, and Table 6, using realistic records leads to a slightly smaller error in DNN, but a slightly larger error in CNN.[6] While the prediction accuracy in the DNN has improved, and the prediction accuracy in the CNN has decreased, the CNN still outperforms DNN in predicting the Northridge seismic record.

Table 6
The comparison of accuracy of the ML prediction of the Northridge signal between the ANNs based on a training data set, where incident waveforms (*i*) do not resemble realistic earthquake signals – Sections 4.1 & 4.2 – versus (*ii*) do resemble them—Section 4.4.

Soil type	Sections 4.1 & 4.2		Section 4.4	
	DNN	CNN	DNN	CNN
Asymptotic	9.07%	3.81%	7.66%	4.68%
Multi-layered	7.68%	3.22%	6.45%	4.37%

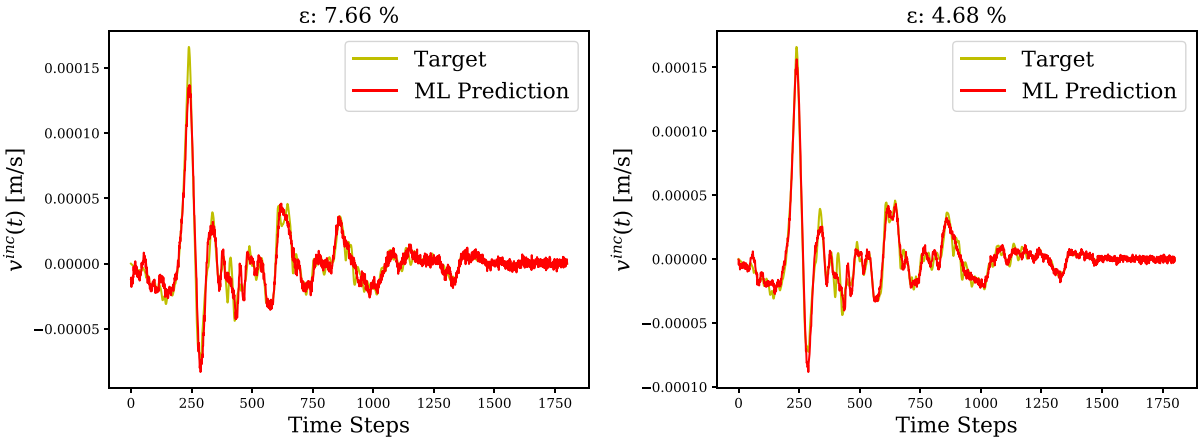
4.5. Impact of subsurface uncertainty on prediction accuracy

Site response analysis, as well as the data-informed framework we discussed herein, rely on the characterization of the site under investigation. That is, they take material properties (e.g., P- and S-wave velocities, attenuation properties, etc.) as known input values. In practice, it is often challenging to accurately profile a site, and our assumptions or knowledge of the subsurface may differ from reality. Quantifying subsurface uncertainty is an active area of research [48, 49]. Herein, we investigate the impact of subsurface uncertainty on the performance of our data-informed framework.

The *true profile* of a site is typically not known in practice, and site characterization techniques (e.g., FWI, SASW, or MASW) produce an *imaged profile* in their quest to find the true profile. Here, we consider six different exemplary true profiles that could possibly give rise to a 4-layered imaged profile[7] (Fig. 27). Shear wave velocity at each true profile is assigned randomly, at 5 m sub-layers, by using a Gaussian distribution. Namely, at each sub-layer, the mean of the Gaussian distribution is the value of the imaged profile at that sub-layer; we also consider three sets of standard deviations, corresponding to 15%, 20%, and 30% of the mean value, reflecting more uncertainty as standard deviation increases. The imaged profile is set to be the same as the aforementioned 4-layered soil profile, which has a depth of 100 m, and comprises four equal sub-layers with $V_{s_1} = 300$ m/s, $V_{s_2} = 250$ m/s, $V_{s_3} = 200$ m/s, and $V_{s_4} = 150$ m/s. We assume that the damping ratio and mass density of the true and imaged profiles are the same.

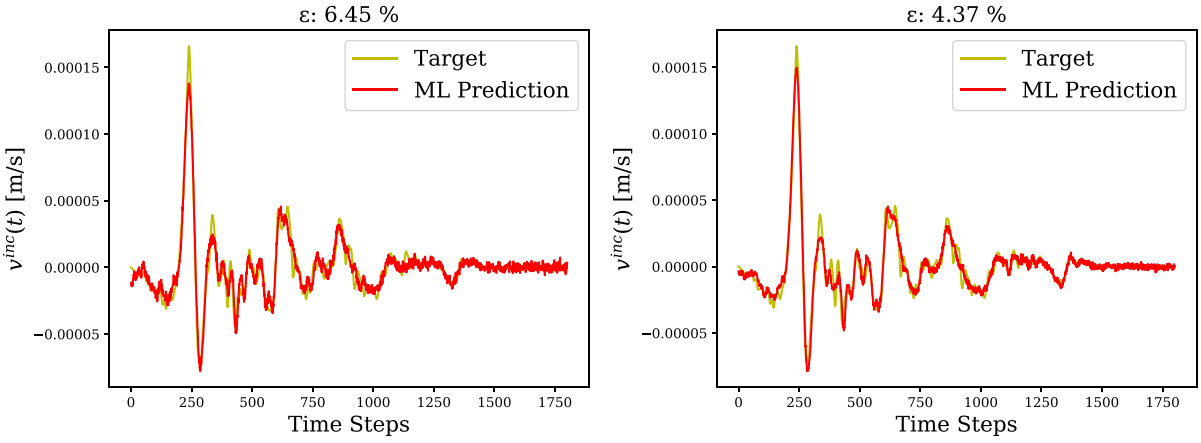
[6] Compared to results presented in Sections 4.1 and 4.2, where the incident waveforms in the training data set did not resemble realistic earthquake records.

[7] It may be more natural to consider one true profile, and six corresponding imaged profiles. However, as it becomes clear later, this choice would necessitate training the ANNs six times. Doing it the other way around limits the training process to one *imaged profile*, reducing our computational effort.



(a) DNN (b) CNN

Fig. 25. ML prediction of the Northridge record for the soil column with an asymptotically-varying shear waves peed, where training data involves actual seismic records.



(a) DNN (b) CNN

Fig. 26. ML prediction of the Northridge record for the 4-layered soil profile, where training data involves actual seismic records.

Next, we train[8] our ANNs by using the imaged profile, i.e., soil profile that deviates from reality. Then, we analyze the performance of our trained ANNs by using the 1994 Northridge record. To resemble a realistic scenario, the Northridge record is propagated through each of the true profiles, using DEEPSOIL. Fig. 28 shows the ANNs that are trained by using the imaged profile are able to predict the Northridge record satisfactorily, with a maximum error of 11.0%. The errors corresponding to these uncertain soil profiles – a reflection of reality – are slightly higher than those reported in Sections 4.2 and 4.4 and are summarized in Tables 7 and 8. Overall, it appears that when there is a mismatch between the true and imaged profiles due to subsurface uncertainty, the ANNs can still produce satisfactory results.

5. Computational cost of the presented method

We compare the computational cost of our data-informed method with that of the gradient-based method discussed by Jeong and Seylabi [41]. In general, providing an accurate comparison is difficult due to using different hardware and software for different parts, where some of the tools or computer runs may have benefited from a more optimized code or faster hardware than others. Nevertheless, our comparison still provides a general idea about the computational cost of each approach.

Data generation for training the neural networks was performed on a CPU, and took us about 3 h, where the random waveform generator (Section 3.4.1) took about 2 h, and using these random waveforms of $v^{\text{inc}}(t)$ in our one-dimensional wave simulator to compute the corresponding time histories at the ground surface of $u(L,t)$ (Section 3.4.2) took about 1 h. We note that the former data of $v^{\text{inc}}(t)$ can be reused for other sites and situations, whereas the latter part of $u(L,t)$ is unique to each site based on its own soil profile. We also note that generating each record is independent from other records, and, therefore, data generation can be parallelized. Once the training data are available, the training itself takes about 16 minutes for the DNN architecture and about 10 minutes for the CNN architecture on a GPU. The inference part takes only a fraction of a second. We note that the data generation and training process can be done before a real seismic event (e.g., Northridge) is recorded. With the advantage of pre-earthquake training, once the recording of the ground motion is completed, the inference task (i.e., running the trained DNN or CNN) can be done almost instantaneously.

On the other hand, the technique, discussed by Jeong and Seylabi [41], uses the apparatus of PDE-constrained optimization to compute the incoming seismic waveform $v^{\text{inc}}(t)$. For problems similar to those

[8] Indeed, the ANNs have already been trained for the 4-layered soil profile by using training data from Section 4.2 (realistic earthquake records excluded from the data-generation process) and Section 4.4 (realistic earthquake records included in the data-generation process).

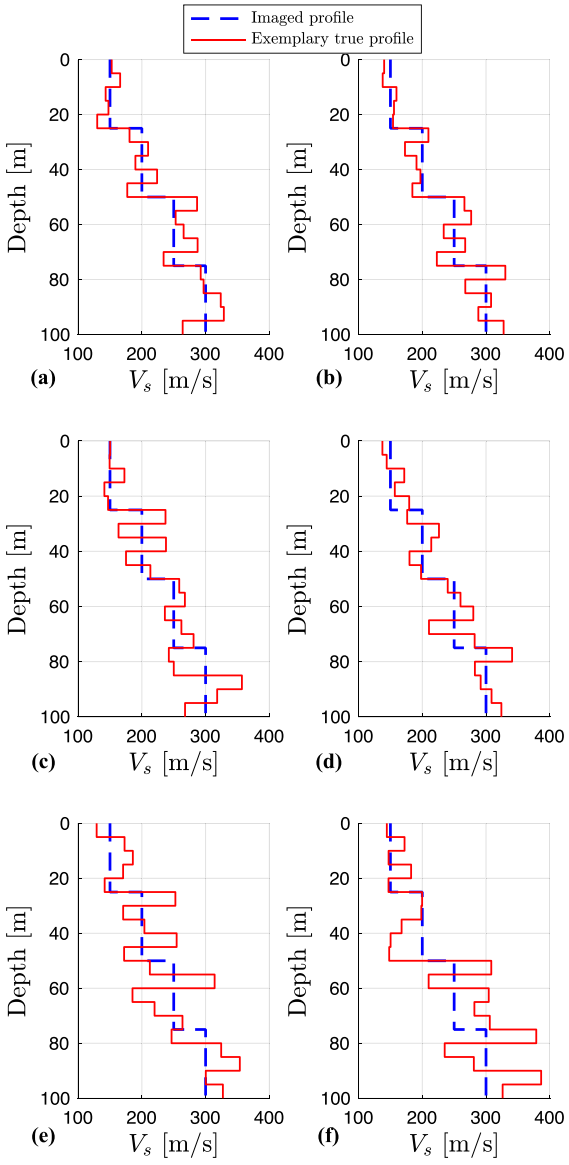


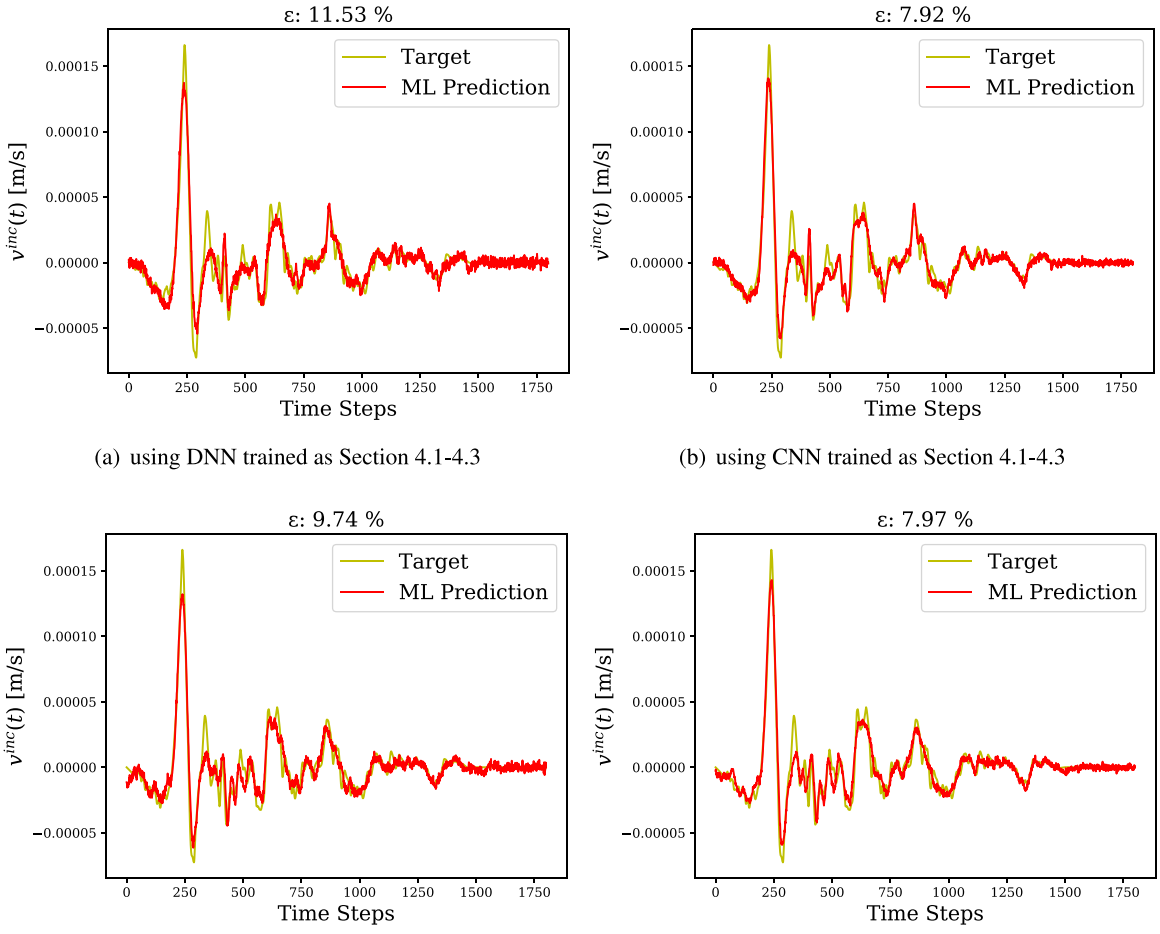
Fig. 27. Six different exemplary *true* soil profiles with: (a,b) 15%, (c,d) 20%, and (e,f) 30% deviation from the 4-layered *imaged* profile. To investigate the impact of subsurface uncertainty, ANNs are trained by using the imaged profile and tested by using the true profiles.

Table 7
Comparison of the ML prediction errors for the Northridge record when subsurface uncertainty is present. ANNs were trained using the 4-layered imaged profile, and tested via different true profiles. Training data are purely synthetic.

Considered soil profile for ANN performance evaluation	DNN	CNN
4-layered imaged profile (results are shown in Section 4.2, Fig. 21)	7.68%	3.22%
True profile a: 15% deviation from the imaged profile	7.21%	5.92%
True profile b: 15% deviation from the imaged profile	7.64%	5.13%
True profile c: 20% deviation from the imaged profile	7.88%	5.63%
True profile d: 20% deviation from the imaged profile	9.42%	6.82%
True profile e: 30% deviation from the imaged profile (Figs. 28(a) and 28(b))	11.53%	7.92%
True profile f: 30% deviation from the imaged profile	11.69%	8.71%

considered in this study, the approach based on PDE-constrained optimization takes several hours to predict the incoming wave motion on a CPU once the recording of the ground motion is finished. Thus, in urgent situations, where $v^{\text{inc}}(t)$ should be inferred immediately after an earthquake, the presented DNN or CNN may be preferable due to its low computational cost.

For the one-dimensional problems we considered here, it appears that our data-informed framework is considerably faster when compared to techniques that rely on PDE-constrained optimization. We speculate this remains true for multi-dimensional incident seismic-wave inversion problems as well. We plan to investigate this hypothesis in the future.



(a) using DNN trained as Section 4.1-4.3

(b) using CNN trained as Section 4.1-4.3

(c) using DNN trained as Section 4.4

(d) using CNN trained as Section 4.4

Fig. 28. Northridge record prediction for the true soil profile in Fig. 27(e) with 30% deviation from the 4-layered imaged profile.

Table 8
Comparison of the ML prediction errors for the Northridge record when subsurface uncertainty is present. ANNs were trained using the 4-layered imaged profile, and tested via different true profiles. $v^{inc}(t)$ in Training data mimic actual recorded seismic events.

Considered soil profiles for ANN performance evaluation	DNN	CNN
4-layered imaged profile (results are shown in Section 4.4: Fig. 26)	6.45%	4.37%
True profile a: 15% deviation from the imaged profile	7.47%	5.85%
True profile b: 15% deviation from the imaged profile	6.70%	4.72%
True profile c: 20% deviation from the imaged profile	7.06%	5.37%
True profile d: 20% deviation from the imaged profile	8.90%	6.02%
True profile e: 30% deviation from the imaged profile (Figs. 28(c) and 28(d))	9.74%	7.97%
True profile f: 30% deviation from the imaged profile	9.84%	8.52%

6. Extension to three dimensions

The presented methodology can be extended to three-dimensional problems. The main enablers include: (a) a three-dimensional wave simulator that likely uses parallel algorithms on modern computing architectures [50–52]; and (b) the domain reduction method (DRM) that brings incoherent waves of any incident angles as equivalent effective seismic body forces at a DRM layer [53]. One could then reformulate the presented ANN framework such that the dynamic force at each discrete node on a DRM layer, within a truncated domain, will be identified based on measured seismic data. The extension of the presented ANN framework to three-dimensional models, powered by the DRM, would likely allow users of the method to accurately identify unknown incoming seismic waves, which are modeled as equivalent effective seismic body forces at a DRM layer, from measured seismic data, likely in real-time. Such a potential future work will need a cluster with a number of GPU processors of high memory capacity because coefficient (e.g., weight) matrices will likely be dense and larger than those considered in the presented one-dimensional study.

7. Conclusions

We demonstrated the effectiveness and robustness of using a data-informed framework for identifying the incoming seismic wave motion on bedrock under a soil column using ground surface measurements. The underlying soil profile is assumed to be one-dimensional, permitting perfectly vertical plane shear waves. We considered both deep and convolutional neural networks and showed their ability to learn important feature relationships between the input-layer data (measured surface displacements) and the output-layer data (velocity history of the incoming wave motion) for the seismic-input-inversion problem. We demonstrated the robustness of our framework by using a blind test

example (the Northridge record) and noisy measurements, where input- and output-layer data differed considerably from those considered in the training data set. We considered two different training sets: the first set contained purely synthetic records (Sections 4.1 to 4.3), whereas the second set involved records of actual seismic events (Section 4.4). We evaluated the performance of our ANNs in both situations. We also investigated the impact of subsurface uncertainty on the performance of our ANNs, where the ANNs were trained by using a 4-layered *imaged* profile that deviated from a corresponding 16-layered *true* profile. The true profile was used for evaluating the performance of the ANNs, and the results indicated satisfactory performance. While we only considered one-dimensional soil profiles in this study, our approach is quite general and can be extended to two- and three-dimensional problems.

Multi-dimensional inverse-source problems require gradient-based methods, where they rely on repeated solutions of a state, adjoint, and control problem [23]. While these techniques are accurate and robust, they are computationally expensive even for one-dimensional problems. We demonstrated that our data-informed method is considerably fast, and we expect to see similar performance for multi-dimensional problems. Based on the successful performance of the data-informed method in the one-dimensional cases we considered in this study, we plan to investigate the extension of our approach to multi-dimensional settings.

Nomenclature

Symbol	Comment
x, y, z	Horizontal, vertical, and anti-plane directions, respectively
t	Time
T	Final time; total observation duration
L	The x-coordinate of the top surface of the soil column
$\mathbf{K}, \mathbf{C}, \mathbf{M}$	Global stiffness, damping, and mass matrices
$\mathbf{F}$	Global force vector
$u(x,t)$	Displacement field of the wave motion
u^{inc}	Displacement time-history of an incident wave
$v^{\text{inc}}(t)$	Incoming seismic wave
μ, μ_B	Shear modulus of a soil layer, and the bedrock
V_s, V_B	Shear wave speed of a soil layer, and the bedrock
ρ, ρ_B	Mass density of a soil layer, and the bedrock
$\mathbf{C}_{\text{R}}$	Rayleigh damping matrix
ξ	Damping ratio of a soil layer
f_i, f_j	Control frequencies for the Rayleigh damping
η	Mode number
$\bar{V}_s$	Average shear wave speed of the soil column
w	Weight coefficient between two successive neurons at neighboring layers
q	Feature (or neuron) from a previous layer
d	Size of the data from a previous layer
b	Bias associated with each neuron
a	Outcome of a specific ANN layer after application of weights and bias
i	The ith sample; the ith neuron
j	The jth time step
f_{LReLU}	LReLu activation function
r	The outcome from the application of LReLU activation function
n	Number of samples of data
m	Number of time steps of data
$\hat{v}^{\text{inc}}$	Counterpart of the ANN-predicted output-layer data
MSE	Mean-squared error
ϵ	Error function between a target seismic wave and its counterpart
f, P	Frequency and amplitude of a Ricker wavelet signal
s_x, s_y	Horizontal and vertical coordinates of the location of source in a 2D domain
A, A^n	Non-normalized and normalized components of the data matrix
$A^{\text{mean}}_{\text{train}}$	Mean value of the training set of the data matrix
$A^{\text{max}}_{\text{train}}$, $A^{\text{min}}_{\text{train}}$	Maximum, and minimum values of the training set of the data matrix.
u^{noised}	Noisy displacement signal
u^{max}	Maximum value of $u(L,t)$ on a soil column
B	Array of normal-distributed random numbers ranged (−1,1)
δ	Noise level

CRediT authorship contribution statement

Shashwat Maharjan: Formal analysis, Investigation, Methodology, Software (development), Validation, Visualization, Writing – original draft, Writing – review & editing. **Bruno Guidio:** Formal analysis, Investigation, Methodology, Software (development), Validation, Visualization, Writing – original draft, Writing – review & editing. **Arash Fathi:** Conceptualization, Investigation, Methodology, Writing – original draft, Writing – review & editing. **Chanseok Jeong:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software (development), Supervision, Validation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This material is based upon work supported by the National Science Foundation, USA, under Award CMMI-2044887 and CMMI-2053694. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors are also grateful for the support by the Faculty Research and Creative Endeavors (FRCE) Research, USA Grant-48058 at Central Michigan University. The authors would also like to thank the Office of Research and Graduate Studies (ORGS) at Central Michigan University for supporting this research through Undergraduate Summer Scholars Program 2021. We are also grateful to the reviewers for their constructive comments.

Appendix. Verification of the one-dimensional wave simulator

Prior to the data generation for training, validating, and testing the presented ANNs, we verify our one-dimensional wave simulator, written in MATLAB, by comparing our wave response on the top surface of a soil column with the reference solution obtained by using the DEEPSOIL software, developed by Hashash et al. [54]. This verification considers the 4-layered soil column and the underlying bedrock described in Section 3.4.2. In addition, the velocity time-history of the 1979 Coyote Lake earthquake is used as the incident wave, $v^{\text{inc}}(t)$, in the soil column. In our one-dimensional setting, the soil column is discretized by using an element size of 0.01 m, while in the DEEPSOIL simulation, each one of the four layers is divided into 25 sub-layers of 1 m in length. The time-step is 0.01 s. Fig. A.29 shows excellent agreement between the acceleration field of the wave response from

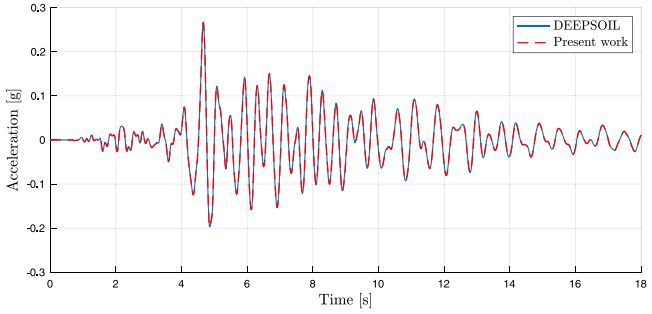


Fig. A.29. Comparison of acceleration time-histories at the ground surface: DEEPSOIL (reference solution) vs. one-dimensional wave simulator used in this study.

our one-dimensional wave simulator and that corresponding to the reference solution, both on the top surface of the soil column. This comparison provides reasonable confidence in the correctness of our wave simulator to generate data samples for our ANNs.

References

[1] Poursartip B, Fathi A, Tassoulas JL. Large-scale simulation of seismic wave motion: A review. Soil Dyn Earthq Eng 2020;129:105909.

[2] Roesset JM, Chang D-W, Stokoe I, Kenneth H, Aouad M. Modulus and thickness of the pavement surface layer from SASW tests. Transp Res Rec 1990;(1260).

[3] Luke BA, Stokoe II KH. Application of SASW method underwater. J Geotech Geoenviron Eng 1998;124(6):523–31. http://dx.doi.org/10.1061/(ASCE)1090-0241(1998)124:6(523).

[4] Brown LT, Boore DM, Stokoe KH. Comparison of shear-wave slowness profiles at 10 strong-motion sites from noninvasive SASW measurements and measurements made in boreholes. Bull Seismol Soc Am 2002;92(8):3116–33.

[5] Park CB, Miller RD, Xia J. Multichannel analysis of surface waves. Geophysics 1999;64(3):800–8.

[6] Rahimi S, Wood CM, Himel AK. Application of microtremor horizontal to vertical spectra ratio (MHVSR) and multichannel analysis of surface wave (MASW) for shallow bedrock mapping for transportation projects. In: Geo-congress 2020: Modeling, geomaterials, and site characterization. American Society of Civil Engineers Reston, VA; 2020, p. 622–32.

[7] Cox BR, Teague DP. Layering ratios: a systematic approach to the inversion of surface wave data in the absence of a priori information. Geophys J Int 2016;207(1):422–38.

[8] Teague DP, Cox BR, Rathje EM. Measured vs. predicted site response at the Garner Valley Downhole Array considering shear wave velocity uncertainty from borehole and surface wave methods. Soil Dyn Earthq Eng 2018;113:339–55.

[9] Kallivokas L, Fathi A, Kucukcoban S, Stokoe K, Bielak J, Ghattas O. Site characterization using full waveform inversion. Soil Dyn Earthq Eng 2013;47:62–82, SI: José Manuel Rosset.

[10] Fathi A, Kallivokas LF, Poursartip B. Full-waveform inversion in three-dimensional PML-truncated elastic media. Comput Methods Appl Mech Engrg 2015;296:39–72.

[11] Fathi A, Poursartip B, Stokoe II KH, Kallivokas LF. Three-dimensional P- and S-wave velocity profiling of geotechnical sites using full-waveform inversion driven by field data. Soil Dyn Earthq Eng 2016;87:63–81.

[12] Askan A, Akcelik V, Bielak J, Ghattas O. Full waveform inversion for seismic velocity and anelastic losses in heterogeneous structures. Bull Seismol Soc Am 2007;97(6):1990–2008. http://dx.doi.org/10.1785/0120070079, arXiv:https://pubs.geoscienceworld.org/bssa/article-pdf/97/6/1990/3666603/1990.pdf.

[13] Askan A, Akcelik V, Bielak J, Ghattas O. Parameter sensitivity analysis of a nonlinear least-squares optimization-based anelastic full waveform inversion method. C R Méc 2010;338(7):364–76. http://dx.doi.org/10.1016/j.crme.2010.07.002, URL: http://www.sciencedirect.com/science/article/pii/S1631072110000963, Inverse problems.

[14] Poursartip B, Fathi A, Kallivokas LF. Seismic wave amplification by topographic features: A parametric study. Soil Dyn Earthq Eng 2017;92:503–27.

[15] Poursartip B, Kallivokas LF. Model dimensionality effects on the amplification of seismic waves. Soil Dyn Earthq Eng 2018;113:572–92.

[16] Asimaki D, Mohammadi K. On the complexity of seismic waves trapped in irregular topographies. Soil Dyn Earthq Eng 2018;114:424–37.

[17] Ayoubi P, Mohammadi K, Asimaki D. A systematic analysis of basin effects on surface ground motion. Soil Dyn Earthq Eng 2021;141:106490.

[18] Mejia L, Dawson E. Earthquake deconvolution for FLAC. FLAC Numer 2006.

[19] Ju SH. A Deconvolution scheme for determination of seismic loads in Finite Element Analyses. Bull Seismol Soc Am 2013;103(1):258–67. http://dx.doi.org/10.1785/0120120034, URL: http://www.bssaonline.org/content/103/1/258.

[20] Poul MK, Zerva A. Efficient time-domain deconvolution of seismic ground motions using the equivalent-linear method for soil-structure interaction analyses. Soil Dyn Earthq Eng 2018;112:138–51.

[21] Poul MK, Zerva A. Nonlinear dynamic response of concrete gravity dams considering the deconvolution process. Soil Dyn Earthq Eng 2018;109:324–38. http://dx.doi.org/10.1016/j.soildyn.2018.03.025, URL: http://www.sciencedirect.com/science/article/pii/S0267726117308825.

[22] Guidio B, Jeong C. Identification of seismic ground motions in a near-surface 2D domain subject to unknown SH incident waves. In: Geo-Congress 2022. p. 599–607. http://dx.doi.org/10.1061/9780784484043.058.

[23] Guidio B, Jeremić B, Guidio L, Jeong C. Passive seismic inversion of SH wave input motions in a truncated domain. Soil Dyn Earthq Eng 2022;158:107263. http://dx.doi.org/10.1016/j.soildyn.2022.107263.

[24] Bielak J, Loukakis K, Hisada Y, Yoshimura C. Domain reduction method for three-dimensional earthquake modeling in localized regions, Part I: Theory. Bull Seismol Soc Am 2003;93(2):817–24.

[25] Guidio BP, Jeong C. Full-waveform inversion of incoherent dynamic traction in a bounded 2D domain of scalar wave motions. J Eng Mech 2021;147(4):04021010. http://dx.doi.org/10.1061/(ASCE)EM.1943-7889.0001909.

[26] Wang H, Alkhalifah T. Microseismic imaging using a source function independent full waveform inversion method. Geophys J Int 2018;214(1):46–57.

[27] Song C, Alkhalifah T. Microseismic event estimation based on an efficient wavefield inversion. IEEE J Sel Top Appl Earth Obs Remote Sens 2019;12(11):4664–71.

[28] Röth G, Tarantola A. Neural networks and inversion of seismic data. J Geophys Res: Solid Earth 1994;99(B4):6753–68.

[29] Vantassel JP, Kumar K, Cox BR. Using convolutional neural networks to develop starting models for 2D full waveform inversion. 2021, arXiv:2104.01626.

[30] Iturrarán-Viveros U, Muñoz-García AM, Castillo-Reyes O, Shukla K. Machine learning as a seismic prior velocity model building method for full-waveform inversion: a case study from Colombia. Pure Appl Geophys 2021;178(2):423–48.

[31] Kazei V, Ovcharenko O, Plotnitskii P, Peter D, Zhang X, Alkhalifah T. Mapping full seismic waveforms to vertical velocity profiles by deep learning. Geophysics 2021;86(5):R711–21.

[32] Wu Y, Lin Y. InversionNet: An efficient and accurate data-driven full waveform inversion. IEEE Trans Comput Imaging 2019;6:419–33.

[33] Ren Y, Xu X, Yang S, Nie L, Chen Y. A physics-based neural-network way to perform seismic full waveform inversion. IEEE Access 2020;8:112266–77.

[34] Song C, Alkhalifah T. Wavefield reconstruction inversion via physics-informed neural networks. IEEE Trans Geosci Remote Sens 2022;60:1–12.

[35] Yang F, Ma J. Deep-learning inversion: A next-generation seismic velocity model building method. Geophysics 2019;84(4):R583–99.

[36] Haile MA, Zhu E, Hsu C, Bradley N. Deep machine learning for detection of acoustic wave reflections. Struct Health Monit 2020;19(5):1340–50.

[37] Li Z, Meier M-A, Hauksson E, Zhan Z, Andrews J. Machine learning seismic wave discrimination: Application to earthquake early warning. Geophys Res Lett 2018;45(10):4773–9.

[38] Wang H, Alkhalifah T. Direct microseismic event location and characterization from passive seismic data using convolutional neural networks. Geophysics 2021;86(6):KS109–21.

[39] Wang H, Alkhalifah T, Waheed Ub, Birnie C. Data-driven microseismic event localization: An application to the oklahoma arkoma basin hydraulic fracturing data. IEEE Trans Geosci Remote Sens 2022;60:1–12.

[40] Zheng Y, Zhang Q, Yusifov A, Shi Y. Applications of supervised deep learning for seismic interpretation and inversion. Lead Edge 2019;38(7):526–33.

[41] Jeong C, Seylabi EE. Seismic input motion identification in a heterogeneous halfspace. J Eng Mech 2018;144(8):04018070. http://dx.doi.org/10.1061/(ASCE)EM.1943-7889.0001495.

[42] Phillips C, Hashash YM. Damping formulation for nonlinear 1D site response analyses. Soil Dyn Earthq Eng 2009;29(7):1143–58.

[43] Zhang W, Yang G, Lin Y, Ji C, Gupta MM. On definition of deep learning. In: 2018 world automation congress (WAC). IEEE; 2018, p. 1–5.

[44] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings; 2010, p. 249–56.

[45] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision, 2015, p. 1026–34.

[46] Tan HH, Lim KH. Vanishing gradient mitigation with deep learning neural network optimization. In: 2019 7th international conference on smart computing communications (ICSCC). 2019, p. 1–4. http://dx.doi.org/10.1109/ICSCC.2019.8843652.

[47] Chollet F. Deep learning with python. Manning Publications Co.; 2018.

[48] Zhu H, Li S, Fomel S, Stadler G, Ghattas O. A Bayesian approach to estimate uncertainty for full-waveform inversion using a priori information from depth migration. Geophysics 2016;81(5):R307–23.

[49] Tewari A, Wheelock B, Paiva A, Fathi A, Cheon M-S. Towards practical Bayesian inversion of geobodies using geologic priors. In: SEG/AAPG/SEPM first international meeting for applied geoscience & energy. SEG international exposition and annual meeting, 2021, p. 1641–5.

[50] Komatitsch D, Tromp J. Spectral-element simulations of global seismic wave propagation—I. validation. Geophys J Int 2002;149(2):390–412.

[51] Hanindhito B, Gourounas D, Fathi A, Trenev D, Gerstlauer A, John LK. GAPS: GPU-acceleration of PDE solvers for wave simulation. In: Proceedings of the 36th ACM international conference on supercomputing. ICS '22, New York, NY, USA: Association for Computing Machinery; 2022, p. 1–13.

[52] Hanindhito B, Li R, Gourounas D, Fathi A, Govil K, Trenev D, Gerstlauer A, John L. Wave-PIM: Accelerating wave simulation using processing-in-memory. In: 50th international conference on parallel processing. ICPP 2021, New York, NY, USA: Association for Computing Machinery; 2021, p. 1–11.

[53] Yoshimura C, Bielak J, Hisada Y, Fernandez A. Domain reduction method for three-dimensional earthquake modeling in localized regions, part II: Verification and applications. Bull Seismol Soc Am 2003;93(2):825–41.

[54] Hashash YMA, Musgrove MI, Harmon JA, Ilhan O, Xing G, Numanoglu O, Groholski D, Phillips CA, Park D. DEEPSOIL V7.0, user manual. Technical report, Urbana, IL: Board of Trustees of University of Illinois at Urbana-Champaign; 2020.