Convolutional-neural-network-based reflection-waveform inversion

Yulang Wu[1], George McMechan[2], and Yanfei Wang[1]

[1]Institute of Geology and Geophysics, Chinese Academy of Sciences
[2]The University of Texas at Dallas

November 22, 2022

Abstract

\justifying Full-waveform inversion (FWI) is a non-linear optimization algorithm to estimate the velocity model by fitting the observed seismic data. With a smooth starting velocity model, FWI mainly inverts for the shallower background velocity model by fitting the observed direct, diving and refracted data, and updates the interfaces by fitting the observed reflected data. As the deeper background velocity model cannot be effectively updated by fitting the reflected data in FWI, the deeper interfaces are less accurate than the shallower interfaces. To update the deeper background velocity model, many reflection-waveform inversion (RWI) algorithms were proposed to separate the tomographic and migration components from the reflection-related gradient. We propose a convolutional-neural-network-based reflection-waveform inversion (CNN-RWI) to repeatedly apply the iteratively-updated CNN to predict the true velocity model from the smooth starting velocity model (the tomographic components), and the high-resolution migration image (the migration components). The CNN is iteratively updated based on the more representative training dataset, which is obtained from the latest CNN-predicted velocity model by the proposed spatially-constrained divisive hierarchical k-means parcellation method. The more representative training velocity models are, the more accurate CNN-predicted velocity model. Synthetic examples using different portions of the Marmousi2 P-wave velocity model show that CNN-RWI inverts for both the shallower and deeper velocity model more accurately than the conjugate-gradient FWI (CG-FWI) does. Both the CNN-RWI and the CG-FWI are sensitive to the accuracy of the starting velocity model and the complexity of the unknown true velocity model.

Convolutional-neural-network-based reflection-waveform inversion

Yulang Wu[1,2], George A. McMechan[3], and Yanfei Wang[1,2,4]

[1]Key Laboratory of Petroleum Resources Research, Institute of Geology and Geophysics, Chinese Academy of Sciences, Beijing 100029, China
[2]Innovation Academy for Earth Science, Chinese Academy of Sciences, Beijing 100029, China
[3]The University of Texas at Dallas, Richardson, TX 75080-3021, USA
[4]University of Chinese Academy of Sciences, Beijing 100049, China

Key Points:

- The proposed method adaptively shifts distributions of the training and predicted velocity models towards the unknown true velocity model.
- The proposed method does less overfitting to predict a more accurate velocity model by reducing the selection bias in the training dataset.

Corresponding author: Yanfei Wang, yfwang@mail.iggcas.ac.cn

Abstract

Full-waveform inversion (FWI) is a non-linear optimization algorithm to estimate the velocity model by fitting the observed seismic data. With a smooth starting velocity model, FWI mainly inverts for the shallower background velocity model by fitting the observed direct, diving and refracted data, and updates the interfaces by fitting the observed reflected data. As the deeper background velocity model cannot be effectively updated by fitting the reflected data in FWI, the deeper interfaces are less accurate than the shallower interfaces. To update the deeper background velocity model, many reflection-waveform inversion (RWI) algorithms were proposed to separate the tomographic and migration components from the reflection-related gradient. We propose a convolutional-neural-network-based reflection-waveform inversion (CNN-RWI) to repeatedly apply the iteratively-updated CNN to predict the true velocity model from the smooth starting velocity model (the tomographic components), and the high-resolution migration image (the migration components). The CNN is iteratively updated based on the more representative training dataset, which is obtained from the latest CNN-predicted velocity model by the proposed spatially-constrained divisive hierarchical k-means parcellation method. The more representative training velocity models are, the more accurate CNN-predicted velocity model. Synthetic examples using different portions of the Marmousi2 P-wave velocity model show that CNN-RWI inverts for both the shallower and deeper velocity model more accurately than the conjugate-gradient FWI (CG-FWI) does. Both the CNN-RWI and the CG-FWI are sensitive to the accuracy of the starting velocity model and the complexity of the unknown true velocity model.

Plain Language Summary

Full waveform inversion (FWI) is generally a combination of tomography and migration techniques. However, the conventional FWI cannot estimate the deeper background velocity model accurately. By separating the tomographic and migration components, reflection-waveform inversion (RWI) effectively utilizes the tomographic component to update the deeper background velocity model. We propose a convolutional-neural-network-based reflection-waveform inversion (CNN-RWI) to combine the smooth starting velocity model (the tomographic component) and the corresponding migration image (the migration component), to accurately predict both the background velocity and interfaces in the velocity model. The conventional CNN application predicts the velocity model, by following the training dataset preparation, CNN training, and the CNN prediction, once only. In contrast, the CNN-RWI contains an innovative outer loop to iteratively update the CNN to predict the more accurate velocity model from the original starting velocity model and migration image, by dynamically recreating the more representative training velocity models for the CNN training. We also propose a novel spatially-constrained divisive hierarchical k-means parcellation method to obtain the more representative velocity models by parcellating the latest CNN-predicted velocity model into a model basis.

1 Introduction

Full waveform inversion (FWI) (Bamberger et al., 1982; Lailly, 1983; Tarantola, 1984; Gauthier et al., 1986; Mora, 1987; Crase et al., 1990; Pratt et al., 1998; Pratt, 1999) is a powerful algorithm to perform a least-squares non-linear data-fitting optimization to estimate a high-resolution velocity model. The conventional FWI gradient, obtained by zero-lag cross-correlating both the source and residual wavefields, can be divided into three components (Xu et al., 2012; Alkhalifah, 2014; Wu & Alkhalifah, 2015; Yao et al., 2020): (1) the low-wavenumber component obtained from the direct, refracted, and diving waves, (2) the low-wavenumber (tomographic) component and (3) the high-wavenumber (migration) component obtained from reflected waves. Because of the shallower penetration depth of the direct, refracted, and diving waves (relative to the reflected waves) and also because of the overlapping of their wavepaths in both the source and residual wavefields, this compo-

nent of the gradient can effectively update mainly the shallower background velocity model, provided that the traveltime errors are within a half cycle (Virieux & Operto, 2009). As the initial model is normally smooth, at the first iteration of FWI, the source wavefield only meets the reflection-related residual wavefield at the interfaces. Thus, at the first iteration, reflected waves provide mainly the high-wavenumber component of the FWI gradient at the interface (migration component) (Mora, 1989; Yao et al., 2020). In the following iterations, both the source and reflection-related residual wavefields create corresponding scattered waves at the interfaces. Then, the low-wavenumber (tomographic) component of the FWI gradient is obtained from the reflected waves, because the wavepaths of the source and reflection-related residual wavefields overlap with the scattered waves, which are created by each other at the interfaces. The reflection-related high-wavenumber (migration) component at the interface is generally one order of magnitude higher than the reflection-related low-wavenumber (tomographic) component above the interfaces (Yao et al., 2020). Therefore, with a smooth starting velocity model, conventional FWI mainly updates the deeper interfaces, rather than the deeper background velocity model, by fitting the observed reflection data.

To effectively update the background velocity model by tomographic component, there are mainly three types of reflection-waveform inversion (RWI) proposed to retain the tomographic component but remove the migration component from the gradient. The first strategy is to separate the up- and down-going wavefield based on the assumption that the incident and reflection wavefields propagates downward and upward, respectively (Hu & McMechan, 1987; Liu et al., 2011; Wang et al., 2013; Fei et al., 2015; Chi et al., 2017; Lian et al., 2018). However, if a layer has a large dipping angle, the migration component cannot be removed completely and so causes the background velocity model to be updated in the wrong direction. To separate the incident and reflection wavefields in both the vertical and horizontal direction, Irabor and Warner (2016) propose to use a pair of 1D two-way wave equations to separately simulate the vertical and horizontal wave propagations. The second strategy is to apply scattering angle filtering, based on the assumption that the tomographic component corresponds to the scattering angle close to 180^{o} (Khalil et al., 2014; Alkhalifah, 2014; Wu & Alkhalifah, 2015; Wu & Alkhalifah, 2017; Yao et al., 2018, 2019). The third strategy is to apply Born modeling to separate tomographic and migration components (Xu et al., 2012; Wu & Alkhalifah, 2017; Sun et al., 2017; Yao & Wu, 2017). Ma et al. (2012) propose to utilize image-guided interpolation and its adjoint operator to get a sparse model from the migration image and to constrain the inversion to the blocky model, interpolated from the sparse model. Wang et al. (2021) propose a generalized internal multiple imaging-based RWI (GIMI-RWI) to convolve the data residuals with the reflection kernel stored for each source-receiver pair to update the tomographic velocity. Thus, GIMI-RWI avoids the Born modeling for demigration and is also source independent to update the velocity along the wavepaths. We refer Yao et al. (2020) for an overview of the reflection-waveform inversion.

The convolutional neural network (CNN) is first developed in the computer vision field for image classification (Fukushima & Miyake, 1982; LeCun et al., 1989, 1990, 1998; Krizhevsky et al., 2012). The success of CNN in computer vision is partially attributed to many open-access large datasets, such as the ImageNet dataset (Deng et al., 2009) and the Microsoft COCO dataset (Lin et al., 2014), both of which contain thousands of randomly-selected images with labels. The large image datasets are then divided into training, test, and validation datasets for CNN application. Because of the success of the CNN in computer vision, it became a popular tool in seismic inversion, to predict the velocity model directly from a raw seismic data or a migration image (Araya-Polo et al., 2018; Lin & Wu, 2018; Yang & Ma, 2019; Wang & Ma, 2020; Liu et al., 2021; Zhang & Gao, 2021). These CNN applications to predict the velocity model replace the conventional wave-equation-guided procedures (e.g., calculation of the model gradient in FWI and RWI to fit the seismic data), by training CNN to approximate the mapping relations in the training dataset. However, unlike the image datasets which have a relatively small variety of types, shapes and profiles

of targets (e.g., human, animals, objects, etc.), the earth contains realistic, complex geologic models with numerous, random, shapes of various layers, salts, and faults. What is worse, most of the geologic models are either unknown or not open-access. Therefore, many CNN applications to predict velocity models rely on different model generation methods to create random velocity models for CNN training, validation, and testing. Araya-Polo et al. (2018) generate velocity models with random faults and salt bodies. Lin and Wu (2018), and Zhang and Gao (2021) generate velocity models with random flat subsurface layers, and curved subsurface layers with faults. Yang and Ma (2019) generate random velocity models with smooth interface curvatures and increasing velocity values with depth. Then, a salt body with a random shape and position was embedded into each random model. Wang and Ma (2020) convert natural images (e.g., from the Microsoft COCO dataset), which are rich in structure and details, to velocity models. Liu et al. (2021) generate dense-layer, fault, and salt body models, by applying multiple random trigonometric and linear equations to generate continuous, fluctuating, and complicated curves.

The open-access image datasets (e.g., the ImageNet dataset and the Microsoft COCO dataset) randomly select worldwide images with realistic targets in different realistic scenes. In contrast, the random velocity models are generated or selected based on the specific rules, rather than the prior information on the realistic geologic structures in the targeted regions. Thus, the random velocity models created in these ways suffer more or less from the selection bias (i.e. the velocity models do not randomly represent the targeted regions). Thus, dividing the random velocity models into training, validation, and test datasets to train and test the CNN, cannot guarantee the accurate prediction of the realistic velocity models in the targeted regions, because of the selection bias of the random velocity models. Therefore, successful application of CNN to approximate the inverse operator mapping from the observed seismic data or migration images to the unknown true velocity model (Araya-Polo et al., 2018; Lin & Wu, 2018; Yang & Ma, 2019; Wang & Ma, 2020; Liu et al., 2021; Zhang & Gao, 2021) also relies on the selection-unbiased training models as well as their related seismic data or migration images. Thus, the critical issues in conventional FWI (e.g., updating the background velocity model, cycle-skipping, etc.) are converted to the selection bias of the training velocity models and the overfitting of CNN. The overfitting is a phenomenon (Chicco, 2017) that the minimized performance error in the CNN training does not lead the CNN to predict the unknown true velocity model more accurately. Therefore, whether the training velocity models represent the unknown true velocity model (or the unknown true velocity model is within the distribution of the training models), becomes a valid concern and a critical issue in application of CNN to predict velocity models. Kazei et al. (2021) propose to generate random velocity models from a guiding model, by applying complicated image processing procedures (flipping, cropping, distortion, etc.). Then, the CNN is trained to approximate the mapping from full seismic waveforms to 1D vertical velocity profiles, which, compared to a 2D velocity model, are more randomly sampled and selected (less selection-biased). Thus, their method can effectively train a less-overfitted CNN to invert for the velocity model profile. Alternatively, CNN-domain FWI is proposed to apply CNN as a functional approximator to reparameterize and then replace a given starting velocity model to automatically capture its salient features as prior information (Wu & McMechan, 2018, 2019; Zhu et al., 2020; He & Wang, 2021). Then, the CNN-domain FWI iteratively constrains the inversion mainly to these captured features in CNN hidden layers, as an implicit regularization, to minimize the data residuals.

In this paper, we first propose a novel, spatially-constrained, divisive, hierarchical, k-means parcellation method to maximumly partition the velocity model into small features as the model basis. Each feature in the model basis will then be uniformly assigned by a value, which is randomly drawn from each feature's velocity distribution, to create a random velocity model. With the implementation of the proposed innovative parcellation method to prepare the training velocity model, we propose a CNN-based reflection-waveform inversion (CNN-RWI) to iteratively apply the CNN to predict the velocity model from the original

starting velocity model and the corresponding original migration image, both of which are the prior information sequentially obtained by the seismic preprocessing.

CNN-RWI mainly contains two loops. The inner loop is the CNN training step which iteratively trains CNN to approximate the training velocity models from the corresponding training starting velocity models and the training migration images. The main difference between the proposed CNN-RWI and the conventional CNN applications to predict velocity model is the outer loop. Conventional CNN applications to predict the velocity model follow the three sequential main steps once only: the training model preparation, the CNN training, and the CNN prediction. In contrast, the outer loop of CNN-RWI iteratively follows these three sequential steps until the CNN-RWI converges (e.g., the data or traveltime errors are minimized). Specifically, in the outer loop of the CNN-RWI, the training velocity models are prepared by applying the proposed parcellation method to partition the CNN-predicted velocity model, obtained at the previous iteration, to generate random training velocity models. Then, after CNN training on the latest generated random training velocity models, CNN is applied to predict the unknown true velocity model again from the original starting velocity model and the corresponding original migration image. The CNN-predicted velocity model will then be partitioned to prepare the training velocity model, at the next iteration. The two-loop mechanism enables CNN-RWI to dynamically adjust the selection-biased training velocity models to gradually predict a more-accurate velocity model, from the original starting velocity model and the corresponding original migration image. Therefore, instead of minimizing the data residuals as the FWI or the RWI does, the CNN-RWI dynamically shifts the distributions of the training velocity models as well as the CNN-predicted velocity models towards the unknown true velocity model, in a iterative deep learning manner, to implicitly reduce the selection bias.

The CNN-RWI possesses the following advantages that the conventional FWI, the CNN-domain FWI, and the CNN applications to predict velocity model cannot compete for. First, for the inversion of the deeper velocity model, the FWI and the RWI mainly update the interface and background velocities, respectively, whereas the CNN-RWI efficiently updates both of them. Secondly, both the FWI and the RWI rely on the lower-frequency data such that the traveltime errors should be within a half cycle to avoid cycle-skipping, whereas the CNN-RWI prefers the higher-frequency data to obtain a higher-resolution migration image as the CNN input data. Thirdly, the conventional CNN application used to predict the velocity model can be treated as a special case of CNN-RWI with only one iteration in the outer loop, to train the CNN to accurately approximate the training models, which generally fails to represent the unknown real velocity model in the targeted region (and thus exhibits selection bias).

The structure of the paper is outlined as follows. We first introduce the CNN-RWI algorithm with flowcharts. Then in the synthetic examples section, the CNN-RWI is compared with the conjugate-gradient FWI (CG-FWI), which is provided by the PySIT Toolbox (Hewett et al., 2020), to invert for three different portions of the Marmousi2 P-wave velocity model (Martin et al., 2006). Synthetic examples on these cropped Marmousi2 models show that the CNN-RWI outperforms the CG-FWI to invert for the velocity models. The synthetic examples also show that a second pass of the CG-FWI is needed to help the CNN-RWI further invert for the shallower parts of the velocity models, which are partially omitted by the CNN-RWI in the first pass.

2 Methodology of CNN-RWI

The CNN-RWI is an iterative inversion to apply the CNN to predict an unknown true velocity model repeatedly from the original starting velocity model and the corresponding original migration image. The original starting velocity model is assumed to be obtained by reflection tomography or migration-based velocity analysis (MVA) from the observed data. The corresponding original migration image $\mathbf{I}(\mathbf{x})$ can be obtained by reverse-time migration

(RTM) (Baysal et al., 1983; McMechan, 1983b; Whitmore, 1983), in which the image is given by

$$\mathbf{I}(\mathbf{x}) = \int_0^{\mathrm{T}} \mathbf{U}_{\mathrm{s}}(\mathbf{x}, \mathrm{t}) \mathbf{U}_{\mathrm{r}}(\mathbf{x}, \mathrm{t}) \mathrm{dt}, \tag{1}$$

where $\mathbf{U}_{\mathrm{s}}(\mathbf{x}, \mathrm{t})$ and $\mathbf{U}_{\mathrm{r}}(\mathbf{x}, \mathrm{t})$ are the source and receiver wavefields propagated into the original starting velocity model, at the location of $\mathbf{x}$ and the time t, respectively. T is the maximum traveltime.

Figure 1 shows the workflow of the CNN-RWI, which contains an outer and inner loops. The outer loop (indicated by the orange arrows in Figure 1) contains four main parts at each iteration: the preparation of the training velocity models (the purple box) , the CNN training (the red boxes), the CNN prediction (the black boxes), and the error evaluation. The inner loop is the CNN training to iteratively update CNN to fit the training velocity models (in the red box in Figure 1).

Once the original starting velocity model and the corresponding original migration (RTM) image are calculated at the beginning of the CNN-RWI, they are fixed and are repeatedly input to CNN, in the CNN prediction part (the black boxes), to predict the unknown true velocity model, at each iteration. As the CNN input data for the CNN prediction is fixed at each iteration, the main task of the CNN-RWI is to dynamically adjust the training velocity models with less selection bias to update the CNN. Thus, the core idea of the CNN-RWI is that, the better the representation of the unknown true velocity model by the training velocity models, the less the over-fitting in the CNN training part, and the better the prediction of the unknown true velocity model by CNN. The following five subsections detail the CNN-RWI algorithm.

2.1 Preparation of the training velocity models

A key novelty of the CNN-RWI is the preparation of the training velocity models from the prior velocity model (the purple box). For the first iteration of the CNN-RWI, the best prior velocity model is the original starting velocity model, which is generally obtained by preprocessing (e.g., traveltime tomography or MVA) from the observed seismic data, to estimate the unknown true velocity model. For the following iterations of the CNN-RWI, the best prior velocity model becomes the latest CNN-predicted velocity model.

The preparation of the training velocity models contains two main procedures: the generation of the model basis from a prior model, and the generation of the training velocity models from the model basis. A model basis is created by parcellating the prior velocity model into spatially related and connected features. Then, the model basis is used to create training velocity models.

2.1.1 Generation of the model basis from a prior velocity model

We propose a spatially-constrained, divisive hierarchical k-means parcellation method to parcellate a velocity model into spatially related and connected features, as a model basis. McMechan (1983a) proposed to differentiate a image and look for local maxima to parcellate a migrated common mid-point profile. Thus, the number of features parcellated by this method depends on the gradients of the image. The conventional k-means method (Lloyd, 1982; Forgy, 1965; MacQueen et al., 1967) is also not properly used for parcellation, since the conventional k-means method does not guarantee spatial continuity. The proposed parcellation method combines the k-means with the region growing, in the divisive (top-down) hierarchical structure.

A critical hyper-parameter is the threshold of the minimum feature size, which is the criterion to decide whether the feature should be further subdivided. This threshold plays the key role in weighing the homogeneity and the minimum size of each feature. The threshold should be set to maximize the number of features, without creating meaningless

small scattering regions. The mathematical explanation of the proposed parcellation method is divided into the following seven steps, and is described in the pseudo-code of Algorithm 1.

Step 1. Apply k-means clustering.

Apply k-means clustering (Lloyd, 1982; Forgy, 1965; MacQueen et al., 1967) with k=2 to partition the prior velocity model $\mathbf{V}$ into two disjoint clusters $\mathbf{C}_1$ and $\mathbf{C}_2$, where

$$\mathbf{V} = \mathbf{C}_1 \cup \mathbf{C}_2, \quad \mathbf{C}_1 \cap \mathbf{C}_2 = \emptyset. \tag{2}$$

The two disjoint clusters $\mathbf{C}_1$ and $\mathbf{C}_2$ are obtained, by minimizing the loss function

$$\mathrm{D(V)} = \sum_{\mathrm{i}=1}^{2} \sum_{\mathrm{p}\in\mathbf{C}_\mathrm{i}} ||\mathrm{v(p)} - \mu_\mathrm{i}||^2, \tag{3}$$

where v(p) denotes the velocity at the model grid point p, and μ_i denotes the mean of velocities in $\mathbf{C}_\mathrm{i}$.

The k-means clustering method partitions the velocity model $\mathbf{V}$ into the clusters $\mathbf{C}_1$ and $\mathbf{C}_2$, based on the velocity distribution, without consideration of the spatial relations and connections. Therefore, the clusters $\mathbf{C}_1$ and $\mathbf{C}_2$ may include many spatially unrelated and disconnected small regions, which should be avoided to produce unwanted scattering artifacts in the corresponding seismic data.

Step 2. Find the largest regions.

By independently computing the number of grid points in each spatially disjoint regions in each of clusters $\mathbf{C}_1$ and $\mathbf{C}_2$, the respective largest spatially connected regions are selected as $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively.

Step 3. Apply region growing.

The velocity model $\mathbf{V}$ is repartitioned into three disjoint clusters $\mathbf{V} = \{\mathbf{S}_0, \mathbf{S}_1, \mathbf{S}_2\}$ where $\mathbf{S}_1 \subseteq \mathbf{C}_1$ and $\mathbf{S}_2 \subseteq \mathbf{C}_2$ are the regions selected in step 2, and $\mathbf{S}_0 = \mathbf{V} \cap \overline{\mathbf{S}_1 \cup \mathbf{S}_2}$. Then, each model grid point $\mathrm{p} \in \mathbf{S}_0$ is merged into either $\mathbf{S}_1$ or $\mathbf{S}_2$, based on the Manhattan distance $\mathrm{D_m}$, using

$$\mathrm{p} \in \begin{cases} \mathbf{S}_1, & \text{if } \mathrm{D_m(p,\mathbf{S}_1)} \le \mathrm{D_m(p,\mathbf{S}_2)}, \\ \mathbf{S}_2, & \text{if } \mathrm{D_m(p,\mathbf{S}_1)} > \mathrm{D_m(p,\mathbf{S}_2)}, \end{cases} \tag{4}$$

where $\mathrm{D_m(p,\mathbf{S}_i)}$ (for $i \in \{1,2\}$) is defined (for a 2-D model) as

$$\mathrm{D_m(p,\mathbf{S}_i)} = \min_{\mathrm{s}\in\mathbf{S}_\mathrm{i}} |\mathrm{x_p} - \mathrm{x_s}| + |\mathrm{y_p} - \mathrm{y_s}|, \tag{5}$$

where x and y are the horizontal and vertical coordinates, and s denotes the grid point in the cluster $\mathbf{S}_\mathrm{i}$ (for $i \in \{1,2\}$).

The region growing step is numerically achieved by iteratively merging the model grid points ($\mathrm{p} \in \mathbf{S}_0$) adjacent to each of clusters $\mathbf{S}_1$ and $\mathbf{S}_2$, respectively.

Step 4. Split the velocity model into two submodels.

After the completion of the region growing, the velocity model $\mathbf{V}$ is ultimately parcellated into two spatially related and connected disjoint submodels ($\mathbf{S}_1$ and $\mathbf{S}_2$)

$$\mathbf{V} = \mathbf{S}_1 \cup \mathbf{S}_2, \quad \mathbf{S}_1 \cap \mathbf{S}_2 = \emptyset. \tag{6}$$

Steps 5 and 6. Recursively parcellate submodels.

The spatially related and connected disjoint submodels $\mathbf{S}_1$ and $\mathbf{S}_2$ are then separately and recursively parcellated, until the submodel size (the number of grid points in the submodel to be partitioned) is smaller than the threshold (the smallest tolerated feature size).

Step 7. Save the feature as a component of a model basis.

If the submodel size (the number of grid points in the submodel to be partitioned) is smaller than the threshold (the smallest tolerated feature size), the submodel will not be recursively parcellated. Instead, it will be saved as a component of the model basis (i.e., it becomes a leaf node in the hierarchical tree).

With the recursive parcellation in steps 1-7, the velocity model $\mathbf{V}$ is consequentially parcellated into many spatially-related, and connected, disjoint features (submodels) as a model basis

$$\mathbf{V} = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_\mathrm{n}\}, \tag{7}$$

where n is the number of features and

$$\mathbf{S}_\mathrm{i} \cap \mathbf{S}_\mathrm{j} = \emptyset \quad \text{if} \quad \mathrm{i} \neq \mathrm{j}. \tag{8}$$

Algorithm 1 Spatially-constrained divisive hierarchical k-means

Set a threshold as the stopping criterion when the size (the number of grid points) of the model is less than the threshold
procedure SPATIALLY-CONSTRAINED K-MEANS(model)
  if model size $\geq$ threshold **then**
    1. Apply k-means with k=2 to divide the model into two clusters a and b, based on the velocity distribution.
    2. Find the largest regions x and y in clusters a and b, respectively.
    3. Let regions x and y compete to merge other regions in the model.
    4. Split the model into submodels A and B, according to the regions x and y.
    5. SPATIALLY-CONSTRAINED K-MEANS(submodel A)
    6. SPATIALLY-CONSTRAINED K-MEANS(submodel B)
  end if
  7. Save the model as a leaf node (a component of the model basis).
end procedure

2.1.2 Generation of the training velocity model from the model basis

After the model basis $\{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_\mathrm{n}\}$ is obtained, the training velocity models can be created, based on the mean $\mathrm{E}(\mathbf{S}_\mathrm{i})$ and variance $\mathrm{Var}(\mathbf{S}_\mathrm{i})$ of the velocities in each component of the model basis, as

$$\mathrm{E}(\mathbf{S}_\mathrm{i}) = \frac{1}{\mathrm{N}_\mathrm{i}} \sum_{\mathrm{s} \in \mathbf{S}_\mathrm{i}} \mathrm{s}, \tag{9}$$

and

$$\mathrm{Var}(\mathbf{S}_\mathrm{i}) = \mathrm{E}\{[\mathbf{S}_\mathrm{i} - \mathrm{E}(\mathbf{S}_\mathrm{i})]^2\} = \frac{1}{\mathrm{N}_\mathrm{i}} \sum_{\mathrm{s} \in \mathbf{S}_\mathrm{i}} [\mathrm{s} - \mathrm{E}(\mathbf{S}_\mathrm{i})]^2, \tag{10}$$

where N_i is the number of grid points in each component (parcel) $\mathbf{S}_\mathrm{i}$. Then, the random training velocity model set $\hat{\mathbf{V}} = \{\hat{\mathbf{V}}_1, \hat{\mathbf{V}}_2, \ldots, \hat{\mathbf{V}}_\mathrm{m}\}$ (where m is the number of random training velocity models) are obtained by assigning random values, drawn from the velocity distributions of the components in the model basis, to themselves

$$\hat{\mathbf{V}}_\mathrm{k} = \{\hat{\mathbf{S}}_1, \hat{\mathbf{S}}_2, \ldots, \hat{\mathbf{S}}_\mathrm{n}\}, \quad \mathrm{k} = 1, 2, \ldots, \mathrm{m}, \tag{11}$$

where

$$\hat{\mathbf{S}}_\mathrm{i} \sim \mathcal{N}[\mathrm{E}(\mathbf{S}_\mathrm{i}),\ \mathrm{Var}(\mathbf{S}_\mathrm{i})], \quad \mathrm{i} = 1, 2, \ldots, \mathrm{n}, \tag{12}$$

where $\mathcal{N}$ denotes a normal distribution.

The assignment of a random value to each feature (equation 11), not only increases the diversity of the training velocity models, but also creates training velocity models with many different homogeneous layers (each homogeneous layer has a uniform velocity). The contrasts of velocities between any two adjacent homogeneous layers create reflectors with different reflectivity in the training velocity models, which ensure that the corresponding training seismic data contains reflection events, just like the observed data.

2.2 CNN training

After the training velocity models are created in the preparation of the training velocity models procedure (subsection 2.1), the training starting velocity model set $\widetilde{\mathbf{V}} = \{\widetilde{\mathbf{V}}_1, \widetilde{\mathbf{V}}_2, \ldots, \widetilde{\mathbf{V}}_\mathrm{m}\}$ and RTM image set $\widetilde{\mathbf{I}} = \{\widetilde{\mathbf{I}}_1, \widetilde{\mathbf{I}}_2, \ldots, \widetilde{\mathbf{I}}_\mathrm{m}\}$ could be obtained by the consistent preprocessing (e.g., reflection tomography or MVA) and the corresponding RTM, respectively, from the training seismic data, calculated from the corresponding training velocity model set $\hat{\mathbf{V}} = \{\hat{\mathbf{V}}_1, \hat{\mathbf{V}}_2, \ldots, \hat{\mathbf{V}}_\mathrm{m}\}$ (equation 11). Thus, the training starting velocity model set $\widetilde{\mathbf{V}}$ and the training RTM image set $\widetilde{\mathbf{I}}$ are obtained from synthetic data, in the same way that the original starting velocity model $\mathbf{V}_0$ and RTM image $\mathbf{I}_0$ are obtained from observed data.

Imitating the original processing to obtain the training starting velocity model (e.g., reflection tomography or MVA) is beneficial to reducing the selection bias, since the training starting velocity model is obtained in the same way as the original starting velocity model. However, this imitation is costly for all the training velocity models. Here, we apply a Gaussian filter to imitate the costly reflection tomography or MVA, to obtain the training starting velocity models by smoothing the training velocity models. Finding the best way to efficiently and accurately imitate the reflection tomography or MVA is a key to reduce the computational cost of the CNN-RWI; this is beyond the scope of this paper and so is left for future research.

After the training dataset is prepared, the training starting velocity models $\widetilde{\mathbf{V}}$ and the training RTM images $\widetilde{\mathbf{I}}$ are the input data to train the CNN to accurately reproduce the training velocity models $\hat{\mathbf{V}}$ (the red boxes in Figure 1), by minimizing the loss function

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \frac{1}{2} \sum_{\mathrm{k}=1}^{\mathrm{m}} \|\hat{\mathbf{V}}_\mathrm{k} - \mathbf{G}(\widetilde{\mathbf{V}}_\mathrm{k}, \widetilde{\mathbf{I}}_\mathrm{k}; \mathbf{w})\|_2^2, \tag{13}$$

where $\mathbf{G}$ is the convolutional neural network (CNN), parameterized by the weights $\mathbf{w}$, and $\mathbf{w}^*$ is the optimized CNN weights after CNN training (equation 13). By iteratively minimizing the loss function (equation 13), the CNN is assumed to accurately approximate the mapping from $\widetilde{\mathbf{V}}$ and $\widetilde{\mathbf{I}}$ to $\hat{\mathbf{V}}$, in the training dataset.

2.3 CNN prediction

In the CNN prediction (the black boxes in Figure 1), the combination of the original starting velocity model $\mathbf{V}_0$ and the corresponding RTM image $\mathbf{I}_0$ are input to the trained CNN to predict the unknown true velocity model

$$\mathbf{V}_\mathrm{p} = \mathbf{G}(\mathbf{V}_0, \mathbf{I}_0; \mathbf{w}^*), \tag{14}$$

where $\mathbf{V}_\mathrm{p}$ is the CNN-predicted velocity model.

2.4 Error evaluation

After the velocity model $\mathbf{V}_\mathrm{p}$ is predicted by the CNN, forward modeling is needed to obtain the corresponding synthetic seismic data for error evaluation (e.g., computing the least-squares data residuals). Unlike the conventional FWI which updates the velocity model by minimizing the data error at each iteration, the CNN-RWI calculates the data error for

the indirect evaluation of the accuracy of the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$. If the data error, obtained from the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$, does not meet the convergence criteria (e.g., a threshold of the minimum data error), the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$ will become the latest generated prior model, to be parcellated to generate the training velocity model set (subsection 2.1), for the next iteration.

2.5 Dynamic adjustment of the training dataset

Subsections 2.1-2.4 above correspond to the four main parts in the outer loop of the CNN-RWI (Figure 1). Apart from the novelty to generate the training velocity models from the prior velocity model (subsection 2.1), another key novelty of the CNN-RWI is the dynamic adjustment of the training dataset based on the latest prior velocity model at each iteration in the outer loop (indicated by the orange arrows in Figure 1). The latest prior velocity model is defined as the original starting velocity model $\mathbf{V}_0$ at the first iteration, or the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$, obtained at each of the following iterations.

As the original starting velocity model contains the inaccurate prior content information (e.g., the inaccurate background velocity), the corresponding training velocity model set $\hat{\mathbf{V}}$, at the first iteration, is selection-biased, since $\hat{\mathbf{V}}$ is a good representation of the original starting velocity model $\mathbf{V}_0$, rather than the unknown true velocity model $\mathbf{V}_\mathrm{t}$. In contrast, in the ideal case, the preprocessing to obtain the training starting velocity model set $\widetilde{\mathbf{V}}$ and the training migration image set $\widetilde{\mathbf{I}}$ from the training velocity model set $\hat{\mathbf{V}}$ could be highly consistent with the preprocessing to obtain the original starting velocity model $\mathbf{V}_0$ and the original migration image $\mathbf{I}_0$ from the unknown true velocity model $\mathbf{V}_\mathrm{t}$. In other words, the spatial relations are not as selection-biased as the content information in the training dataset $\{\widetilde{\mathbf{V}}, \widetilde{\mathbf{I}}, \hat{\mathbf{V}}\}$, since the former shares the similar preprocessing with the real application, but the latter are different from the content information in the real dataset $\{\mathbf{V}_0, \mathbf{I}_0, \mathbf{V}_\mathrm{t}\}$. Therefore, by minimizing the loss function (equation 13) to accurately approximate the mapping from $\widetilde{\mathbf{V}}$ and $\widetilde{\mathbf{I}}$ to $\hat{\mathbf{V}}$ in the training dataset, the CNN predicts the unknown velocity model $\mathbf{V}_\mathrm{p}$, which is generally more accurate than the original starting velocity model $\mathbf{V}_0$ but is still far from the unknown true velocity model $\mathbf{V}_\mathrm{t}$. Then, CNN-RWI dynamically adjusts the training dataset $\{\widetilde{\mathbf{V}}, \widetilde{\mathbf{I}}, \hat{\mathbf{V}}\}$, by parcellating the latest CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$ obtained at the previous iteration, as the latest prior velocity model, to create the new model basis and the corresponding training velocity model set $\hat{\mathbf{V}}$ (subsection 2.1) for the CNN training (subsection 2.2).

The convergence of the CNN-RWI relies on the assumption that the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$ is more accurate than the training velocity model set $\hat{\mathbf{V}}$, which is derived from, and so represents, the CNN-predicted velocity model obtained at the previous iteration (or the original starting velocity model at the first iteration). This assumption is reasonable, since the original velocity model $\mathbf{V}_0$ and the original migration image $\mathbf{I}_0$ input to the CNN to predict the velocity model $\mathbf{V}_\mathrm{p}$ (equation 14), at each iteration, are obtained from the observed data (Figure 1), which are acquired from the unknown true velocity model $\mathbf{V}_\mathrm{t}$. In contrast, the CNN input data for the CNN training ($\widetilde{\mathbf{V}}$ and $\widetilde{\mathbf{I}}$ in equation 13) are obtained from the synthetic data, which are calculated from the training velocity model set $\hat{\mathbf{V}}$ (the red boxes in Figure 1). Thus, the CNN-predicted velocity model $\mathbf{V}_\mathrm{p}$ will be more likely to be closer to the unknown true velocity model $\mathbf{V}_\mathrm{t}$ than the training velocity model set $\hat{\mathbf{V}}$ will be. Therefore, this dynamic adjustment of the training velocity model, at each iteration, implicitly reduces the selection bias of the training velocity model set $\hat{\mathbf{V}}$, and so makes the CNN more representative of the unknown true velocity model. Consequentially, the CNN progressively predicts the unknown true velocity model $\mathbf{V}_\mathrm{p}$ more accurately, by implicitly reducing the selection bias, instead of minimizing the data error as FWI does.

3 Numerical Examples

In this section, three different portions of the Marmousi2 P-wave velocity model (Martin et al., 2006) with a grid of 256×256 are used as the true velocity models (the areas inside the red, black, and cyan boxes in Figure 2). The synthetic tests on the three true velocity models are divided into tests 1 and 2, using the starting velocity models, smoothed by a 3×3 Gaussian filter for 400 and 1600 iterations, respectively, to mitigate the preprocessing (e.g., traveltime tomography or MVA) with different accuracy. Tests 1 and 2 compare the performance and the sensitivity of the CNN-RWI with the CG-FWI, which is provided by the PySIT Toolbox (Hewett et al., 2020), to the different accuracy of the starting velocity models. The vertical and horizontal extents of the model are 3.2 km with a 0.0125 km spatial sampling increment. The reason to crop the original Marmousi2 P-wave velocity model into three square velocity models is to evaluate the performance of the CNN-RWI and the CG-FWI to effectively and efficiently update the deeper part of the velocity model from the reflection data. To simulate the observed data, 25 explosive Ricker-wavelet sources with a dominant frequency of 20 Hz (and 7 Hz for the CG-FWI as a reference group) are spaced every 0.125 km, and 256 receivers are spaced every 0.0125 km. All sources and receivers are at the surface. The recording time is 3.5 s with a 1.0 ms time sampling increment.

Here, we use a U-net convolutional neural network (Figure 3) (Ronneberger et al., 2015) to predict the velocity model, which has been empirically shown to be a powerful and an effective CNN architecture for generation tasks (Ronneberger et al., 2015; Yang & Ma, 2019). The down- and up-sampling in the CNN architecture (Figure 3) are achieved by convolution and transposed convolution operations with stride=2, respectively.

As the main contribution of this paper is the CNN-RWI algorithm (the methodology in section 2 and the workflow in Figure 1), rather than the advance of the CNN architecture, other CNN architectures can also be used, as long as they can accurately predict the training velocity models in the CNN training (subsection 2.2).

To prepare the training model set with 24 samples at each iteration, the spatially-constrained, divisive, hierarchical k-means parcellation method (subsection 2.1.1) is applied, to build the model basis from the original starting velocity model for the first iteration, or from the CNN-predicted velocity models for the following iterations, respectively. The threshold of this parcellation method is set to be 655, which is one percent of the total number of model grid points ($256 \times 256 = 65536$). The threshold should be set to maximize the number of components in the model basis (the number of the leaf nodes in the hierarchical tree), with the limitation of not creating meaningless small scattering regions. Another consideration of this threshold is the computational cost. The smaller threshold increases the depth of the hierarchical tree, and so exponentially increases the computational time to further parcellate the submodels into smaller features as the components of the model basis. Therefore, we set the threshold to be 655, one percent of the total number of model grid points (65536), to balance the quality of the model basis and the computational cost. For the velocity model containing large-scale salt bodies or reservoirs, the CNN-domain FWI (Wu & McMechan, 2018, 2019; Zhu et al., 2020; He & Wang, 2021) to focus the inversion mainly to the prior features in the starting velocity model (e.g., salt bodies, reservoir, etc.) might be more appropriate than the CNN-RWI. The comparison between the CNN-domain FWI and CNN-based RWI is beyond the scope of this paper and so is left for future comparison.

Because of the faster convergence of the CNN-RWI, the CNN-RWI iteratively predict the velocity models from the original starting velocity model and its corresponding original migration images in 16 iterations. Then, the velocity models, inverted by the CNN-RWI after 16 iterations, are respectively used as the starting velocity model for the CG-FWI in additional 39 iterations in a second pass. Thus, the overall iteration number of the CNN-RWI in Pass 1 and the CG-FWI in Pass 2 (55 iterations) is a half of that of the CG-FWI (110 iterations) based on the original starting velocity models as the CNN-RWI does.

3.1 Test 1: Synthetic tests on the more accurate starting velocity models

Figure 4 compares the performance of the CNN-RWI and the CG-FWI using the starting velocity models a-c (Figure 4a-4c). Figure 4d-4f are the CNN-RWI-inverted velocity models at the 16th iteration, based on the source with a dominant frequency of 20 Hz. Figure 4g-4i shows the CG-FWI-inverted velocity models at the 39th iteration, by using the CNN-RWI inverted velocity models (Figure 4d-4f) as the starting velocity models in a second pass. Figures 4j-4l and 4m-4o show the CG-FWI-inverted velocity models, at 110th iteration, based on the source with a dominant frequency of 20 Hz and 7 Hz, respectively. Figures 5 shows the corresponding model residuals between the models in Figure 4 and the corresponding true velocity models (Figure 2).

Figures 4 and 5 clearly show that the CNN-RWI inverts for the velocity models (Figure 4d-4f), more accurately than the CG-FWI does (both the 20 Hz and 7 Hz results in Figures 4j-4o), especially in the deeper part of the velocity model. Figure 6d-6o uses the model residuals between the velocity models in Figures 4d-4o and the original starting velocity models a-c (Figure 4a-4c) to analyze which parts of the initial velocity models are predominantly updated by the CNN-RWI and the CG-FWI, respectively. Figure 6d and 6f shows that the CNN-RWI dramatically updates both the background velocity and the interfaces in the velocity models (Figures 4d and 4f), to the correct direction, at almost all depths. The correct updates of the velocity models (Figures 4d and 4f) can be quantitatively characterized by computing the RMS model error, across all positions for each depth (Figure 7). Figure 7 shows that these two CNN-RWI-inverted velocity models (the red lines in Figure 7a and 7c) as well as the corresponding CG-FWI-inverted velocity models in Pass 2 (the blue lines in Figure 7a and 7c) correspond to the minimum RMS model errors, at almost all depths. By comparison, Figures 5e and 6e show that the CNN-RWI selectively updates the salient features of the relatively complicated velocity model (with many faults) more accurately than the CG-FWI does (Figures 5k-5n and 6k-6n), approximately above the 2.6 km in depth. This is consistent with the smaller RMS model errors of the CNN-RWI-inverted velocity model (the red line in Figure 7b) as well as the corresponding CG-FWI-inverted velocity model in Pass 2 (the blue line in Figure 7b) than those of the CG-FWI (the black and magenta lines in Figure 7b), approximately above 2.6 km in depth.

Figures 4j-4l, 5j-5l, and 6j-6l show that the CG-FWI based on the source with a dominant frequency of 20 Hz, mainly updates the shallower background velocities as well as the interfaces at all depths, which is in accordance with the theoretical and numerical analysis of the FWI gradient (Xu et al., 2012; Alkhalifah, 2014; Wu & Alkhalifah, 2015; Yao et al., 2020). In contrast, the CG-FWI based on the source with a dominant frequency of 7 Hz, updates a relatively deeper velocity more accurately, except that some of the high-contrast velocity features are updated in the wrong directions (the two salient layers above and below 2.1 km in depth in Figure 5m and the salient overarching layer between 1.1 km and 2.1 km in depth in Figure 5o). As these layers, which are wrongly updated by the CG-FWI, are correctly updated by the CNN-RWI (Figure 5d and 5f), the CG-FWI based on the source with a dominant frequency of 20 Hz, continues to update these layers more accurately (Figure 5g and 5i), in Pass 2. Figure 7a and 7c also quantitatively illustrates that the velocity models inverted by the CG-FWI in Pass 2 (the blue lines in Figure 7a and 7c) have the lowest RMS model errors among these inversion results, whereas the CG-FWI (at both 20 Hz and 7 Hz) inverts the velocity model least accurately (the black and magenta lines in Figure 7a and 7c). Figure 7b reveals that the CNN-RWI inverts the velocity model with a relatively smaller RMS model error than that of the CG-FWI at both 20 Hz and 7 Hz at each depth, therefore the CG-FWI-inverted velocity model in Pass 2 has the lowest RMS model error (compare the red and blue lines with the black and magenta lines in Figure 7b). Nevertheless, Figure 4 shows that the velocity model inverted by the CNN-RWI (Figure 4e) based on the starting velocity model (model b in Figure 4b) seems visually less accurate than the velocity models inverted by the CG-FWI (Figure 4k and 4n), partially because the interfaces in the velocity models, especially in the shallower depth, inverted by the CG-FWI,

are more obvious than those inverted by the CNN-RWI (compare Figure 6e with Figure 6k and 6n).

Figure 8 compares the CNN-RWI with the CG-FWI, based on the model and data errors. Compared to the velocity model inverted by the CNN-RWI (the red lines in Figures 8a-8c) which dramatically decreases and fast converges, the velocity models inverted by the CG-FWI for both the 20 Hz and 7 Hz cases (the black and magenta lines in Figures 8a-8c) gradually decreases and slowly converges to a relatively much higher RMS model errors. The cyan dots in Figure 8a-8c correspond to the RMS model errors of the 24 training velocity models at the first 16 iterations of the CNN-RWI. It verifies the assumption that the CNN-predicted velocity models will be more accurate than the training velocity models at each iteration, which is the key motivation for CNN-RWI to iteratively predict the more and more accurate velocity models. Figure 8a and 8d shows that the CG-FWI for the 7 Hz case (the magenta line in Figures 8a and 8d) is trapped into a bad local minimum at the 46th iteration (the magenta circle in Figure 8a and 8d) to continue to minimize the data errors (the magenta line in Figure 8d), by wrongly increasing the velocities in the two salient layers, which are located above and below 2.1 km in Figure 4m. Thus, the model error increases from the 46th to the 110th iterations, with the decreasing data errors in CG-FWI (the magenta line in Figures 8a and 8d). In contrast, the velocity model inverted by the CG-FWI in Pass 2 by using the CNN-RWI-inverted velocity models as the starting velocity model are gradually more accurate, so both the RMS model and data errors are reduced at each iteration (the blue lines in 8a and 8d).

Figure 8 shows that the RMS model errors of the CNN-RWI (the red lines in Figure 8a-8c) decrease dramatically in the first several iterations and then converge with slight fluctuations, whereas the RMS data errors of the CNN-RWI (the red lines in Figure 8d-8f) decrease sharply but then converge with severe fluctuations. The main reason for these inconsistent behaviors of the fluctuations is that the CNN-RWI applies the approximated spatial mapping, gradually learned from the dynamically adjustable training dataset at each iteration (subsection 2.5), to predict the velocity model, instead of steadily minimizing the data errors as FWI does. When the CNN-predicted velocity model is less accurate, the spatial mapping in the training dataset (subsection 2.2), efficiently guides CNN to predict a more accurate velocity model, which corresponds to a sharp decreases in both the RMS model and data error (the red lines at the first several iterations in Figure 8). These more accurate velocity models still contain some variation of velocities in local areas, whose velocities and locations may vary at each iteration, due to the imperfect prediction of the velocity by the highly non-linear CNN. As the average accuracy of the CNN-predicted velocity model is dramatically increased, these local velocity errors and the corresponding data errors are less predominant, so both the RMS model and data errors do not fluctuate, at the first several iterations. However, when the CNN-predicted velocity model converges, the average accuracy of the CNN-predicted velocity model cannot sharply increase. Thus, the variance of the local errors in the CNN-predicted velocity model (the slight fluctuation) plays a predominant role in affecting the data errors (the severe fluctuation). As the seismic data is very sensitive to the model error, the data error fluctuates more severely than the model error does. For example, the incorrect shallower velocity predicted by the CNN in the CNN-RWI-inverted velocity model (circled by the white and black boxes in Figures 4f and 5f, respectively) is the main reason that the corresponding data error dramatically increases (the red line in Figure 8f). Although the velocity model predicted by the CNN is, on average, much more accurate with the iterative inversion (the decreasing trend of the red line in Figure 8c), the accumulated incorrect velocities in this shallower area significantly increases the corresponding data error (the increasing trend of the red line in Figure 8f from the 7th to the 16th iterations). After these incorrect velocities are corrected by the CG-FWI (compare the white boxes in Figure 4f and 4i and the black boxes in Figure 5f and 5i), the RMS data error dramatically decreases (compare the red and blue lines in Figure 8f), whereas the RMS model error decreases slightly (compare the red and blue lines in Figure 8c), since this shallower area is not salient in the whole velocity model.

The comparison of the performance of the CNN-RWI and CG-FWI on different models reveals that the CNN-RWI outperforms the CG-FWI to invert for the velocity model (especially the background velocity model) at almost all depths. In contrast, CG-FWI effectively updates the shallower background velocity model along the diving and refracted waves more accurately. The background velocity and the interfaces in the deeper velocity models can be updated by the CG-FWI, however they are less accurate and may even be updated in the wrong direction. By applying CG-FWI in Pass 2 to continue to invert for the velocity model, which is better recovered by the CNN-RWI in Pass 1, CG-FWI can effectively update interfaces, but not the background velocity models (Figures 6a-6c). Therefore, the CNN-RWI and the CG-FWI are sensitive to the complexity of the velocity model (e.g., the velocity model in the black box in Figure 2), and the CNN-RWI inverts for the velocity models more accurately than CG-FWI does, for all velocity models, at almost all depths (Figures 4-8).

3.2 Test 2: Synthetic tests on the less accurate starting velocity models

To analyze the sensitivity, of the performance of the CNN-RWI and the CG-FWI, to the accuracy of the starting velocity model, both approaches are tested to invert for the velocity models, using the less accurate starting velocity models A-C (Figure 9a-9c), by following the exactly the same test and comparison procedures as those in Test 1. Compared to the more accurate starting velocity models a-c (Figure 4a-4c) containing salient features, the starting velocity models A-C (Figure 9a-9c) are too smooth to contain salient features. Similar to the results in Test 1, the CNN-RWI outperforms the CG-FWI (at both 20 Hz and 7 Hz) to invert for the velocity models (compare Figure 9d-9f with Figures 9j-9l and 9m-9o, respectively). However, all of the inversion results in Test 2 (Figure 9) are less accurate than the corresponding inversion results in Test 1 (Figure 4). The comparison of Figures 4 and 9 reveal that both the CNN-RWI and the CG-FWI are sensitive to the accuracy of the starting velocity models.

The residual model in Figure 10d shows that the CNN-RWI slightly updates the two salient layers, which are located above and below 2.1 km in depth, in the correct direction. In contrast, Figure 10j and 10m shows that the CG-FWI incorrectly updates these salient layers in the velocity model (Figure 9j and 9m). The RMS model errors across all positions for each depth (Figure 11a) quantitatively verify that the RMS model errors of the CG-FWI for both 20 Hz and 7 Hz are the highest in comparison with others (compare the RMS errors indicated by the arrows 1-4 in Figure 11). In contrast, these two salient layers inverted by the CNN-RWI in Pass 1 as well as the CG-FWI in Pass 2 (the red and blue lines in Figure 11a, respectively) are the lowest. Actually, Figures 9-11 illustrate that the velocity models inverted by the CNN-FWI and then by the CG-FWI in Pass 2 are most accurate results at almost all depths, in Test 2.

Figure 12 shows similar trends, in both RMS model and data errors of both CNN-RWI and CG-FWI in Test 2, to those in Test 1 (Figure 8), except that all of the model and data error levels are higher in Test 2 than those in Test 1. The velocity model errors of the CG-FWI-inverted velocity models increase (both the 20 Hz and 7 Hz cases), with the decrease in the corresponding data errors, just like those in Test 1 (compare the black and magenta lines in Figures 8a and 12a and in Figures 8d and 12d). The RMS model and data errors corresponding to the CNN-RWI dramatically decrease and converge to error levels which are higher than those in Test 1 (compare the red lines in Figures 8 and 12). The fluctuations in the model and data error curves, respectively, are similar to those in Test 1 (compare in the white boxes in Figures 4 and 9 and in the black boxes in Figures 5 and 10). The underlying reason for the opposite directions in the trends of the model and data error curves has been analyzed in the previous subsection and so is omitted here.

The synthetic tests on the three portions of Marmousi models (Figure 3), using the different smoothed starting velocity models (Figure 4a-4c in Test 1 and Figure 9a-9c in

Test 2) illustrate that the performance of both the CNN-RWI and CG-FWI are sensitive to the complexity of the true velocity models and the accuracy of the corresponding starting velocity models. The synthetic tests 1 and 2 also show that the CNN-RWI inverts for the deeper velocity model more accurately than the CG-FWI.

4 Discussion

We illustrate how the CNN-RWI iteratively predicts increasingly accurate velocity models from the original starting velocity model and the corresponding migration image, without the need of the data-fitting procedure in the conventional FWI (e.g., CG-FWI). The highest accuracy of the CNN-predicted velocity model is partially constrained by the accuracy of the original starting velocity model, for two reasons. Firstly, the more accurate the original starting velocity model used is, the more accurate the original migration image will be. Both of them reduce the ill-posedness of the spatial mapping from themselves to the unknown true velocity model, which enables the CNN to predict the velocity model more accurately (subsection 2.3). Secondly, a more accurate original starting velocity model contains more abundant, and accurate, prior information for the generation of the model basis, which increases the accuracy (i.e., decreases the selection bias) of the training velocity model set and the corresponding CNN-predicted velocity model. Therefore, the CNN-RWI apparently replaces the data-fitting procedure, in conventional FWI, by the CNN training and prediction. However, the accurate prior information in the starting velocity model is needed for the CNN-RWI to reduce the ill-posedness of the mapping and to generate a more accurate model basis for the CNN training, just as it is needed for the conventional FWI to better tackle the cycle-skipping issue. In the synthetic tests 1 and 2, the starting velocity models are obtained by applying a 3×3 Gaussian filter to smooth the true velocity models for 400 and 1600 iterations, respectively, to imitate the preprocessing. Synthetic tests 1 and 2 clearly show that both the conventional FWI and the CNN-RWI are sensitive to the accuracy of the starting velocity models, as well as to the complexity of the unknown true velocity models.

In the generation of the model basis from a prior velocity model (subsection 2.1.1), one of the main differences between the proposed spatially-constrained, divisive hierarchical k-means parcellation method, and the conventional k-means clustering, is that the proposed parcellation method retains the structural relations between features, and the connectivity within each feature, which is more reasonable and suitable for preparing the training velocity models. In contrast, the conventional k-means clustering methods (Lloyd, 1982; Forgy, 1965; MacQueen et al., 1967) may contain many spatially unrelated and disconnected regions, which become unwanted scattering points. Another main difference is the purpose of the clustering. The conventional k-means clustering aims to cluster and compress a dense model (with a relatively large number of parameters) to a sparse model (with relatively few parameters), by minimizing the difference between the sparse and dense models. Therefore, the sparse model, compressed by conventional k-means clustering methods, is still an accurate approximation and replacement of the dense parameter model. In contrast, the goal of the clustering in CNN-RWI is to parcellate the prior velocity model into small features as the model basis, to create random training velocity models with different homogeneous features and sharp boundaries. These training velocity models are then used to represent the unknown true velocity model for the CNN training. Therefore, the training velocity models, created by the proposed parcellation method, are considered as extremely lossy and unacceptable sparse models in their reconstruction and compression aspects, since the training velocity models are not optimized to fit the original velocity model.

In the conventional FWI (e.g., CG-FWI), as the data misfit is minimized at each iteration, the RMS data error curve typically decreases. In contrast, the CNN-RWI does not require a (data) misfit function to be minimized to update the velocity model as FWI does. Instead, the original starting velocity model (the tomographic component) and the original high-resolution migration image (the migration component) are repeatedly input to CNN as

prior information to constrain the CNN to predict the velocity model, based on these prior information as well as the spatial mapping learned from the dynamically adjusted training dataset (subsection 2.5). Nevertheless, the data misfit as well as other validation tools (e.g., travel-time misfits, etc.) can be used as error evaluations to analyze the performance of CNN-RWI at each iteration.

The computational cost of the proposed spatially-constrained, divisive, hierarchical k-means parcellation method can be ignored since it is very fast to parcellate a prior velocity model to generate the random training velocity models, at each iteration. Therefore, the computational cost of the CNN-RWI mainly depends on the number of samples in the training dataset and the efficiency to obtain the training velocity models and the training migration images, as well as the CNN training. In Synthetic tests 1 and 2, 24 samples are experimentally proved to be adequate and very efficient for the CNN training in a GPU environment. If the training migration images are obtained in a parallel programming environment, the computation time of the training migration images will be approximately equal to that of one migration image, which will make the migration processing most efficient, at each iteration. The most dispute and uncertain procedure in the CNN-RWI is the means to obtain the training starting velocity models at each iteration. Applying the original processing (e.g., traveltime tomography or MVA) to obtain the training starting velocity models at each iteration, is very accurate but time-consuming. Instead, smoothing the training velocity models to imitate the original processing to obtain the training velocity models is very fast but less accurate. Therefore, development of a better way to balance the accuracy and the computational cost to obtain the training starting velocity models is a critical issue, which needs to be solved for real application of the CNN-RWI, but beyond the scope of this paper. It is therefore left for future research.

5 Conclusions

A novel method of convolutional-neural-network-based reflection-waveform inversion (CNN-RWI) is proposed to iteratively predict more accurate velocity models. The CNN-RWI contains inner and outer loops. The inner loop trains the CNN to accurately predict the training velocity models from the training starting velocity models and the corresponding migration images. The outer loop iteratively uses the latest prior velocity model predicted by the CNN after the CNN training, to create a more representative training dataset for the CNN training at the next iteration. The creation of the training velocity models from the prior velocity model is achieved by the proposed spatially constrained, divisive, hierarchical k-means parcellation method. The outer loop iteratively improves the training velocity models, to enable CNN to progressively predict more accurate velocity models. Synthetic examples using three portions of the Marmousi2 P-wave velocity shows that CNN-RWI predicts both the shallow and deep parts of the velocity models more accurately than the conjugate-gradient FWI does. Both the CNN-RWI and the CG-FWI are sensitive to the accuracy of the starting velocity model as well as to the complexity of the unknown true velocity model.

Acknowledgments

The research is supported by the Original Innovation Research Program of the Chinese Academy of Sciences (CAS) under grant number ZDBS-LY-DQC003, and the Key Research Program of the Institute of Geology & Geophysics, CAS under grant numbers IGGCAS-2019031 & SZJJ-201901. The participation of G.A.M is supported by the University of Texas at Dallas Geophysical Consortium. This paper is Contribution No. XXXX for the Department of Geosciences at the University of Texas at Dallas. The associated data used by this paper are available at the following link with the DOI from Zenodo: https://doi.org/10.5281/zenodo.5920504

References

Alkhalifah, T. (2014). Scattering-angle based filtering of the waveform inversion gradients. *Geophysical Journal International*, *200*(1), 363–373.

Araya-Polo, M., Jennings, J., Adler, A., & Dahlke, T. (2018). Deep-learning tomography. *The Leading Edge*, *37*(1), 58–66.

Bamberger, A., Chavent, G., Hemon, C., & Lailly, P. (1982). Inversion of normal incidence seismograms. *Geophysics*, *47*(5), 757–770.

Baysal, E., Kosloff, D. D., & Sherwood, J. W. (1983). Reverse time migration. *Geophysics*, *48*(11), 1514–1524.

Chi, B., Gao, K., & Huang, L. (2017). Least-squares reverse time migration guided full-waveform inversion. In *SEG Technical Program Expanded Abstracts 2017* (pp. 1471–1475). Society of Exploration Geophysicists.

Chicco, D. (2017). Ten quick tips for machine learning in computational biology. *BioData Mining*, *10*(1), 1–17.

Crase, E., Pica, A., Noble, M., McDonald, J., & Tarantola, A. (1990). Robust elastic nonlinear waveform inversion: Application to real data. *Geophysics*, *55*(5), 527–538.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255).

Fei, T. W., Luo, Y., Yang, J., Liu, H., & Qin, F. (2015). Removing false images in reverse time migration: The concept of de-primary reverse time migration. *Geophysics*, *80*(6), S237–S244.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, *21*, 768–769.

Fukushima, K., & Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and Cooperation in Neural Nets* (pp. 267–285). Springer.

Gauthier, O., Virieux, J., & Tarantola, A. (1986). Two-dimensional nonlinear inversion of seismic waveforms: Numerical results. *Geophysics*, *51*(7), 1387–1403.

He, Q., & Wang, Y. (2021). Reparameterized full-waveform inversion using deep neural networks. *Geophysics*, *86*(1), V1–V13.

Hewett, R. J., Demanet, L., & Team, T. P. (2020, January). *PySIT: Python Seismic Imaging Toolbox.* Zenodo. Retrieved from `https://doi.org/10.5281/zenodo.3603367` doi: 10.5281/zenodo.3603367

Hu, L., & McMechan, G. A. (1987). Wave-field transformations of vertical seismic profiles. *Geophysics*, *52*(3), 307–321.

Irabor, K., & Warner, M. (2016). Reflection FWI. In *SEG Technical Program Expanded Abstracts 2016* (pp. 1136–1140). Society of Exploration Geophysicists.

Kazei, V., Ovcharenko, O., Plotnitskii, P., Peter, D., Zhang, X., & Alkhalifah, T. (2021). Mapping full seismic waveforms to vertical velocity profiles by deep learning. *Geophysics*, *86*(5), 1–50.

Khalil, A., Sun, J., Zhang, Y., & Poole, G. (2014). RTM noise attenuation and image enhancement using time-shift gathers. In *76th EAGE Conference and Exhibition 2014* (Vol. 2014, pp. 1–5).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems* (pp. 1097–1105).

Lailly, P. (1983). The Seismic Inverse Problem as a Sequence of Before Stack Migrations. In *Conference on Inverse Scattering–Theory and Application* (Vol. 11, p. 206). SIAM.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541–551.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems* (pp. 396–404).

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lian, S., Yuan, S., Wang, G., Liu, T., Liu, Y., & Wang, S. (2018). Enhancing low-wavenumber components of full-waveform inversion using an improved wavefield decomposition method in the time-space domain. *Journal of Applied Geophysics*, *157*, 10–22.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In *European Conference on Computer Vision* (pp. 740–755).

Lin, Y., & Wu, Y. (2018). Inversionnet: a real-time and accurate full waveform inversion with convolutional neural network. *The Journal of the Acoustical Society of America*, *144*(3), 1683–1683.

Liu, B., Yang, S., Ren, Y., Xu, X., Jiang, P., & Chen, Y. (2021). Deep-learning seismic full-waveform inversion for realistic structural models. *Geophysics*, *86*(1), R31–R44.

Liu, F., Zhang, G., Morton, S. A., & Leveille, J. P. (2011). An effective imaging condition for reverse-time migration using wavefield decomposition. *Geophysics*, *76*(1), S29–S39.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Ma, Y., Hale, D., Gong, B., & Meng, Z. (2012). Image-guided sparse-model full waveform inversion. *Geophysics*, *77*(4), R189–R198.

MacQueen, J., et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (Vol. 1, pp. 281–297).

Martin, G. S., Wiley, R., & Marfurt, K. J. (2006). Marmousi2: An elastic upgrade for Marmousi. *The Leading Edge*, *25*(2), 156–166.

McMechan, G. A. (1983a). Application of image enhancement and pattern recognition algorithms to seismic reflection data. *Bulletin of the Seismological Society of America*, *73*(1), 307–314.

McMechan, G. A. (1983b). Migration by extrapolation of time-dependent boundary values. *Geophysical Prospecting*, *31*(3), 413–420.

Mora, P. (1987). Nonlinear two-dimensional elastic inversion of multioffset seismic data. *Geophysics*, *52*(9), 1211–1228.

Mora, P. (1989). Inversion= migration+ tomography. *Geophysics*, *54*(12), 1575–1586.

Pratt, R. G. (1999). Seismic waveform inversion in the frequency domain; part 1, theory and verification in a physical scale model. *Geophysics*, *64*(3), 888–901.

Pratt, R. G., Shin, C., & Hick, G. (1998). Gauss–newton and full newton methods in frequency–space seismic waveform inversion. *Geophysical Journal International*, *133*(2), 341–362.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention* (pp. 234–241). Springer.

Sun, D., Jiao, K., Cheng, X., Xu, Z., Zhang, L., & Vigh, D. (2017). Born modeling based adjustive reflection full waveform inversion. In *SEG Technical Program Expanded Abstracts 2017* (pp. 1460–1465). Society of Exploration Geophysicists.

Tarantola, A. (1984). Inversion of seismic reflection data in the acoustic approximation. *Geophysics*, *49*(8), 1259–1266.

Virieux, J., & Operto, S. (2009). An overview of full-waveform inversion in exploration geophysics. *Geophysics*, *74*(6), WCC1–WCC26.

Wang, F., Chauris, H., Donno, D., & Calandra, H. (2013). Taking advantage of wave field decomposition in full waveform inversion. In *75th EAGE Conference & Exhibition Incorporating SPE EUROPEC 2013* (pp. cp–348). European Association of Geoscientists & Engineers.

Wang, G., Guo, Q., Alkhalifah, T., & Wang, S. (2021). Frequency-domain reflection waveform inversion with generalized internal multiple imaging. *Geophysics*, *86*(5), R701–R710.

Wang, W., & Ma, J. (2020). Velocity model building in a crosswell acquisition geometry with image-trained artificial neural networks. *Geophysics*, *85*(2), U31–U46.

Whitmore, N. D. (1983). Iterative depth migration by backward time propagation. In *SEG Technical Program Expanded Abstracts 1983* (pp. 382–385). Society of Exploration Geophysicists.

Wu, Y., & McMechan, G. A. (2018). Feature-capturing full waveform inversion using a convolutional neural network. In *SEG Technical Program Expanded Abstracts 2018* (pp. 2061–2065). Society of Exploration Geophysicists.

Wu, Y., & McMechan, G. A. (2019). Parametric convolutional neural network-domain full-waveform inversion. *Geophysics*, *84*(6), R881–R896.

Wu, Z., & Alkhalifah, T. (2015). Simultaneous inversion of the background velocity and the perturbation in full-waveform inversion. *Geophysics*, *80*(6), R317–R329.

Wu, Z., & Alkhalifah, T. (2017). Efficient scattering-angle enrichment for a nonlinear inversion of the background and perturbations components of a velocity model. *Geophysical Journal International*, *210*(3), 1981–1992.

Xu, S., Wang, D., Chen, F., Lambaré, G., & Zhang, Y. (2012). Inversion on reflected seismic wave. In *SEG Technical Program Expanded Abstracts 2012* (pp. 1–7). Society of Exploration Geophysicists.

Yang, F., & Ma, J. (2019). Deep-learning inversion: A next-generation seismic velocity model building method. *Geophysics*, *84*(4), R583–R599.

Yao, G., da Silva, N., Kazei, V., Wu, D., & Yang, C. (2019). Extraction of the tomography mode with nonstationary smoothing for full-waveform inversion. *Geophysics*, *84*(4), R527–R537.

Yao, G., da Silva, N., Warner, M., & Kalinicheva, T. (2018). Separation of migration and tomography modes of full-waveform inversion in the plane wave domain. *Journal of Geophysical Research: Solid Earth*, *123*(2), 1486–1501.

Yao, G., & Wu, D. (2017). Reflection full waveform inversion. *Science China Earth Sciences*, *60*(10), 1783–1794.

Yao, G., Wu, D., & Wang, S.-X. (2020). A review on reflection-waveform inversion. *Petroleum Science*, *17*(2), 334–351.

Zhang, W., & Gao, J. (2021). Deep-learning full-waveform inversion using seismic migration images. *IEEE Transactions on Geoscience and Remote Sensing*, 1-18.

Zhu, W., Xu, K., Darve, E., Biondi, B., & Beroza, G. C. (2020). Integrating deep neural networks with full-waveform inversion: Reparametrization, regularization, and uncertainty quantification. *arXiv preprint arXiv:2012.11149*.

Figure 1. Flowchart of CNN-RWI. The larger red and black boxes on the right side show the detailed procedures in the smaller red and black boxes on the left side, respectively.

Figure 2. Marmousi2 P-wave velocity model. The three different portions of the Marmousi2 velocity model (in the red, black, and cyan boxes) are the three true velocity models in Tests 1 and 2.

Figure 3. U-net convolutional neural network architecture. The black layer is the input layer in which the starting velocity model and RTM image are concatenated as two channels. The red layer is the output layer which outputs the velocity model in either the CNN training (equation 13) or CNN prediction (equation 14). The down- and up-sampling are implemented by the convolution, and the transposed convolution, with stride=2, respectively.

Figure 4. Comparison of the velocity models in Test 1. (a)-(c) are the three starting velocity models a-c, smoothed by a 3×3 Gaussian filter for 400 iterations, from the true velocity models in the red, black and cyan boxes in Figure 2. (d)-(f) are the velocity models inverted by the CNN-FWI after 16 iterations. (g)-(i) are the velocity models inverted by the CG-FWI after 39 iterations, by using the CNN-RWI-inverted velocity models in (d)-(f) as the starting velocity models, respectively. (j)-(l) and (m)-(o) are the velocity models inverted by the CG-FWI after 110 iterations, based on sources with 20 Hz and 7 Hz dominant frequencies, respectively.

Figure 5. Comparison of the residual velocity models between the velocity models in Figure 4 and the true velocity models in Figure 2, respectively, in Test 1.

Figure 6. Comparison of the residual velocity models between the velocity models in Figure 4 and the starting velocity models a-c in Figure 4a-4c, respectively, except for (a)-(c), which are the residual velocity models between the CG-FWI-inverted velocity models in Pass 2 in Figure 4g-4i and the CNN-RWI-inverted velocity models in Pass 1 in Figure 4d-4f, respectively.

Figure 7. Comparison of the RMS model errors of the velocity models in Figure 4, across all positions for each depth, in Test 1. Columns a, b, and c correspond to the velocity models in in the left, middle, and right columns in Figure 4, respectively. The cyan lines correspond to the starting velocity models in Figure 4a-4c, respectively. The red, blue, and black lines correspond to the velocity models inverted by the CNN-RWI in Pass 1 in Figure 4d-4f, the CG-FWI in Pass 2 in Figure 4g-4i, and the CG-FWI in Figure 4j-4l, respectively, based on sources with 20 Hz dominant frequency. The magenta lines correspond to the velocity models inverted by the CG-FWI in Figure 4m-4o based on sources with 7 Hz dominant frequency.

Figure 8. Comparison of the RMS model (the left column) and data errors (the right column) in Test 1. The top, middle, and bottom rows correspond to the inverted velocity models, using the starting velocity models a, b, and c (Figures 4a-4c), respectively. The red, blue, and black lines in both the RMS model (the left column) and data (the right column) errors correspond to the velocity models inverted by the CNN-RWI in Pass 1, the CG-FWI in Pass 2, and the CG-FWI, respectively, based on sources with 20 Hz dominant frequency. The magenta lines in both the RMS model (the left column) and data (the right column) errors correspond to the velocity models inverted by the CG-FWI, based on sources with 7 Hz dominant frequency. The cyan dots in panels a-c correspond to the RMS errors of the training velocity models in the CNN-RWI in Pass 1, at each iteration.

Figure 9. Comparison of the velocity models in Test 2. (a)-(c) are the three starting velocity models a-c, smoothed by a 3×3 Gaussian filter for 1600 iterations, from the true velocity models in the red, black and cyan boxes in Figure 2. (d)-(f) are the velocity models inverted by the CNN-FWI after 16 iterations. (g)-(i) are the velocity models inverted by the CG-FWI after 39 iterations, by using the CNN-RWI-inverted velocity models in (d)-(f) as the starting velocity models, respectively. (j)-(l) and (m)-(o) are the velocity models inverted by the CG-FWI after 110 iterations, based on sources with 20 Hz and 7 Hz dominant frequencies, respectively.

Figure 10. Comparison of the residual velocity models between the velocity models in Figure 9 and the true velocity models in Figure 2, respectively, in Test 2.

Figure 11. Comparison of the RMS model errors of the velocity models in Figure 9, across all positions for each depth, in Test 2. Columns a, b, and c correspond to the velocity models in in the left, middle, and right columns in Figure 9, respectively. The cyan lines correspond to the starting velocity models in Figure 9a-9c, respectively. The red, blue, and black lines correspond to the velocity models inverted by the CNN-RWI in Pass 1 in Figure 9d-9f, the CG-FWI in Pass 2 in Figure 9g-9i, and the CG-FWI in Figure 9j-9l, respectively, based on sources with 20 Hz dominant frequency. The magenta lines correspond to the velocity models inverted by the CG-FWI in Figure 9m-9o based on sources with 7 Hz dominant frequency.

Figure 12. Comparison of the RMS model (the left column) and data errors (the right column) in Test 2. The top, middle, and bottom rows correspond to the inverted velocity models, using the starting velocity models a, b, and c (Figures 9a-9c), respectively. The red, blue, and black lines in both the RMS model (the left column) and data (the right column) errors correspond to the velocity models inverted by the CNN-RWI in Pass 1, the CG-FWI in Pass 2, and the CG-FWI, respectively, based on sources with 20 Hz dominant frequency. The magenta lines in both the RMS model (the left column) and data (the right column) errors correspond to the velocity models inverted by the CG-FWI, based on sources with 7 Hz dominant frequency. The cyan dots in panels a-c correspond to the RMS errors of the training velocity models in the CNN-RWI in Pass 1, at each iteration.

Figure 1.

Observed data

Original starting model

Parcellate the latest velocity model to build a model basis

Assign a random velocity to each parcel in model basis

Train CNN to fit training dataset

Original RTM image

Apply CNN to predict model

CNN-predicted model

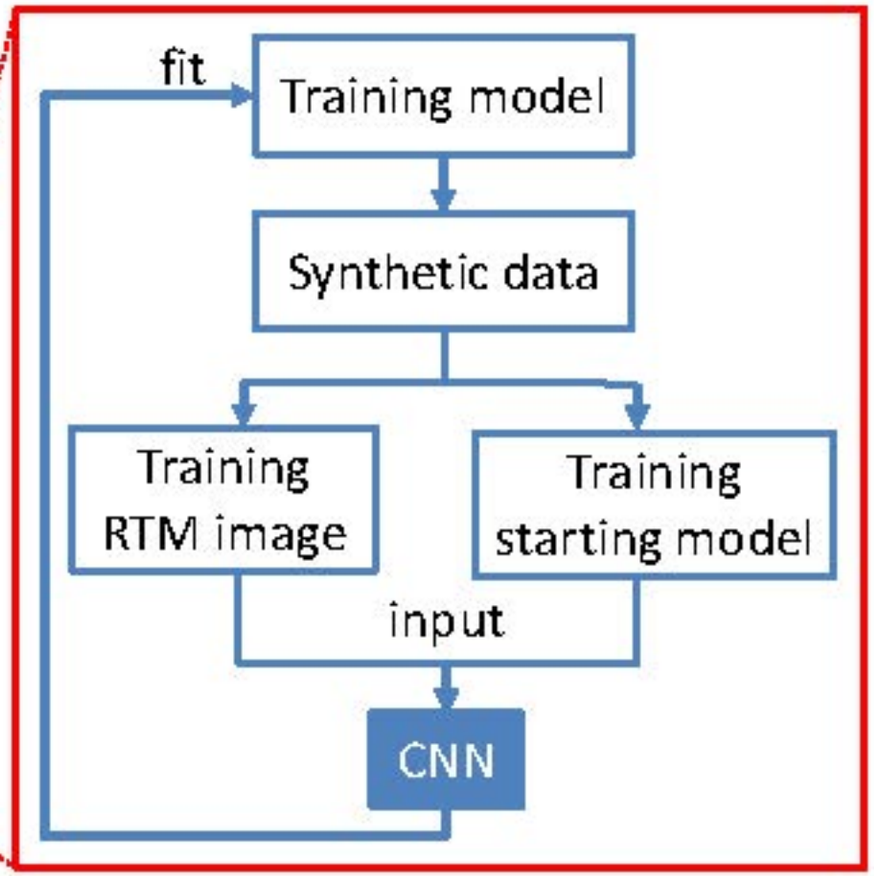
Error evaluation

Converged?

Yes

End

No



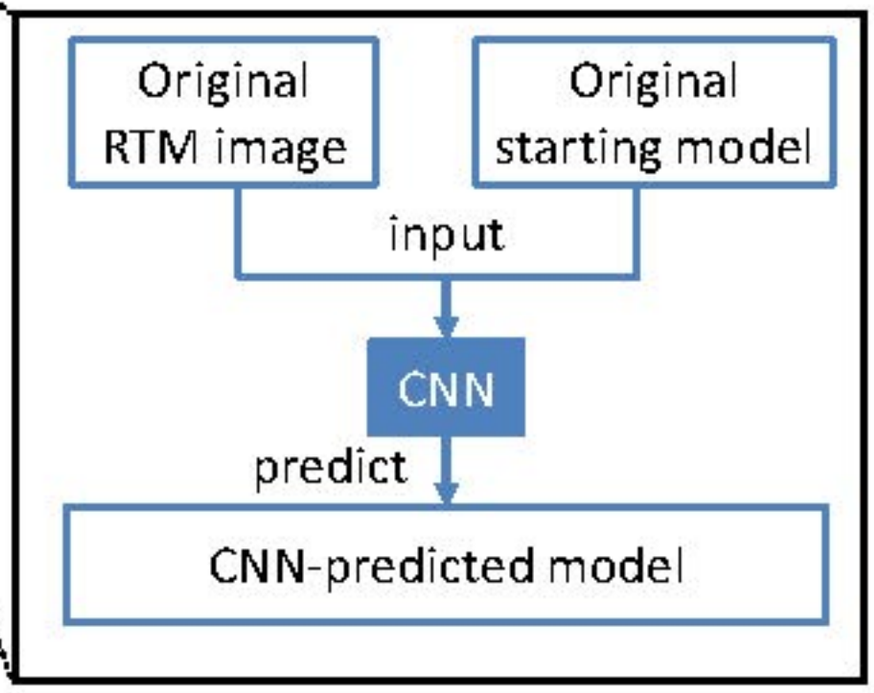


Figure 2.

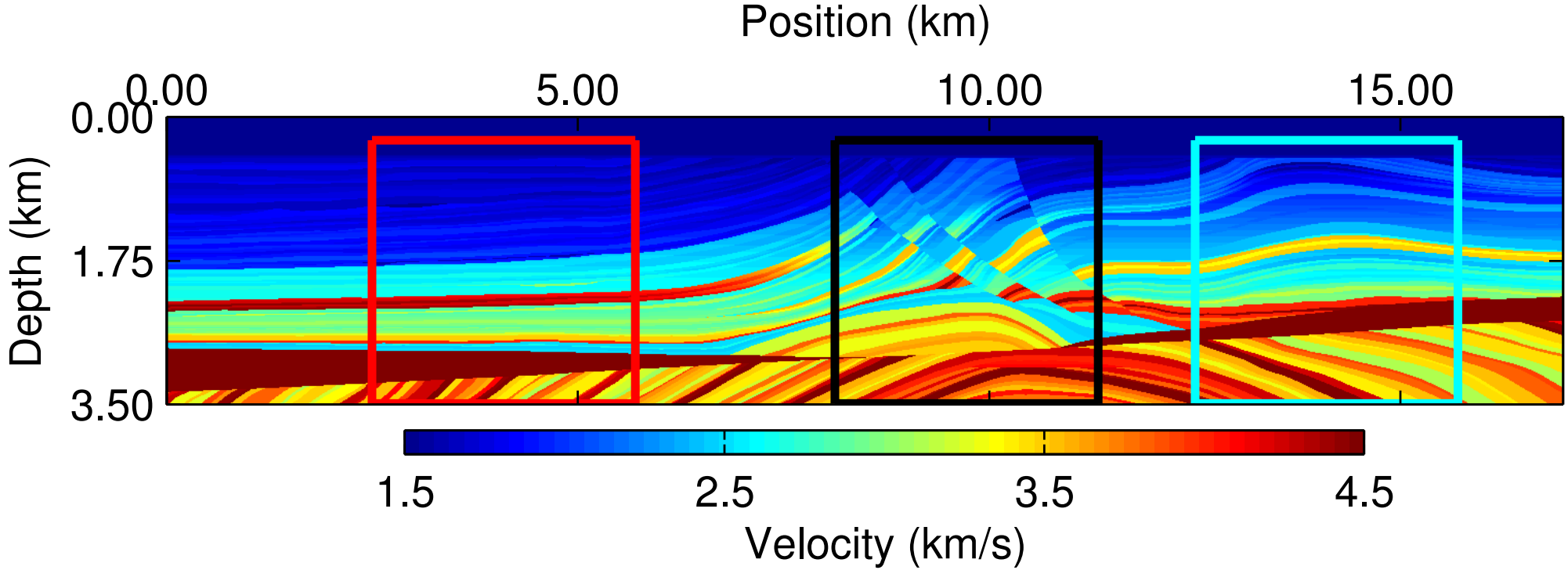

Position (km)
0.00
5.00
10.00
15.00
Depth (km)
0.00
1.75
3.50
1.5
2.5
3.5
4.5
Velocity (km/s)

Figure 3.

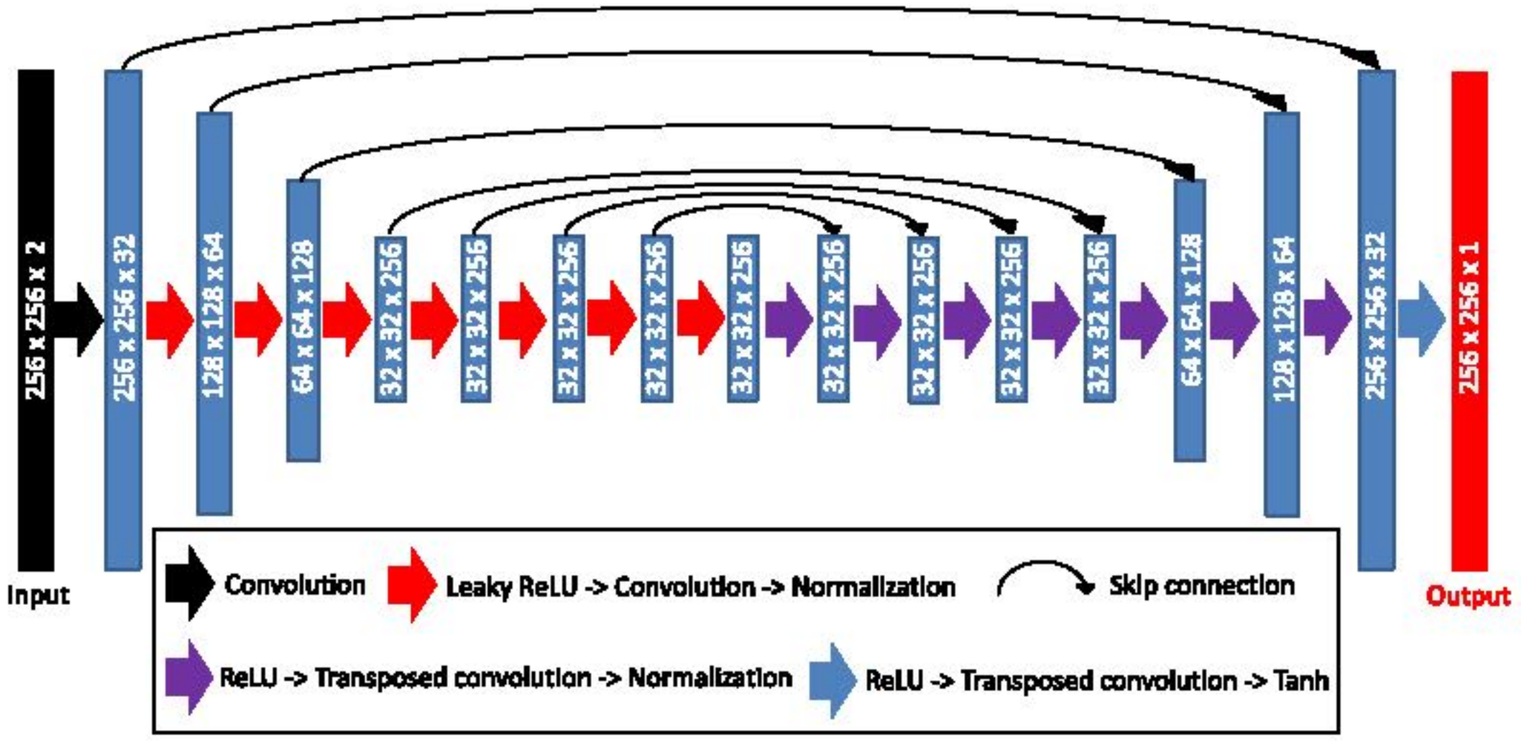

256 x 256 x 2
256 x 256 x 32
128 x 128 x 64
64 x 64 x 128
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
32 x 32 x 256
64 x 64 x 128
128 x 128 x 64
256 x 256 x 32
256 x 256 x 1
Input
Output
Convolution
Leaky ReLU -> Convolution -> Normalization
Skip connection
ReLU -> Transposed convolution -> Normalization
ReLU -> Transposed convolution -> Tanh

Figure 4.

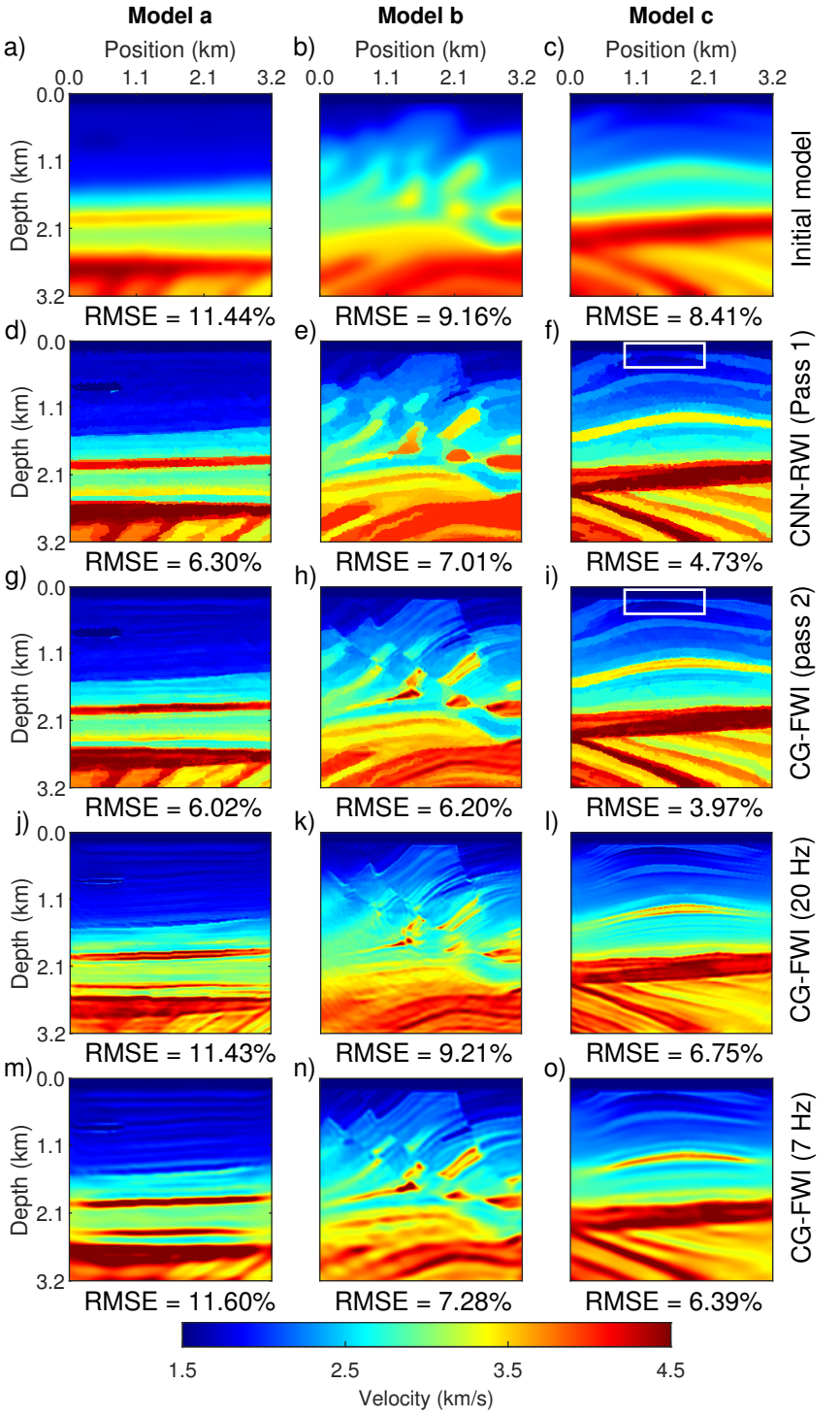

Model a
Model b
Model c
a)
b)
c)
Position (km)
0.0
1.1
2.1
3.2
Depth (km)
Initial model
RMSE = 11.44%
RMSE = 9.16%
RMSE = 8.41%
d)
e)
f)
CNN-RWI (Pass 1)
RMSE = 6.30%
RMSE = 7.01%
RMSE = 4.73%
g)
h)
i)
CG-FWI (pass 2)
RMSE = 6.02%
RMSE = 6.20%
RMSE = 3.97%
j)
k)
l)
CG-FWI (20 Hz)
RMSE = 11.43%
RMSE = 9.21%
RMSE = 6.75%
m)
n)
o)
CG-FWI (7 Hz)
RMSE = 11.60%
RMSE = 7.28%
RMSE = 6.39%
1.5
2.5
3.5
4.5
Velocity (km/s)

Figure 5.

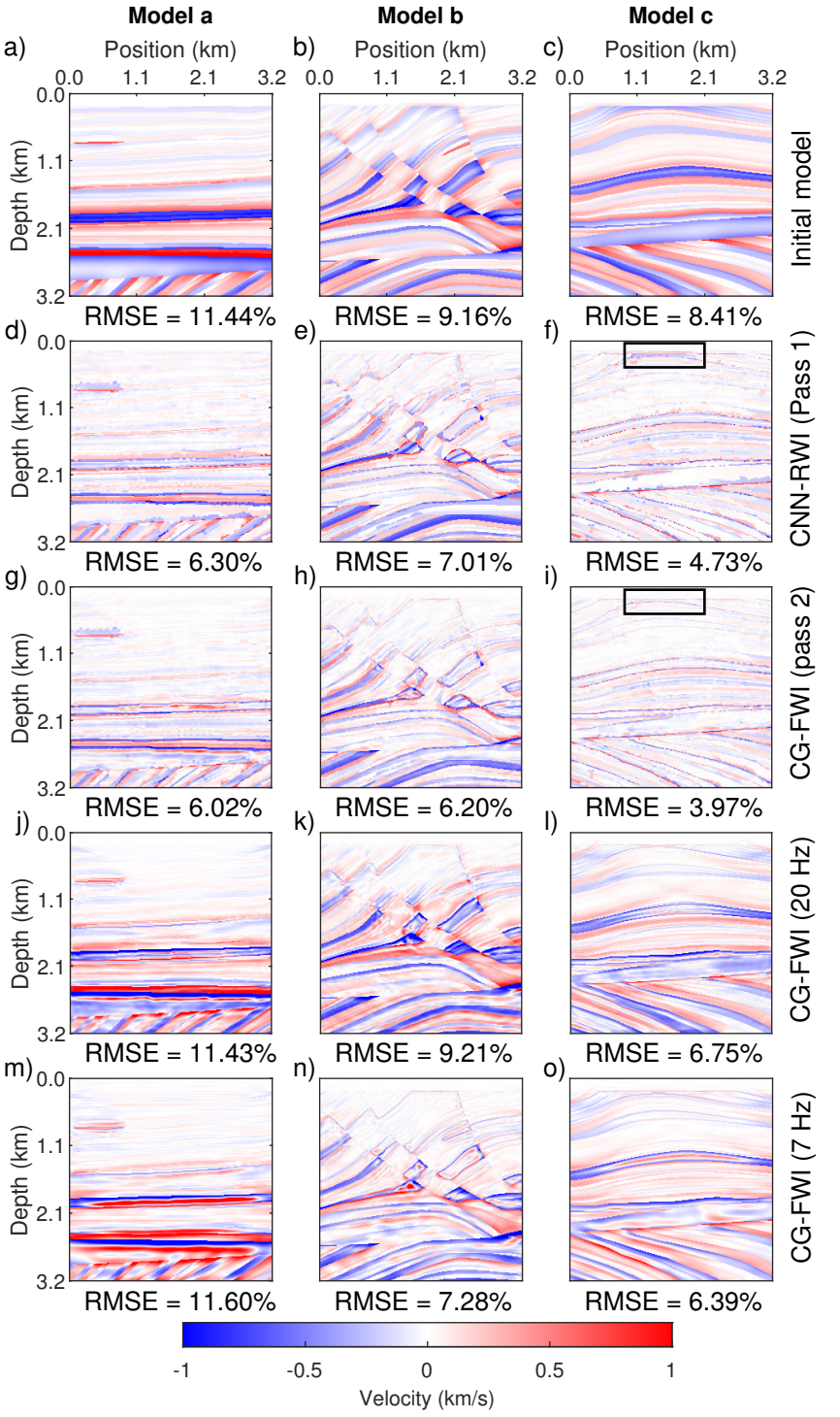

Model a
Model b
Model c
a)
b)
c)
Position (km)
0.0
1.1
2.1
3.2
Depth (km)
Initial model
RMSE = 11.44%
RMSE = 9.16%
RMSE = 8.41%
d)
e)
f)
CNN-RWI (Pass 1)
RMSE = 6.30%
RMSE = 7.01%
RMSE = 4.73%
g)
h)
i)
CG-FWI (pass 2)
RMSE = 6.02%
RMSE = 6.20%
RMSE = 3.97%
j)
k)
l)
CG-FWI (20 Hz)
RMSE = 11.43%
RMSE = 9.21%
RMSE = 6.75%
m)
n)
o)
CG-FWI (7 Hz)
RMSE = 11.60%
RMSE = 7.28%
RMSE = 6.39%
-1
-0.5
0
0.5
1
Velocity (km/s)

Figure 6.

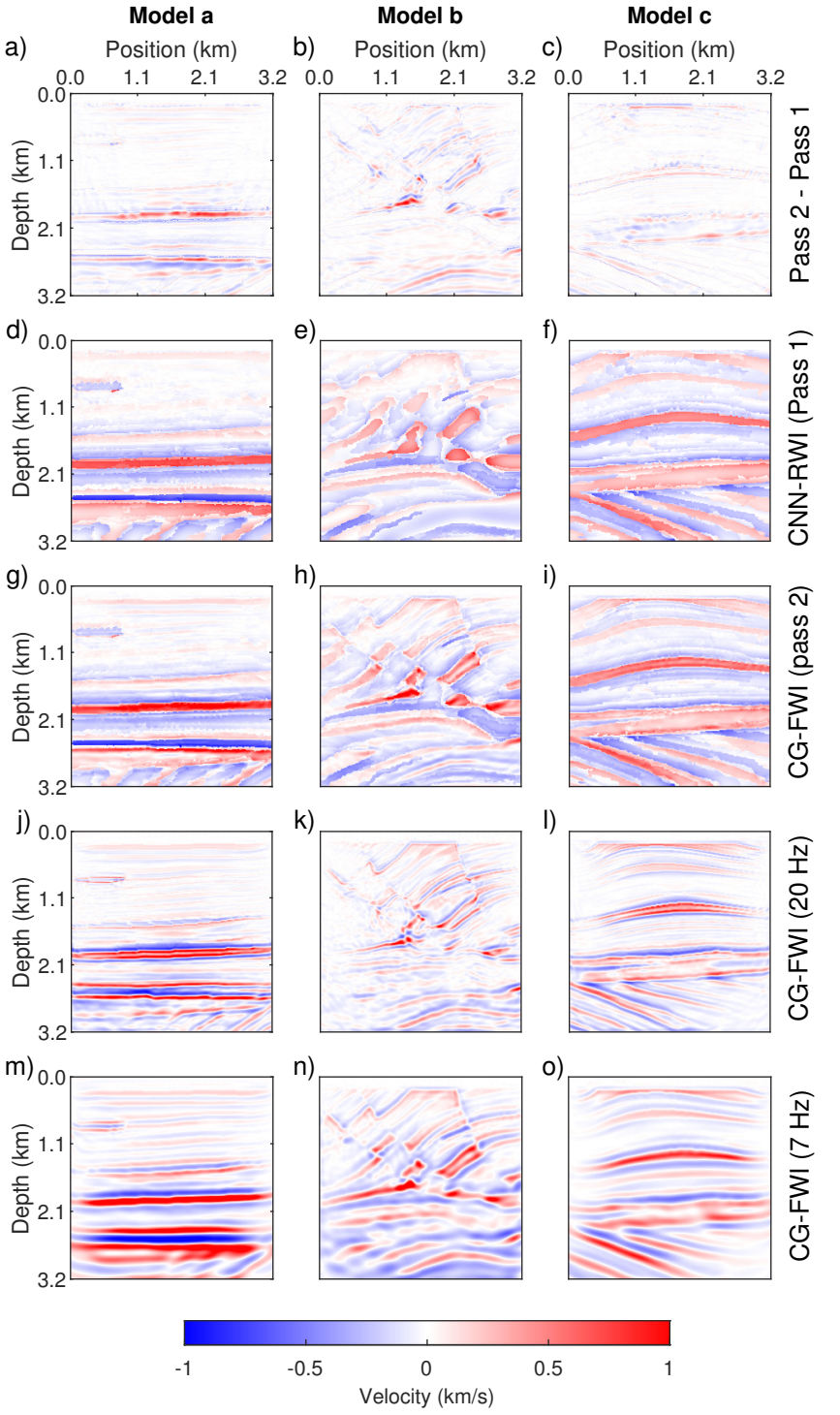

Model a
Model b
Model c
a)
b)
c)
Position (km)
0.0
1.1
2.1
3.2
Depth (km)
Pass 2 - Pass 1
d)
e)
f)
CNN-RWI (Pass 1)
g)
h)
i)
CG-FWI (pass 2)
j)
k)
l)
CG-FWI (20 Hz)
m)
n)
o)
CG-FWI (7 Hz)
-1
-0.5
0
0.5
1
Velocity (km/s)

Figure 7.

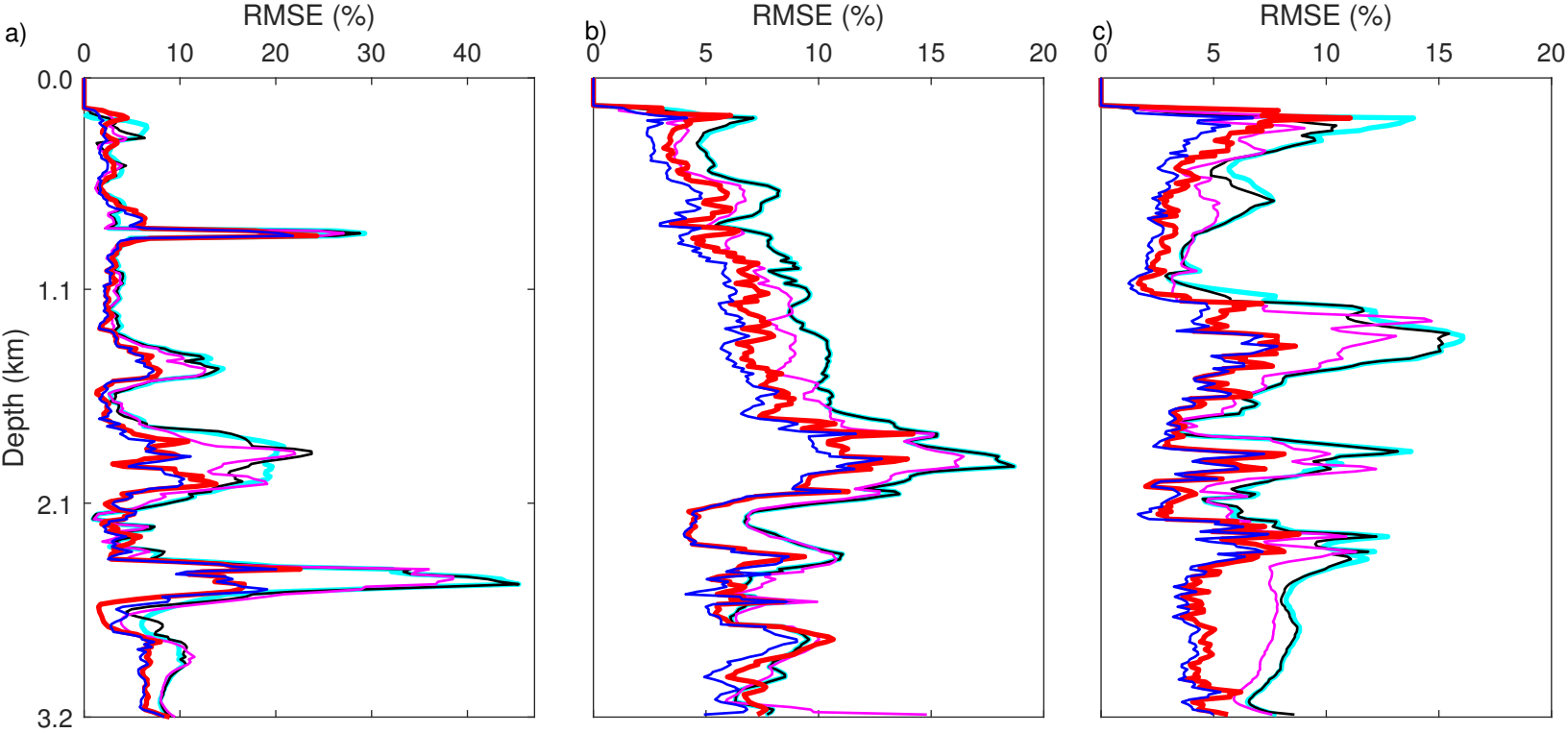

a)
RMSE (%)
0
10
20
30
40
b)
RMSE (%)
0
5
10
15
20
c)
RMSE (%)
0
5
10
15
20
Depth (km)
0.0
1.1
2.1
3.2

Figure 8.

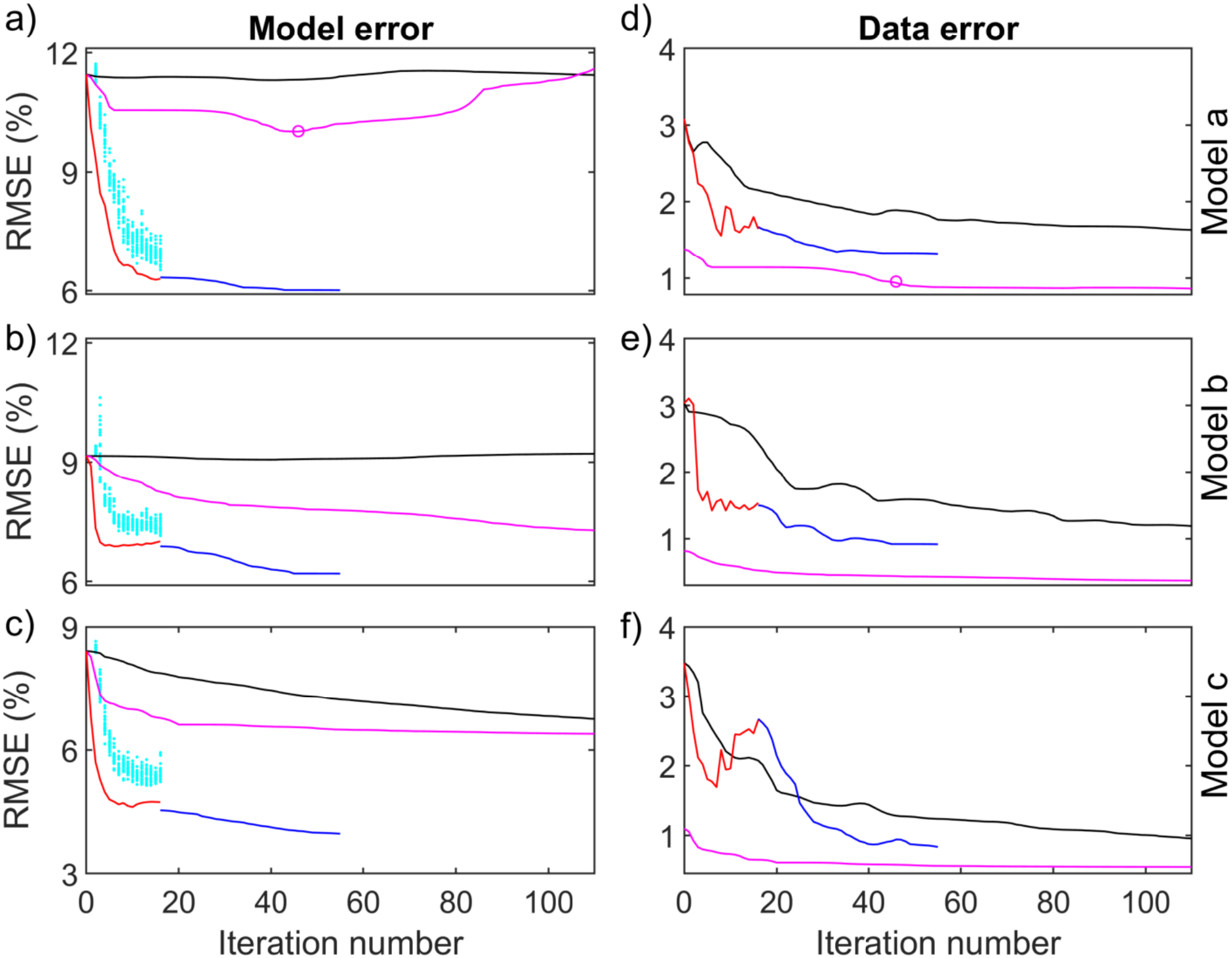

a)
Model error
d)
Data error
RMSE (%)
Model a
b)
e)
Model b
c)
f)
Model c
Iteration number
Iteration number

Figure 9.

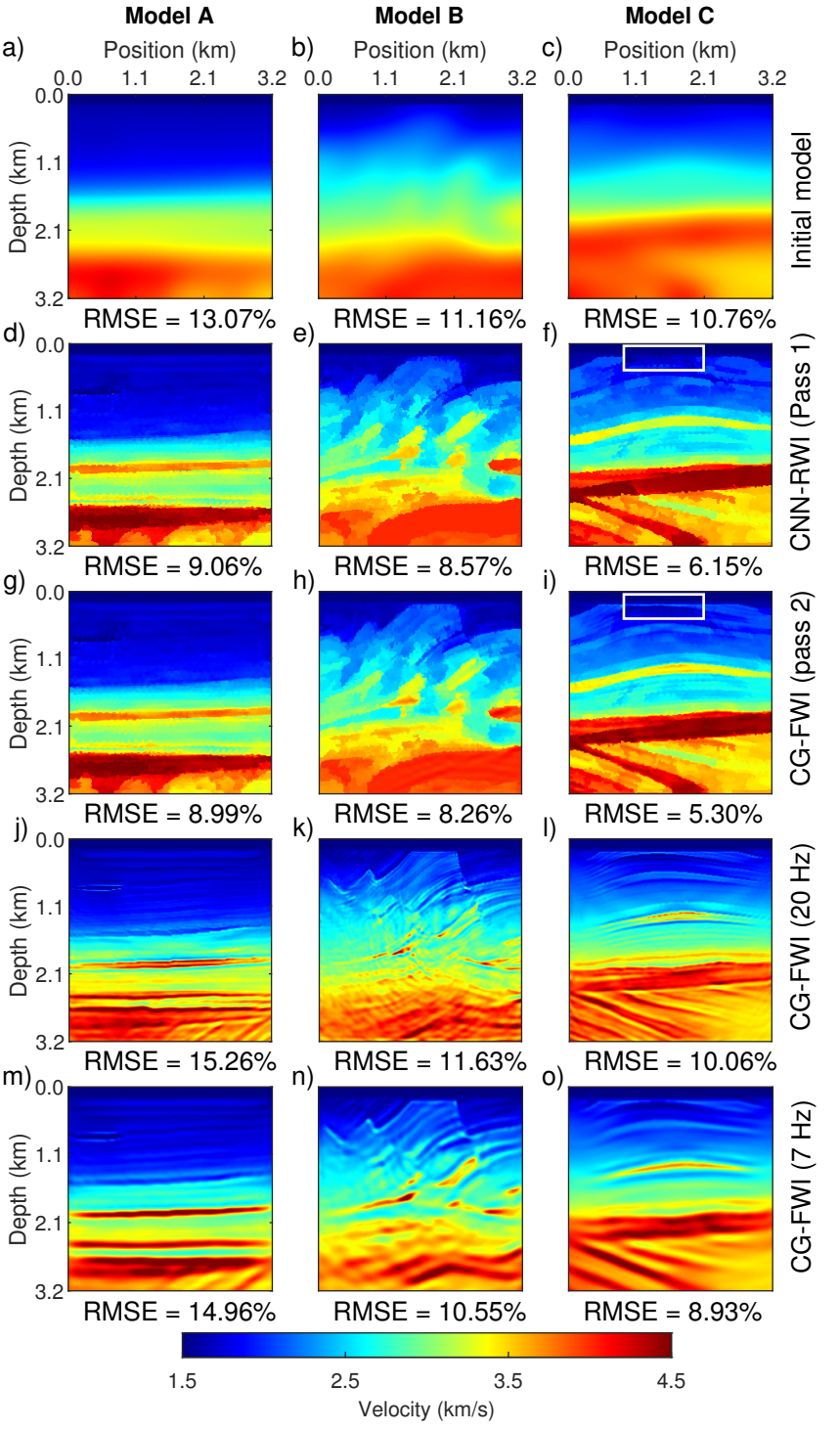

Model A
Model B
Model C
a)
b)
c)
Position (km)
0.0
1.1
2.1
3.2
Depth (km)
Initial model
RMSE = 13.07%
RMSE = 11.16%
RMSE = 10.76%
d)
e)
f)
CNN-RWI (Pass 1)
RMSE = 9.06%
RMSE = 8.57%
RMSE = 6.15%
g)
h)
i)
CG-FWI (pass 2)
RMSE = 8.99%
RMSE = 8.26%
RMSE = 5.30%
j)
k)
l)
CG-FWI (20 Hz)
RMSE = 15.26%
RMSE = 11.63%
RMSE = 10.06%
m)
n)
o)
CG-FWI (7 Hz)
RMSE = 14.96%
RMSE = 10.55%
RMSE = 8.93%
1.5
2.5
3.5
4.5
Velocity (km/s)

Figure 10.

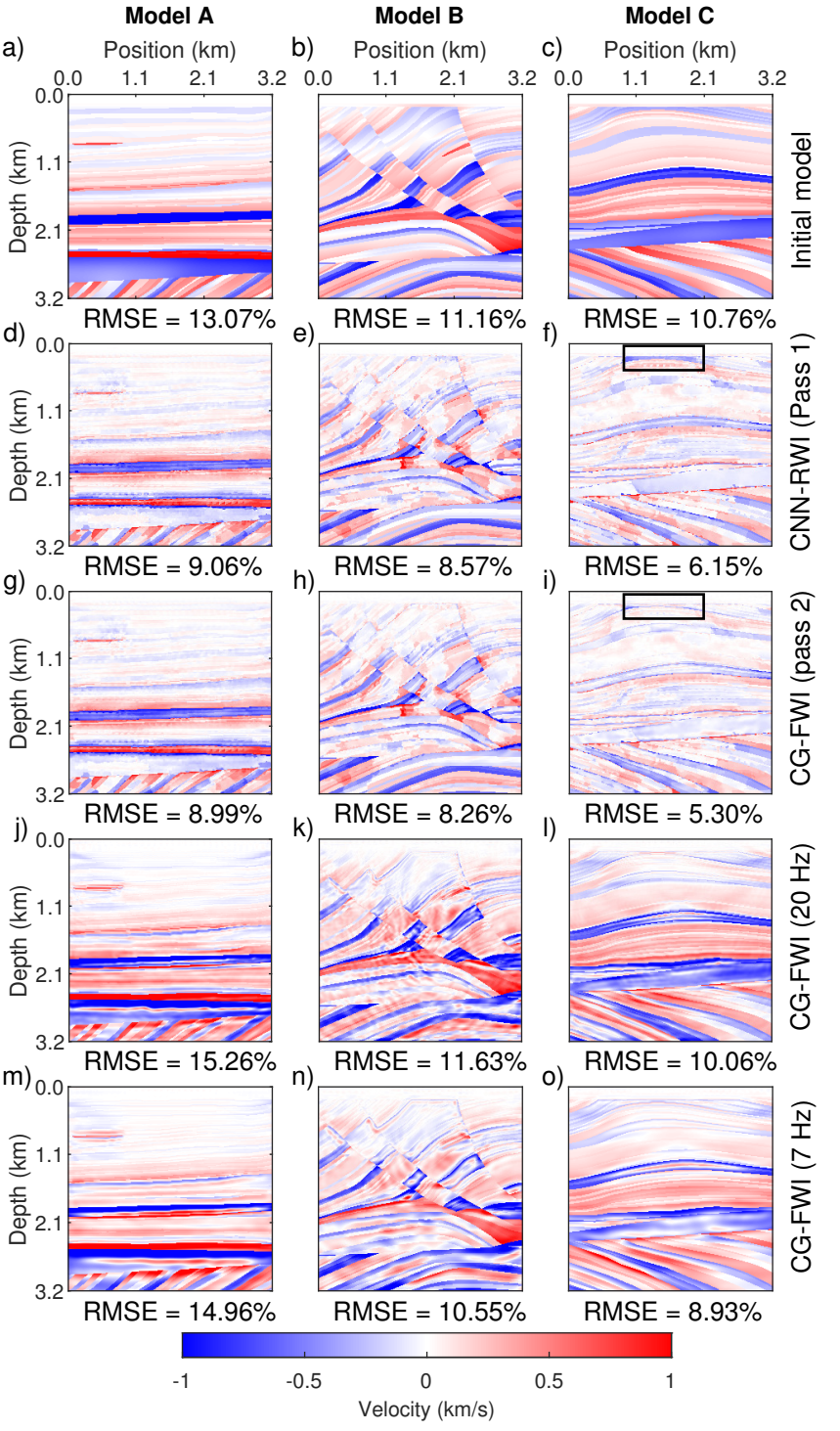

Model A
Model B
Model C
a)
b)
c)
Position (km)
0.0
1.1
2.1
3.2
Depth (km)
Initial model
RMSE = 13.07%
RMSE = 11.16%
RMSE = 10.76%
d)
e)
f)
CNN-RWI (Pass 1)
RMSE = 9.06%
RMSE = 8.57%
RMSE = 6.15%
g)
h)
i)
CG-FWI (pass 2)
RMSE = 8.99%
RMSE = 8.26%
RMSE = 5.30%
j)
k)
l)
CG-FWI (20 Hz)
RMSE = 15.26%
RMSE = 11.63%
RMSE = 10.06%
m)
n)
o)
CG-FWI (7 Hz)
RMSE = 14.96%
RMSE = 10.55%
RMSE = 8.93%
-1
-0.5
0
0.5
1
Velocity (km/s)

Figure 11.

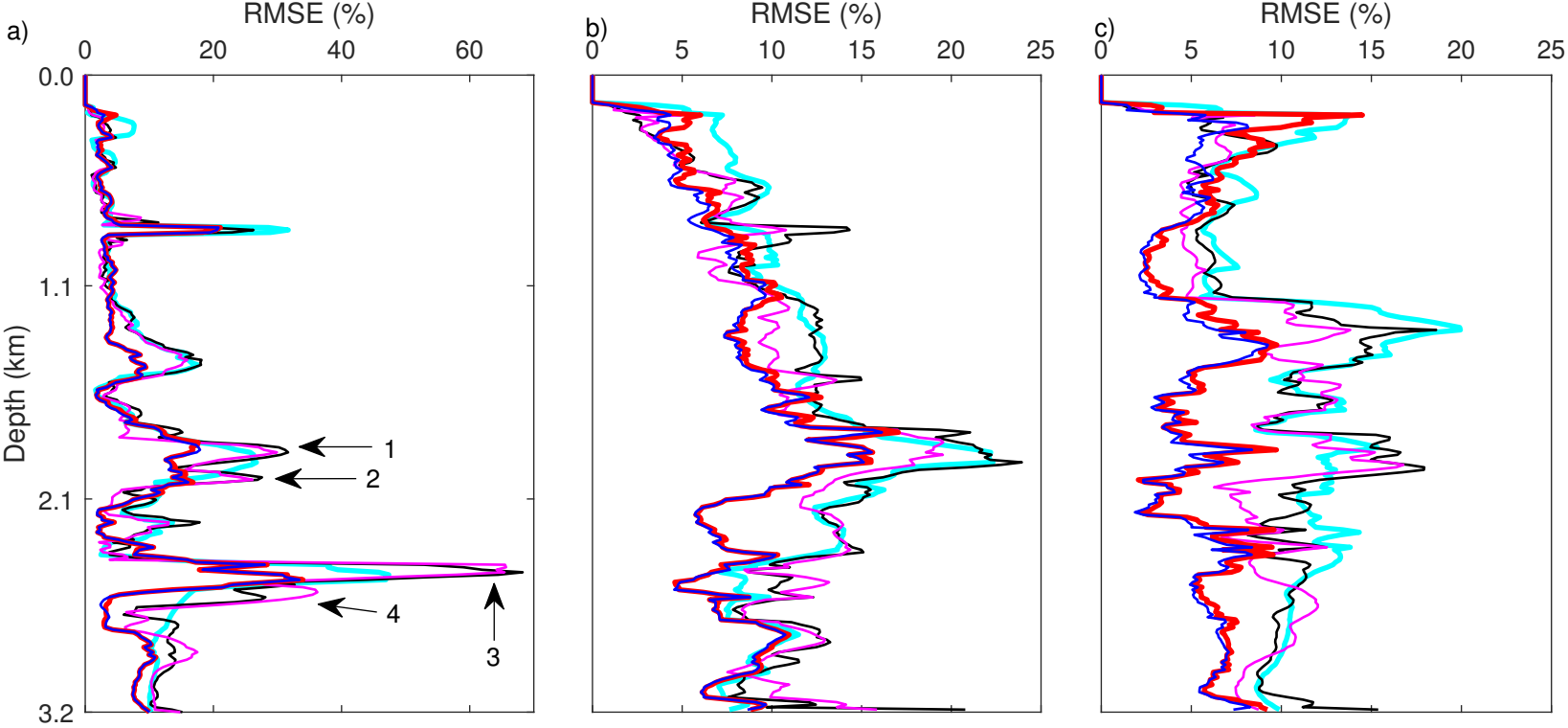
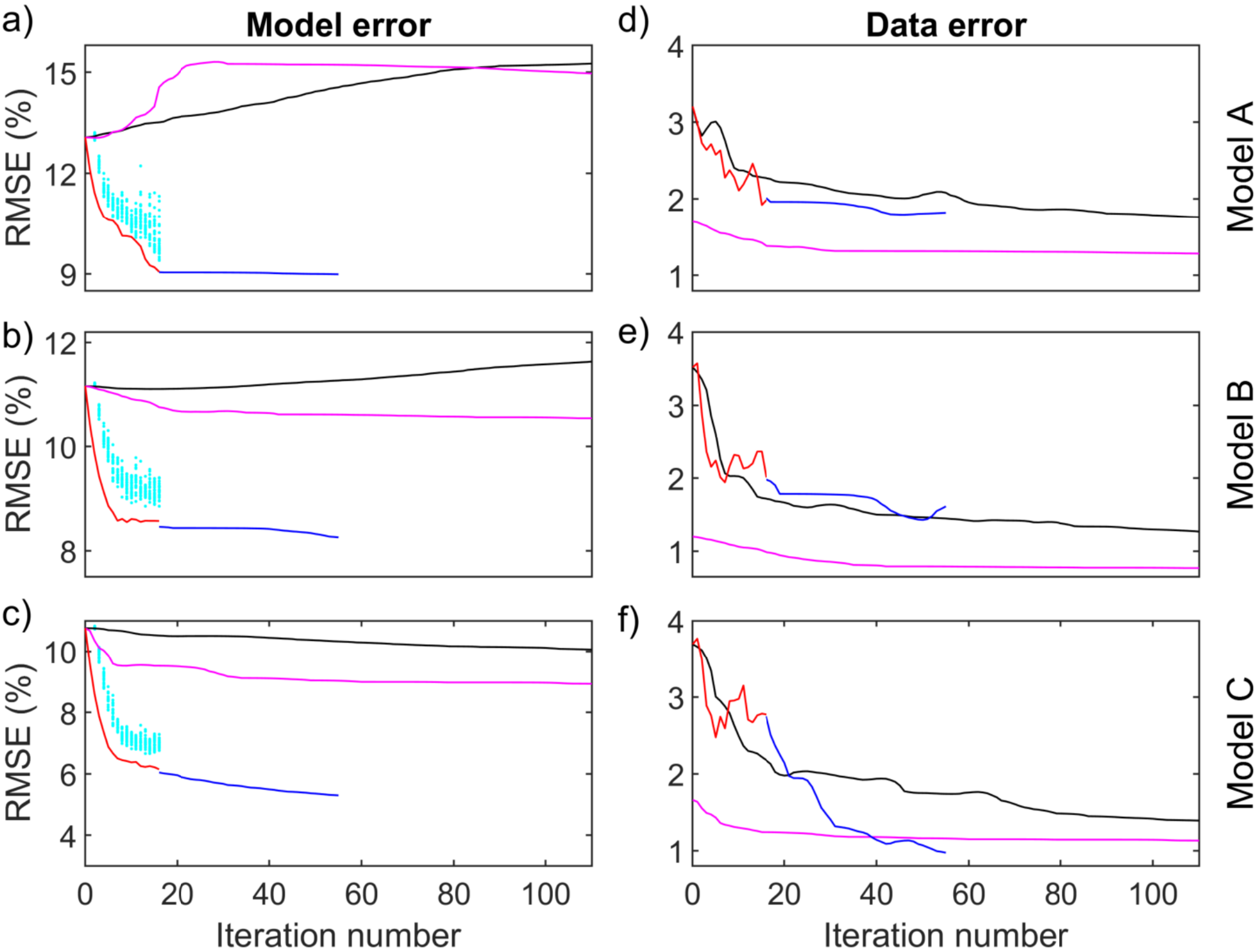

a)
RMSE (%)
0
20
40
60
0.0
1.1
2.1
3.2
Depth (km)
1
2
3
4
b)
RMSE (%)
0
5
10
15
20
25
c)
RMSE (%)
0
5
10
15
20
25

Figure 12.


a)
Model error
d)
Data error
RMSE (%)
15
12
9
4
3
2
1
Model A
b)
e)
RMSE (%)
12
10
8
4
3
2
1
Model B
c)
f)
RMSE (%)
10
8
6
4
4
3
2
1
Model C
0
20
40
60
80
100
Iteration number
Iteration number