https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

Solid Earth Discussions

Deep learning for fast simulation of seismic waves in complex media

Ben Moseley[1], Tarje Nissen-Meyer[2], and Andrew Markham[1]
[1]Department of Computer Science, University of Oxford, UK
[2]Department of Earth Sciences, University of Oxford, UK
Correspondence: Ben Moseley (bmoseley@robots.ox.ac.uk)

Abstract. The simulation of seismic waves is a core task in many geophysical applications. Numerical methods such as Finite Difference (FD) modelling and Spectral Element Methods (SEM) are the most popular techniques for simulating seismic waves in complex media, but for many tasks their computational cost is prohibitively expensive. In this work we present two types of deep neural networks as fast alternatives for simulating seismic waves in horizontally layered and faulted 2D acoustic media. In contrast to the classical methods both networks are able to simulate the seismic response at multiple locations within the media in a single inference step, without needing to iteratively model the seismic wavefield through time, resulting in an order of magnitude reduction in simulation time. This speed improvement could pave the way to real-time seismic simulation and benefit seismic inversion algorithms based on forward modelling, such as full waveform inversion. Our first network is able to simulate seismic waves in horizontally layered media. We use a WaveNet network architecture and show this is more accurate than a standard convolutional network design. Furthermore we show that seismic inversion can be carried out by retraining the network with its inputs and outputs reversed, offering a fast alternative to existing inversion techniques. Our second network is significantly more general than the first; it is able to simulate seismic waves in faulted media with arbitrary layers, fault properties and an arbitrary location of the seismic source on the surface of the media. It uses a convolutional autoencoder network design and is conditioned on the input source location. We investigate the sensitivity of different network designs and training hyperparameters on its simulation accuracy. We compare and contrast this network to the first network. To train both networks we introduce a time-dependent gain in the loss function which improves convergence. We discuss the relative merits of our approach with FD modelling and how our approach could be generalised to simulate more complex Earth models.

1 Introduction

Seismic simulations are essential for many areas of geophysics. In seismic hazards analysis, they are a key tool for quantifying the ground motion of potential earthquakes (Cui et al., 2010). In oil and gas prospecting, they allow the seismic response of hydrocarbon reservoirs to be modelled (Lubrano-Lavadera et al., 2017; Siddiqui et al., 2017). In geophysical surveying, they show how the subsurface is illuminated by different survey designs (Xie et al., 2006). Seismic simulations are used in global geophysics to obtain snapshots of the Earth's interior dynamics (Hosseini et al., 2019) and to decipher source and path effects from individual seismograms (Krischer et al., 2017). They are also heavily used in seismic inversion, which estimates the elastic properties of a medium given its seismic response (Schuster, 2017). In Full Waveform Inversion (FWI), a strategy widespread in the field of seismic imaging, simulations are used thousands of times to iteratively estimate a medium's elastic properties (Virieux et al., 2017).

Numerous methods exist for simulating seismic waves (Igel, 2016). The most popular are Finite Difference (FD) modelling and Spectral Element Methods (SEM) (Moczo et al., 2007; Komatitsch and Tromp, 1999). They are able to capture a large range of physics, including the effects of undulating solid-fluid interfaces (Leng et al., 2019), intrinsic attenuation (van Driel and Nissen-Meyer, 2014a) and anisotropy (van Driel and Nissen-Meyer, 2014b). These methods solve for the propagation of the full seismic wavefield by discretising the elastodynamic equations of motion. For an acoustic heterogeneous medium these are given by the scalar linear equation of motion

$$\rho\nabla\cdot\left(\frac{1}{\rho}\nabla p\right)-\frac{1}{v^2}\frac{\partial^2 p}{\partial t^2}=-\rho\frac{\partial^2 f}{\partial t^2}\ , \qquad (1)$$

where p is the acoustic pressure, f is a point source of volume injection (the seismic source), and $v=\sqrt{\kappa/\rho}$ is the velocity of the medium, with ρ the density of the medium and κ the adiabatic compression modulus (Long et al., 2013).

Whilst FD and spectral element methods are the primary means to simulate seismic waves in complex media, a major disadvantage of these methods is their computational cost. FD modelling can involve millions of grid points and at each time step the wavefield must be iteratively updated at each grid point. This computational cost is often prohibitively expensive; supercomputers are typically required for large simulations (Leng et al., 2016). A faster method for seismic simulation would enable many applications. For example, it would benefit real-time seismic simulation and seismic inversion methods which are heavily limited by the computational cost of forward simulation (Bohlen, 2002).

The field of deep learning has recently shown promise in its ability to make approximate yet sufficiently accurate predictions of physical phenomena. These approaches are able to learn about highly non-linear physics and often offer much faster inference times than traditional simulation (Guo et al., 2016; Perol et al., 2018). In this work we study the ability of deep neural networks for simulating seismic waves in 2D acoustic media. Our contribution is as follows;

- We present two deep neural networks which are able to simulate seismic waves in 2D acoustic media. Both networks are an order of magnitude faster than FD simulation. Our first network uses a WaveNet network architecture (van den Oord et al., 2016) and simulates the pressure response from a fixed point source at multiple locations in a horizontally layered velocity model. We show that this network design is more accurate than a standard convolutional neural network. We also show that seismic inversion can be carried out by retraining the network with its inputs and outputs reversed, offering a fast alternative to existing inversion techniques, at least for layered media.

- Our second network is significantly more general than the first; it uses a conditional autoencoder network design and it is able to simulate seismic waves in faulted media with arbitrary layers, fault properties and an arbitrary location of the source on the surface of the media. The network is conditioned on the input source location. We compare this network with our WaveNet network. We also investigate the sensitivity of its simulation accuracy with different network designs and training hyperparameters.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

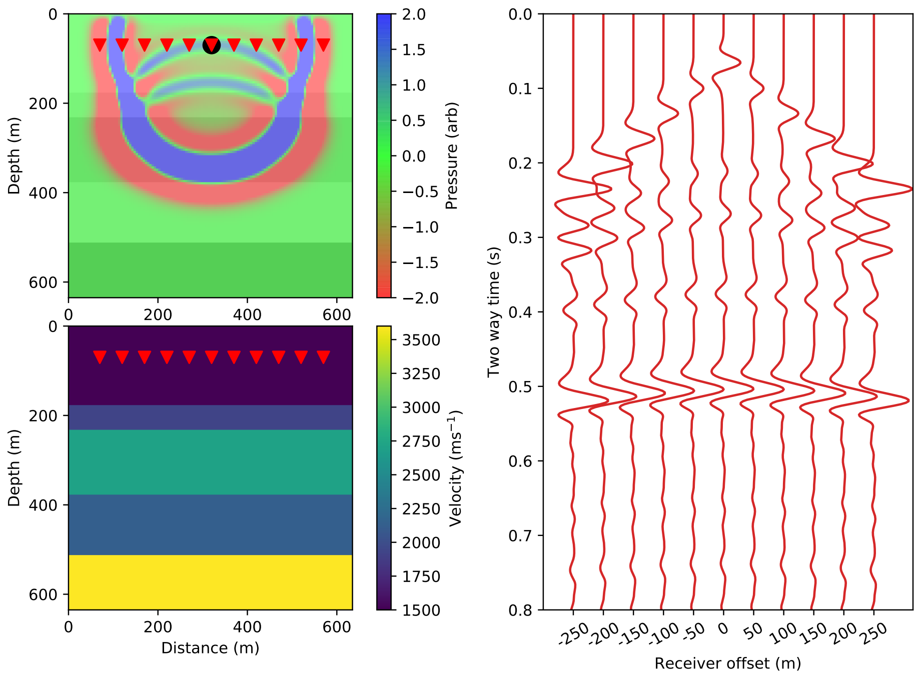

Figure 1. Ground truth FD simulation example. Left, top: A 20 Hz Ricker seismic source is emitted close to the surface and propagates through a 2D horizontally layered acoustic Earth model. The black circle shows the source location. 11 receivers are placed at the same depth as the source with a horizontal spacing of 50 m (red triangles). The full wavefield is overlain for a single snapshot in time. Note seismic reflections occur at each velocity interface. Left, bottom: The Earth velocity model. The Earth model has a constant density of 2200 kgm^{-2}. Right: The resulting ground truth pressure response recorded by each of the receivers, using FD modelling. A $t^{2.5}$ gain is applied to the receiver responses for display.

– For both networks we present a loss function with a time-varying gain which improves training convergence. Finally, we discuss the relative merits of our approach with FD modelling how our approach could be generalised to simulate more complex Earth models.

1.1 Related Work

Applying deep learning to physics problems is a burgeoning field of research and there is much active work in this area. Lerer et al. (2016) presented a deep convolutional network which could accurately predict whether randomly stacked wooden towers would fall or remain stable, given 2D images of the tower. Guo et al. (2016) demonstrated that convolutional neural networks could estimate flow fields in complex Computational Fluid Dynamics (CFD) calculations two orders of magnitude faster than a traditional GPU-accelerated CFD solver.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

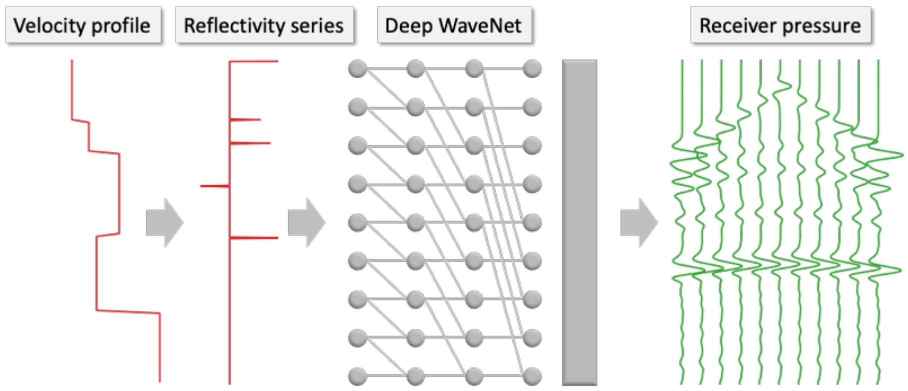

Figure 2. Our WaveNet simulation workflow. Given a 1D Earth velocity profile as input (left), our WaveNet deep neural network (middle) outputs a simulation of the pressure responses at the 11 receiver locations in Fig 1. The raw input 1D velocity profile sampled in depth is converted into its normal incidence reflectivity series sampled in time before being input into the network. The network is composed of 9 time-dilated causally-connected convolutional layers with a filter width of 2 and dilation rates which increase exponentially with layer depth. Each hidden layer of the network has same length as the input reflectivity series, 256 channels and a ReLU activation function. A final causally-connected convolutional layer with a filter width of 101 samples, 11 output channels and an identity activation is used to generate the output simulation.

Geophysicists are also starting to use deep learning for seismic-related problems. Perol et al. (2018) presented an earthquake identification method using convolutional networks which is orders of magnitude faster than traditional techniques. In seismic inversion, Araya-Polo et al. (2018) proposed an efficient deep learning concept for carrying out seismic tomography using the semblance of common mid-point receiver gathers as input. Wu et al. (2018) proposed a convolutional autoencoder network to carry out seismic inversion, whilst Yang and Ma (2019) adapted a U-net network design for the same purpose. Richardson (2018) demonstrated that a recurrent neural network framework can be used to carry out FWI. Sun and Demanet (2018) showed a method for using deep learning to extrapolate low frequency seismic energy to improve the convergence of FWI algorithms.

For seismic simulation Zhu et al. (2017) presented a multi-scale convolutional network for predicting the evolution of the full seismic wavefield in heterogeneous media. Their method was able to approximate wavefield kinematics over multiple time steps, although it suffered from the accumulation of error over time and did not offer a reduction in computational time. Moseley et al. (2018) showed that a convolutional network with a recursive loss function can simulate the full wavefield in horizontally layered acoustic media. Krischer and Fichtner (2017) used a generative adversarial network to simulate seismograms from radially symmetric and smooth Earth models. In this work we present a fast method for simulating seismic waves in faulted and arbitrarily layered 2D acoustic media.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

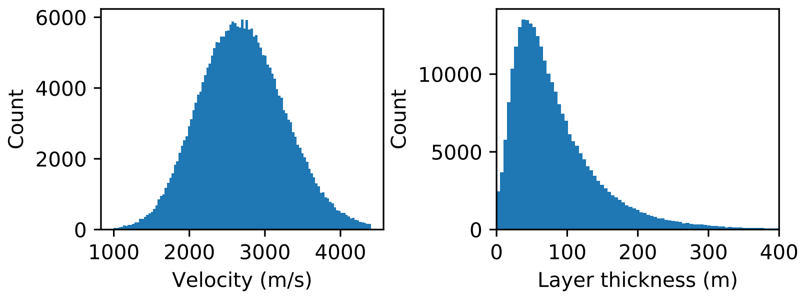

Figure 3. Distribution of layer velocity and layer thickness over all examples in the training set.

2 Fast seismic simulation in 2D horizontally layered acoustic media using WaveNet

In this section we consider simulating seismic waves in horizontally layered 2D acoustic Earth models. We train a deep neural network with a WaveNet architecture to simulate the seismic response recorded at multiple receiver locations in the Earth model, horizontally offset from a point source emitted at the surface of the model. An example simulation we wish to learn is shown in Fig. 1.

Our simulation workflow is shown in Fig. 2. The input to the network is a horizontally layered velocity profile and the output of the network is a simulation of the pressure response recorded at each of receiver location. We will now discuss deep neural networks, our WaveNet architecture, our simulation workflow and our training methodology in more detail below.

2.1 Deep neural networks and the WaveNet network

A neural network is a network of simple computational elements, known as neurons, which perform mathematical operations on multidimensional arrays, or tensors. Each neuron has a set of free parameters, or weights, which are tuned using optimisation techniques such that the network can learn a function of its inputs. In deep learning, these neurons are typically arranged in layers, which allows the network to learn highly non-linear functions.

A standard building block in deep learning is the convolutional layer, where all neurons in the layer share the same weight tensor and each neuron has a limited field of view of its input tensor. The output of the layer is mathematically achieved by cross correlating the weight tensor with the input tensor. Multiple weight tensors, or filters, can be used to increase the depth of the output layer. Such networks have achieved state of the art performance across a wide range of machine learning tasks.

The WaveNet network (van den Oord et al., 2016) proposed multiple alterations to the standard convolutional layer for its use with time series. Each convolutional layer is made causal; that is, the receptive field of each neuron only contains samples from the input layer whose sample times are before or the same as the current neuron's sample time. Furthermore the WaveNet exponentially dilates its causal connections with layer depth. This allows the field of view of its neurons to increase exponentially with layer depth, without increasing the number of weights in the network. These modifications are made to honour time series prediction tasks which are causal and to better model input data which varies over multiple time scales. The WaveNet network recently achieved state of the art performance in text to speech synthesis.

2.2 Simulation workflow

Our workflow consists of a preprocessing step, where we convert each input velocity model into its corresponding normal incidence reflectivity series sampled in time (Fig. 2, left), followed by a simulation step, where we use the WaveNet network to simulate the pressure response recorded by each receiver (Fig. 2, middle).

We chose to convert the velocity model to its corresponding normal incidence reflectivity series because, for the case of horizontally layered Earth models, the normal incidence reflectivity series and the pressure responses are causally correlated. The reflectivity series is typically used in exploration seismology (Russell, 1988) and contains values of the ratio of the amplitude of the reflected wave to the incident wave for each interface in a velocity model. For waves at normal incidence, these values are given by

$$R = \frac{\rho_2 v_2 - \rho_1 v_1}{\rho_2 v_2 + \rho_1 v_1} \,, \tag{2}$$

where ρ_1,v_1 and ρ_2,v_2 are the densities and velocities across the interface. The series is usually expressed in time and each reflectivity value occurs at the time at which the primary reflection of the source from the corresponding velocity interface arrives at a given receiver. These arrival times can be computed by carrying out a depth-to-time conversion of the reflectivity values using the input velocity model.

For horizontally layered velocity models and receivers horizontally offset from the source, the receiver pressure recordings are causally correlated to the normal incidence reflectively series of the zero-offset receiver. Intuitively, a seismic reflection recorded after a short time has only travelled through a shallow part of the velocity model and the pressure responses are at most dependent on the past samples in this reflectivity series.

We constrain our workflow such that it honours this causal correlation, by preprocessing the input velocity model into its corresponding reflectivity series and by using the causal WaveNet network to simulate the receiver response. We input the 1D profile of a 2D horizontally layered velocity model, with a depth of 640 m and a step size of 5 m. We use Eq. 2 and a standard 1D depth to time conversion to convert the velocity model into its normal incidence reflectivity series. The output reflectivity series has a length of 1 s and a sample rate of 2 ms. An example output reflectivity series is shown in Fig. 2 (left).

This reflectivity series is then passed to our WaveNet network, which contains 9 causally-connected convolutional layers (Fig. 2, middle). Each convolutional layer has the same length as the input reflectivity series, 256 hidden channels, a receptive field width of 2 samples and a Rectified Linear Unit (ReLU) activation function (Nair and Hinton, 2010). Similar to the original WaveNet design, we use exponentially increasing dilations at each layer to ensure that the first sample in the input reflectivity series is in the receptive field of the last sample of the output simulation. We add a final causally-connected convolutional layer with 11 output channels, a filter width of 101 samples and an identity activation to generate the output simulation, where each output channel corresponds to a receiver prediction. This results in the network having 1,333,515 free parameters in total.

2.3 Training data generation

To train the network, we generate 50,000 synthetic ground truth example simulations using the SEISMIC_CPML code, which performs 2$^{\mathrm{nd}}$-order acoustic FD modelling (Komatitsch and Martin, 2007). Each example simulation uses a randomly sampled 2D horizontally layered velocity model with a width and depth of 640 m and a sample rate of 5 m in both directions. (Fig. 1, bottom left). For all simulations we use a constant density model of 2200 kgm^{-2}.

In each simulation the layer velocities and layer thickness are randomly sampled from log-normal distributions. We also add a small velocity gradient randomly sampled from a normal distribution to each model such that the velocity values tend to increase with depth, to be more Earth-realistic. The distributions over layer velocities and layer thicknesses for the entire training set are shown in Fig. 3.

We use a 20 Hz Ricker source emitted close to the surface and record the pressure response at 11 receiver locations placed symmetrically around the source, horizontally offset every 50 m (Fig. 1, top left). We use a convolutional perfectly matched layer boundary condition such that waves which reach the edge of the model are absorbed with negligible reflection. We run each simulation for 1 s and use a 0.5 ms sample rate to maintain accurate FD fidelity. We downsample the resulting receiver pressure responses to 2 ms before using them for training.

We run 50,000 simulations and extract a training example from each simulation, where each training example consists of a 1D layered velocity profile and the recorded pressure response at each of the 11 receivers. We withhold 10,000 of these examples as a validation set to measure the generalisation performance of our network during training.

2.4 Training process

The network is trained using the Adam stochastic gradient descent algorithm (Kingma and Ba, 2014). This algorithm computes the gradient of a loss function with respect to the free parameters of the network over a randomly selected subset, or batch, of the training examples. This gradient is used to iteratively update the parameter values, with a step size controlled by a learning rate parameter. We use a L2 loss function with time-varying gain function, given by

$$L = \frac{1}{N} \| G(\hat{Y} - Y) \|_2^2 \, , \tag{3}$$

where $\hat{Y}$ is the simulated receiver pressure response from the network, Y is the ground truth receiver pressure response from FD modelling and N is the number of training examples in each batch. The gain function G has the form $G = t^g$ where t is the sample time and g is a hyperparameter which determines the strength of the gain. We add this to empirically account for the spherical spreading of the wavefield by increasing the weight of samples at later times. In this Section we use a fixed value of $g = 2.5$. We use a learning rate of $1\mathrm{x}10^{-5}$, a batch size of 20 training examples and run training over 500,000 steps.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

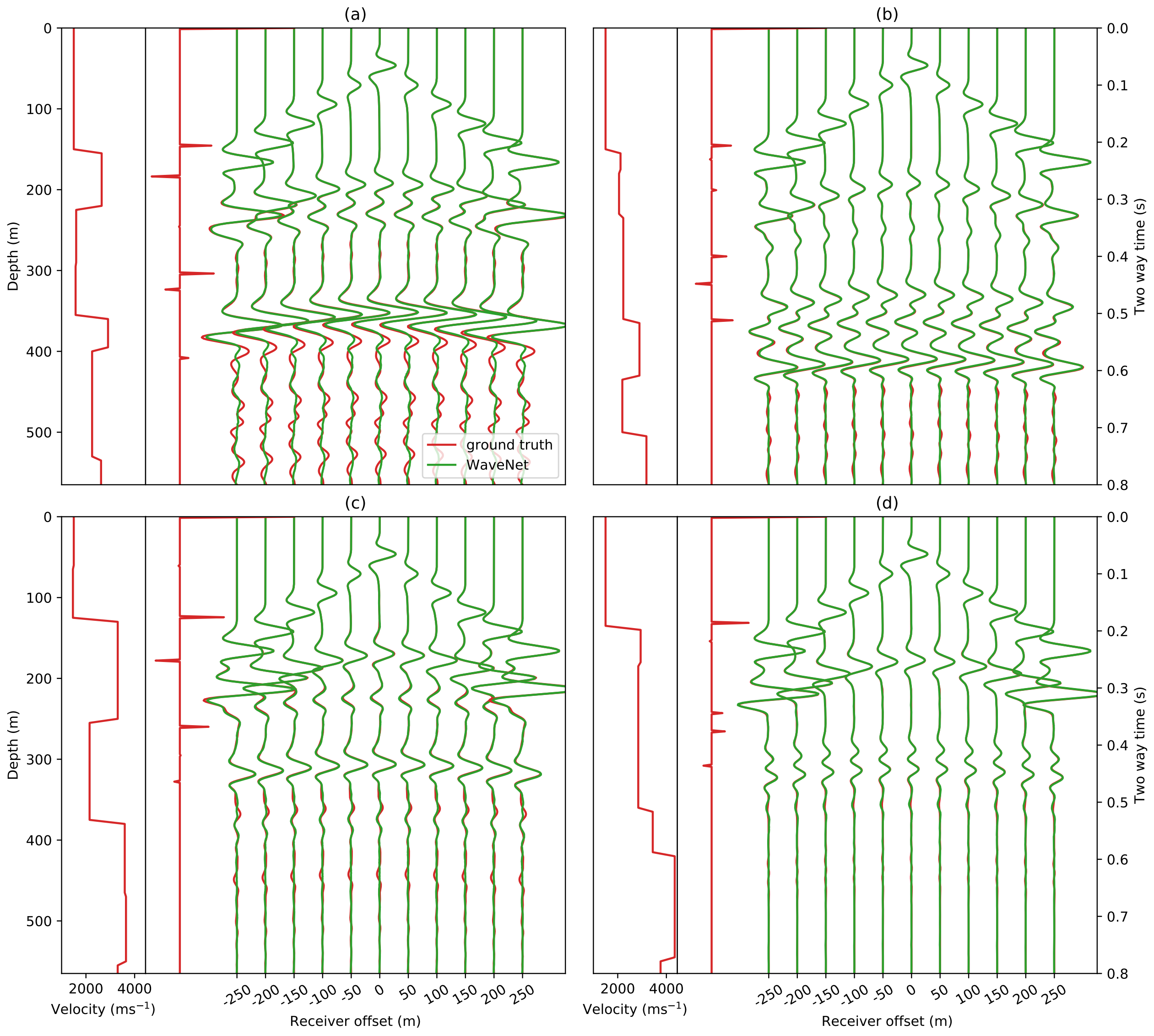

Figure 4. WaveNet simulations for 4 randomly selected examples in the test set. Red shows the input velocity model, its corresponding reflectivity series and the ground truth pressure response from FD simulation at the 11 receiver locations. Green shows the WaveNet simulation given the input reflectivity series for each example. A $t^{2.5}$ gain is applied to the receiver responses for display.

2.5 Results

During training the loss over the training and validation datasets converge to similar values, suggesting the network is generalising well to examples in the validation dataset. To assess the performance of the trained network, we generate a random test set of 1000 unseen examples. The WaveNet simulations for 4 randomly selected examples from this test set are compared to

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

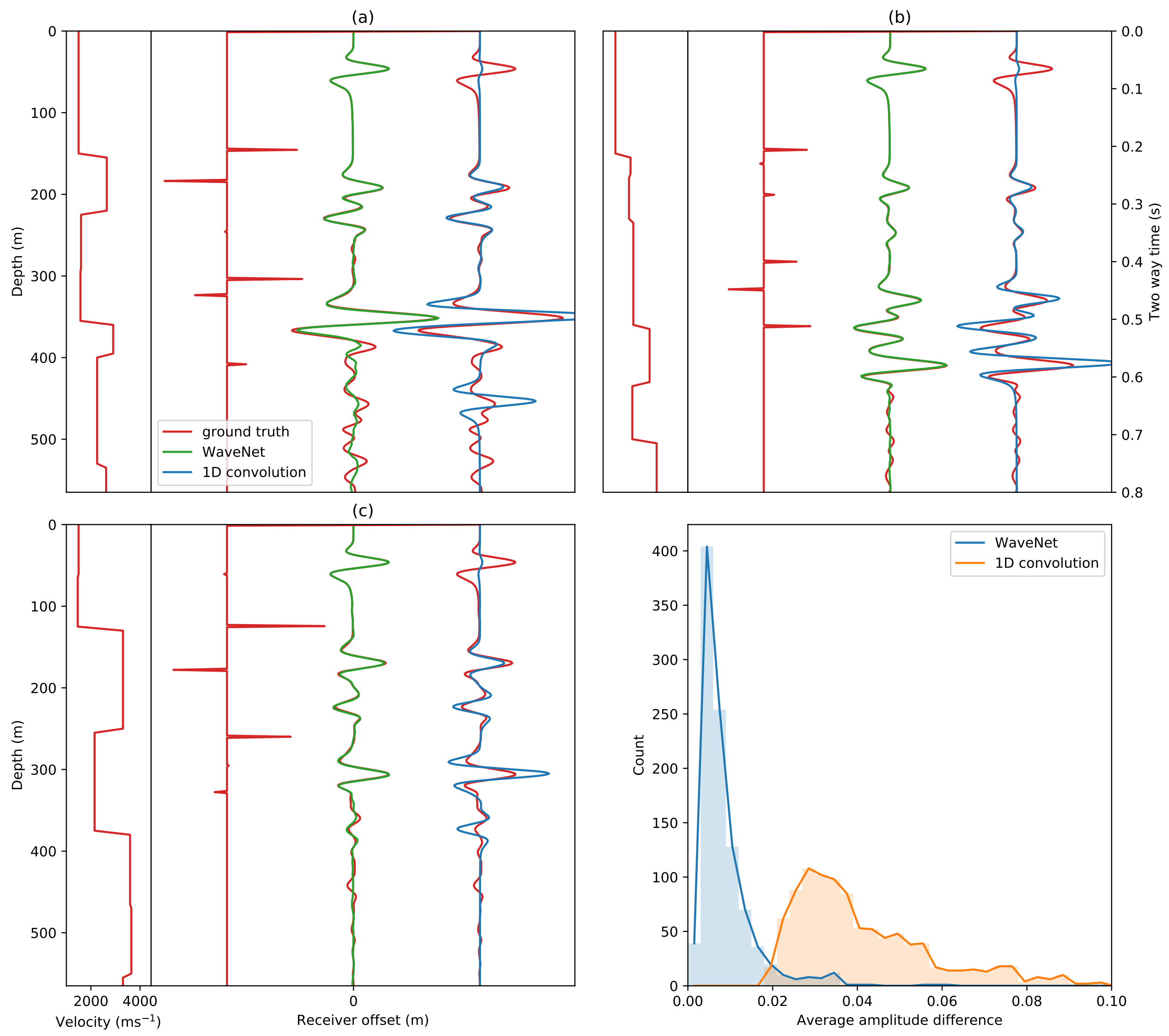

Figure 5. Comparison of WaveNet simulation to 1D convolutional model. We compare our WaveNet simulation for 3 of the examples in Fig 4 to a simple 1D convolutional model. Red shows the input velocity model, its corresponding reflectivity series and the ground truth pressure response at the zero-offset receiver. Green shows the WaveNet simulation at the zero-offset receiver and blue shows the 1D normal incidence convolutional model. Bottom right: histogram of the average absolute amplitude difference between the ground truth FD simulation and the zero-offset simulation from the WaveNet and 1D convolutional model, over the test set of 1000 examples. A $t^{2.5}$ gain is applied to the receiver responses for display.

the ground truth FD modelling simulation in Fig. 4. We also compare our WaveNet simulation to a simple 1D convolutional approximation of the zero-offset receiver response at normal incidence (Russell, 1988), given by $\tilde{Y} = R * S$, where R is the reflectivity series in time and S is the source signature, shown in Fig. 5.

For nearly all time samples our network is able to simulate the receiver pressure responses. Unlike the 1D convolutional model, the WaveNet is able to predict the Normal Moveout (NMO) of the primary layer reflections with receiver offset, the direct arrivals at the start of each receiver recording and the spherical spreading loss of the wavefield over time, though we notice the network struggles to accurately simulate the multiple reverberations at the end of the receiver recordings. We plot the histogram of the average absolute amplitude difference between the ground truth FD simulation and the zero-offset simulation from the WaveNet and 1D convolutional model over the test set in Fig. 5. We observe that the WaveNet simulation has a lower average loss than the 1D convolutional model.

We also investigate the accuracy of the WaveNet simulation with different network designs. Fig. 6 compares the WaveNet simulation to the simulations from using two different convolutional network designs. Both convolutional networks have 9 convolutional layers, each with 256 hidden channels, filter sizes of 3, ReLU activations for all hidden layers and an identity activation function for the output layer. The second network uses exponential dilations whilst the first does not. Both networks have 1,387,531 free parameters in total. We observe that the convolutional network without dilations does not converge during training. We plot the histogram of the average absolute amplitude difference between the ground truth FD simulation and the simulations from the WaveNet and the dilated convolutional network over the test set. The dilated convolutional network has a higher average loss than the WaveNet network.

We measure the average time taken to generate 100 simulations using the SEISMIC_CPML library on a single core of a 2.2 GHz Intel Core i7 processor to be 73 ± 1 s. Using the same core the WaveNet network is able to generate 100 simulations in an average time of 3.79 ± 0.03 s (19 times quicker). Using the TensorFlow library (TensorFlow, 2015) and a Nvidia Tesla K80 GPU produces simulations with an average time of 0.133 ± 0.001 s (549 times quicker). This speedup is likely to be higher than if the GPU was used for accelerating existing numerical methods (Rietmann et al., 2012). The WaveNet network takes approximately 12 hours to train on one Nvidia Tesla K80 GPU, although this training step is only required once and subsequent simulation steps are fast.

3 Fast seismic inversion in 2D horizontally layered acoustic media using WaveNet

In this section we retrain our WaveNet network to carry out fast seismic inversion in horizontally layered 2D acoustic Earth models. This offers a fast alternative to existing inversion algorithms.

3.1 Inversion workflow

We are able to perform seismic inversion in the same media by retraining the WaveNet network with its inputs and output reversed. Its input is now a set of 11 recorded receiver responses and its output is a prediction of the corresponding normal incidence reflectivity series. To recover a prediction of the velocity model, we carry out a standard 1D time-to-depth conversion of the output reflectivity values followed by integration. We use the same WaveNet architecture described in Sect. 2.1, except

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

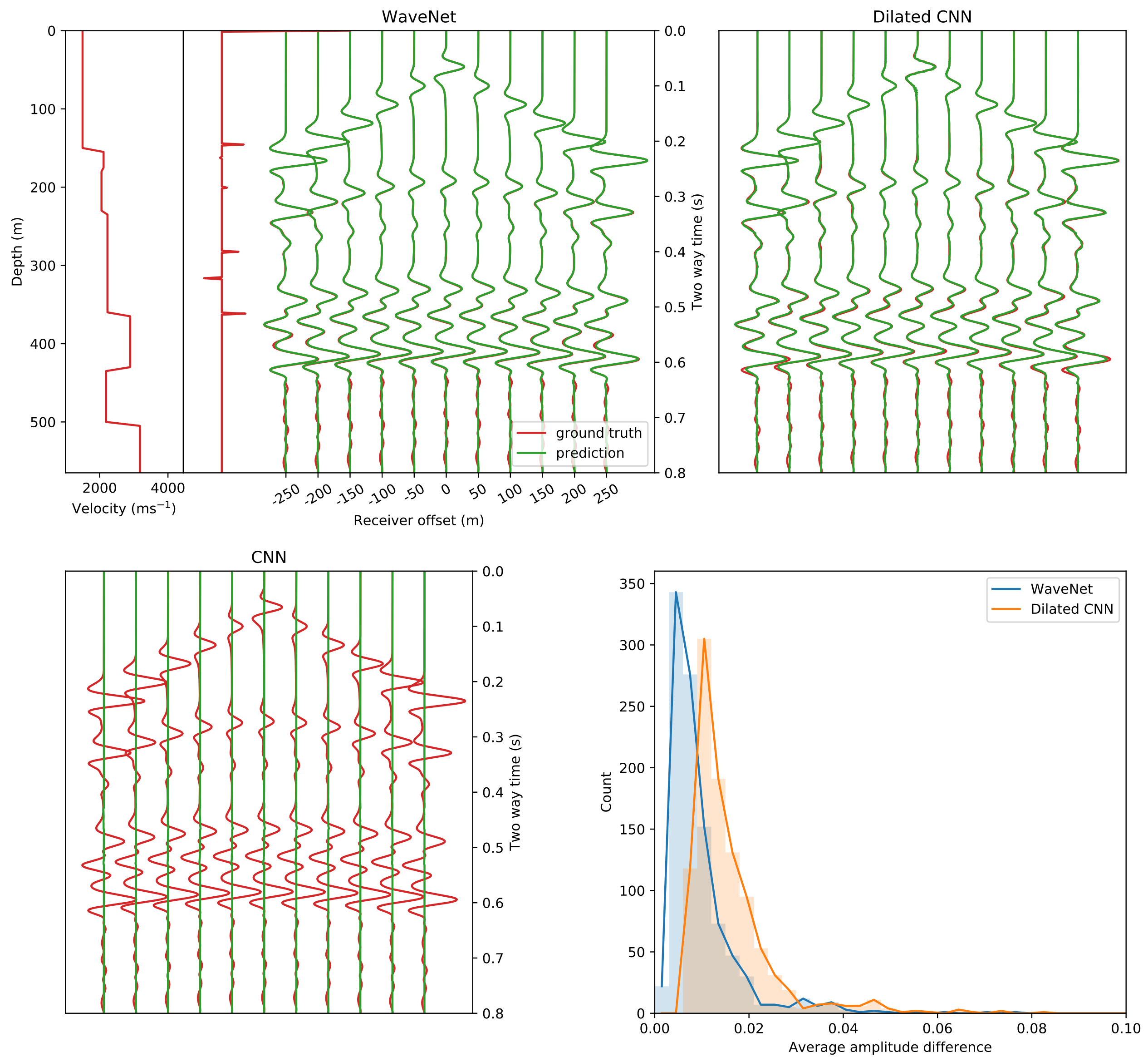

Figure 6. Comparison of network architecture on simulation accuracy. Top left shows the WaveNet simulated pressure response for a randomly selected example in the test set (green) compared to ground truth (red). Top right and bottom left show the simulated response when using convolutional network designs with and without exponential dilations. Bottom right: histogram of the average absolute amplitude difference between the ground truth FD simulation and the simulations from the WaveNet and the dilated convolutional network, over the test set of 1000 examples. A $t^{2.5}$ gain is applied to the receiver responses for display.

that we invert its structure to maintain the causal correlation between the receiver responses and reflectivity series. We also use

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

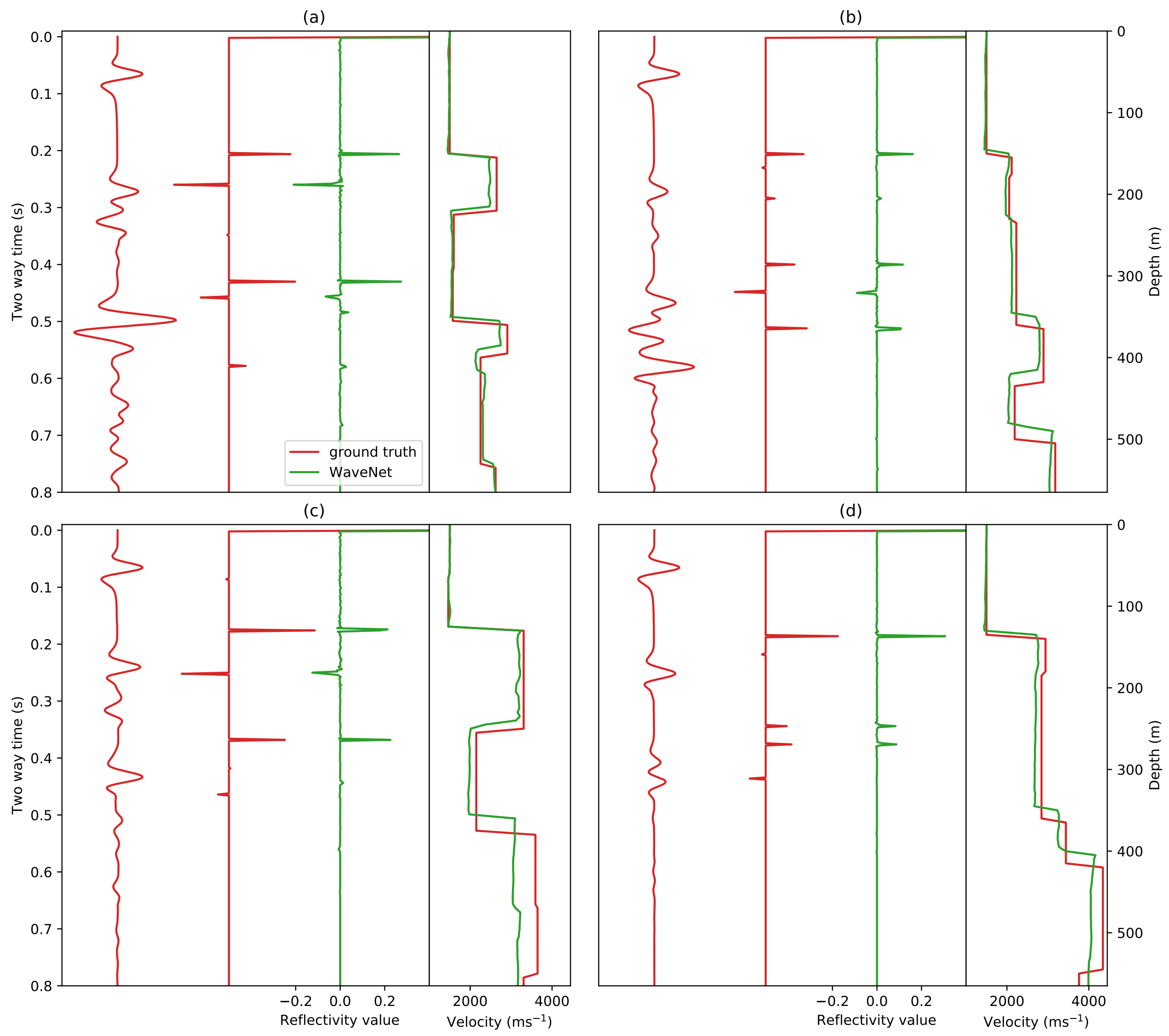

Figure 7. Top: Inverse WaveNet predictions for 4 examples in the test set. Red shows the input pressure response at the zero-offset receiver location, the ground truth reflectivity series and its corresponding velocity model. Green shows the inverse WaveNet reflectivity series prediction and the resulting velocity prediction.

128 instead of 256 hidden channels for each hidden layer. We use exactly the same training data and training strategy described in Sect. 2.3 and 2.4, except that we now use the loss function given by

$$L = \frac{1}{N}\|\hat{R} - R\|_2^2\,, \tag{4}$$

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

where R is the true reflectivity series and $\hat{R}$ is the predicted reflectivity series.

3.2 Results

During training the loss over the training and validation datasets converge to similar values and we test the performance of the trained network using a test set of 1000 unseen examples. Our predictions of the reflectivity series and velocity models for 4 randomly selected examples from this test set are shown in Fig. 7. The inverse WaveNet network is able to predict the underlying velocity model for each example. We observe that in some cases small velocity errors propagate with depth, which is likely a result of the integration of the reflectivity series.

We measure the average time taken to generate 100 velocity predictions on a single core of a 2.2 GHz Intel Core i7 processor to be 1.27 ± 0.02 s. On a Nvidia Tesla K80 GPU this reduces to 0.051 ± 0.001 s. This is likely to be a fraction of the time needed for existing seismic inversion algorithms which rely on forward simulation. We note the prediction time is faster than the forward WaveNet network because the inverse network has less hidden channels in its architecture and therefore requires less computation.

4 Fast seismic simulation in 2D faulted acoustic media using a conditional autoencoder

Our WaveNet network is limited in that it is only able to simulate horizontally layered Earth models. In this section we present a second network which is significantly more general; it simulates seismic waves in 2D faulted acoustic media with arbitrary layers, fault properties and an arbitrary location of the seismic source on the surface of the media.

This is a much more challenging task for multiple reasons. Firstly, the media varies along both dimensions and the resulting seismic wavefield has more complex kinematics than the wavefields in horizontally layered media. Secondly, we allow the output of the network to be conditioned on the input source location which requires the network to learn the effect of the source location. Thirdly, we input the velocity model directly into the network without conversion to a reflectivity series beforehand; the network must learn to carry out its own depth to time conversion to simulate the receiver responses. We chose this approach over our WaveNet workflow because we note that for non-horizontally layered media the pressure responses are not causally correlated to the normal incidence reflectivity series in general and therefore the causality assumption in our WaveNet workflow does not hold.

Similar to Section 2, we simulate the seismic response recorded by a set of receivers horizontally offset from a point source emitted within the Earth model. An example simulation we wish to learn is shown in Fig. 8. We will now discuss our network architecture and training process in more detail below.

4.1 Conditional autoencoder architecture

Our simulation workflow is shown in Fig. 9. Instead of preprocessing the input velocity model to its associated reflectivity model, we input the velocity model directly into our network. Our network is now also conditioned on the source position,

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

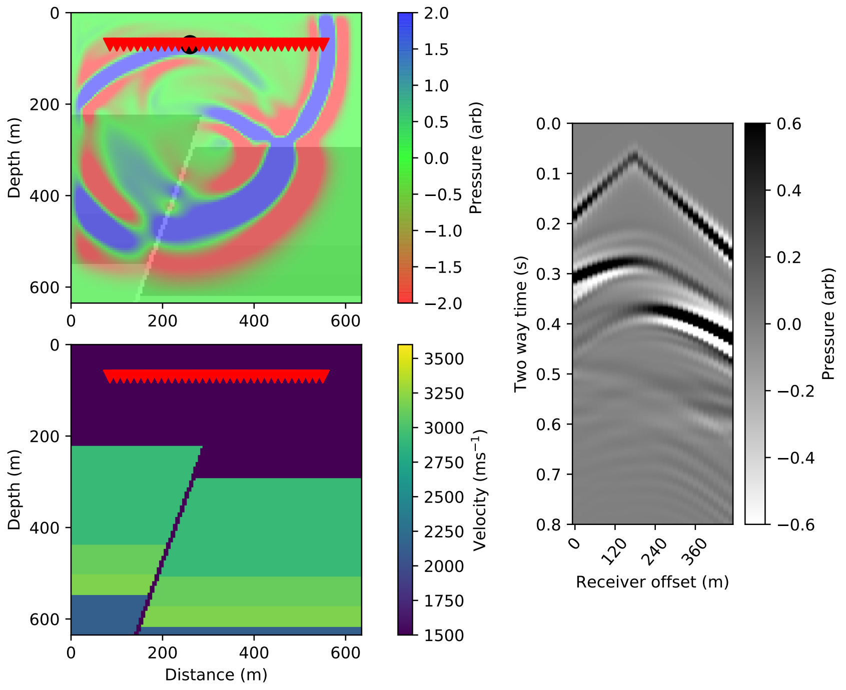

Figure 8. Ground truth FD simulation example, with a 2D faulted media. Left, top: The black circle shows the source location. 32 receivers are placed at the same depth as the source with a horizontal spacing of 15 m (red triangles). The full wavefield pressure is overlain for a single snapshot in time. Left, bottom: The Earth velocity model. Right: The resulting ground truth pressure response recorded by each receiver, using FD modelling. A $t^{2.5}$ gain is applied to the receiver responses for display.

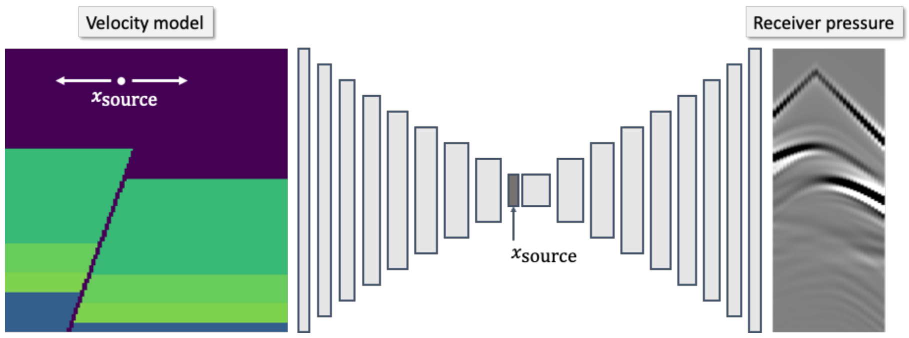

Figure 9. Our conditional autoencoder simulation workflow. Given a 2D velocity model and source location as input, a conditional autoencoder network outputs a simulation of the pressure responses at the receiver locations in Fig. 8. The network is composed of 24 convolutional layers and concatenates the input source location with its latent vector.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

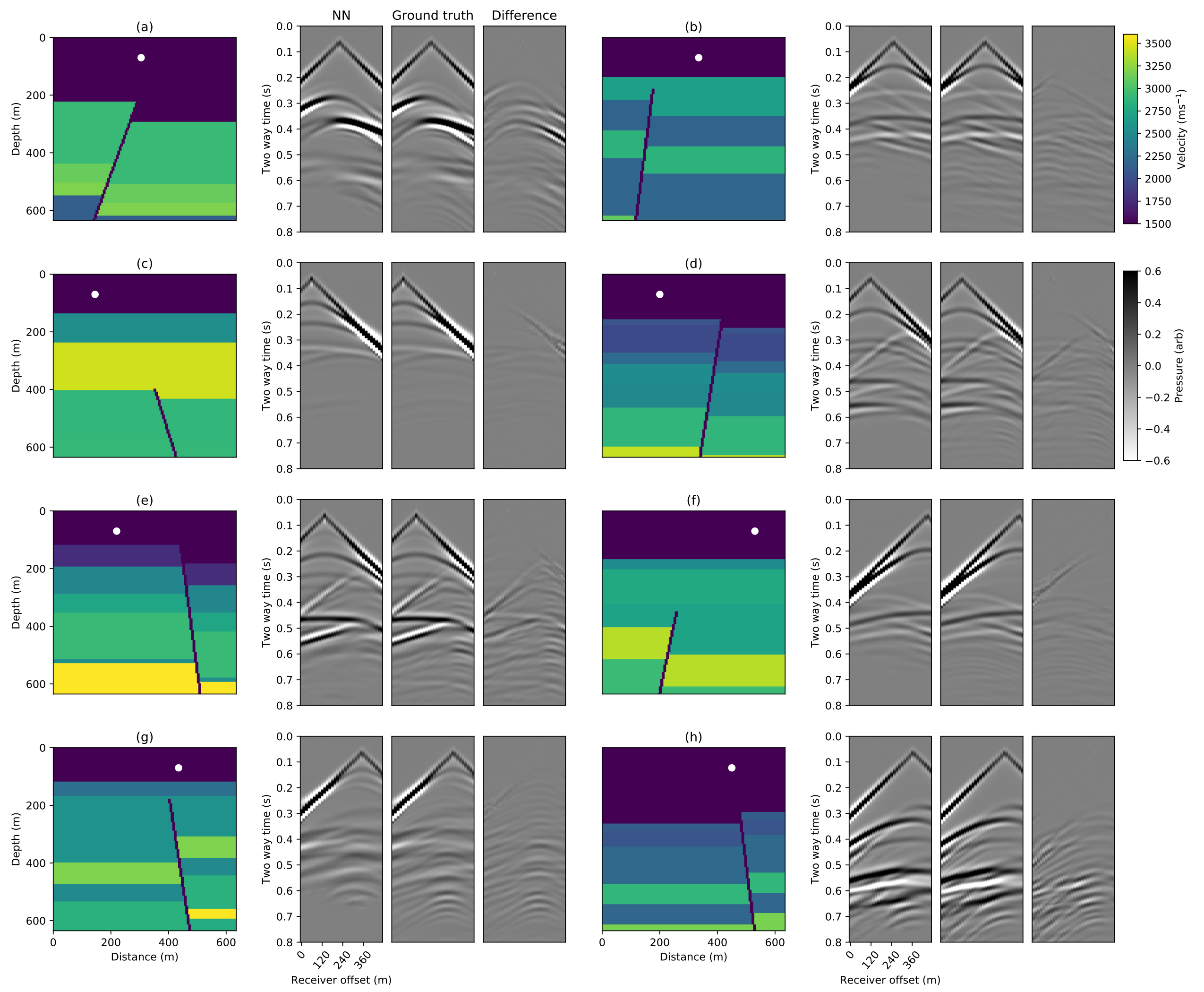

Figure 10. Conditional autoencoder simulations for 8 randomly selected examples in the test set. White circles show the input source location. The left simulation plots show the network predictions, the middle simulation plots show the ground truth FD simulations and the right simulation plots show the difference. A $t^{2.5}$ gain is applied for display.

which can vary along the surface of the Earth model. The output of the network is a simulation of the pressure responses recorded at the 32 fixed receiver locations in the model shown in Fig. 8.

We use a conditional autoencoder network design, shown in Fig 9. The network is composed of 10 convolutional layers which reduce the spatial dimensions of the input velocity model until it has a 1x1 shape with 1024 hidden channels. We term this tensor the latent vector. The input source surface position is concatenated onto the latent vector and 14 transposed convolutional layers are used to expand the size of the latent vector until its output shape is the same as the target receiver gather.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

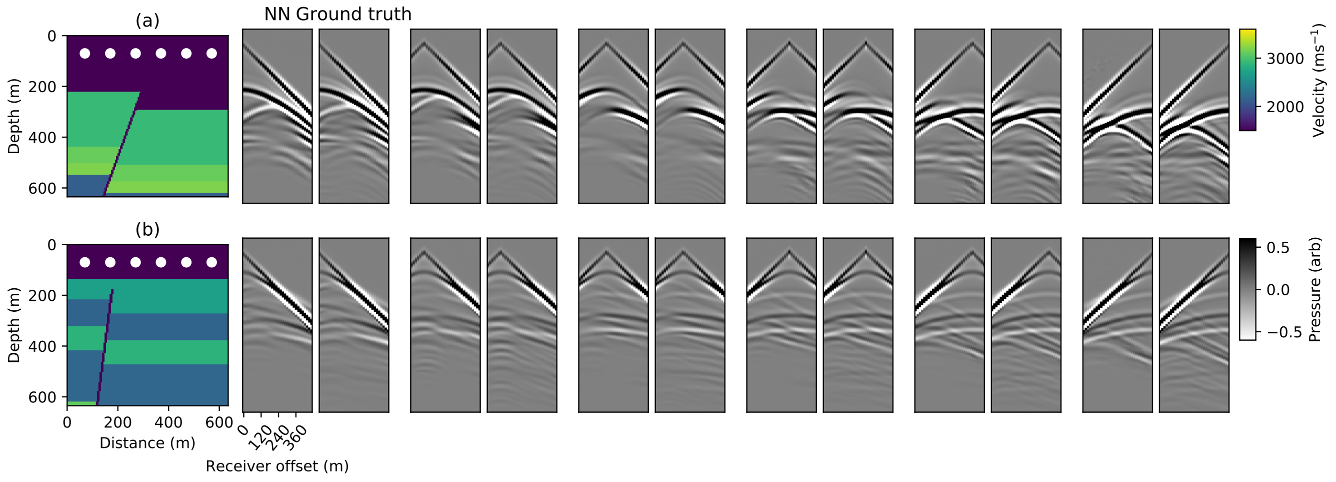

Figure 11. Conditional autoencoder simulation accuracy when varying the source location. The network simulation is shown for 6 different source locations whilst keeping the velocity model fixed. The source positions are regularly spaced across the surface of the velocity model (white circles). Example simulations for 2 different velocity models in the test set are shown, where each row corresponds to a different velocity model. The pairs of simulation plots in each row from left to right correspond to the network prediction (left in the pair) and the ground truth FD simulation (right in the pair), when varying the source location from left to right in the velocity model. A $t^{2.5}$ gain is applied for display.

We choose this encoder-decoder architecture to force the network to compress the velocity model into a set of salient features before expanding them to infer the receiver responses. All hidden layers use ReLU activation functions and the final output layer uses an identity activation function. The resulting network has 18,382,296 free parameters. The full parameterisation of the network is shown in Table 1.

4.2 Training process

We use the same training data generation process described by Section 2.3. When generating velocity models, we add a fault to the model. We randomly sample the length, normal or reverse direction, slip distance and orientation of the fault. Example velocity models drawn from this process are shown in Fig. 10. We generate 100,000 example velocity models and for each model chose three random source locations along the top of the model. This generates a total of 300,000 synthetic ground truth example simulations to use for training the network. We withhold 60,000 of these examples to use as a validation set during training.

We train using the same training process and loss function described in Section 2.4, except that we employ a L1 norm instead of a L2 norm in the loss function (Eq. 3). We use a learning rate of 1x10^{-4}, a batch size of 100 examples and run training over 3,000,000 steps. We use batch normalisation (Ioffe and Szegedy, 2015) after each convolutional layer to help regularise the network during training.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

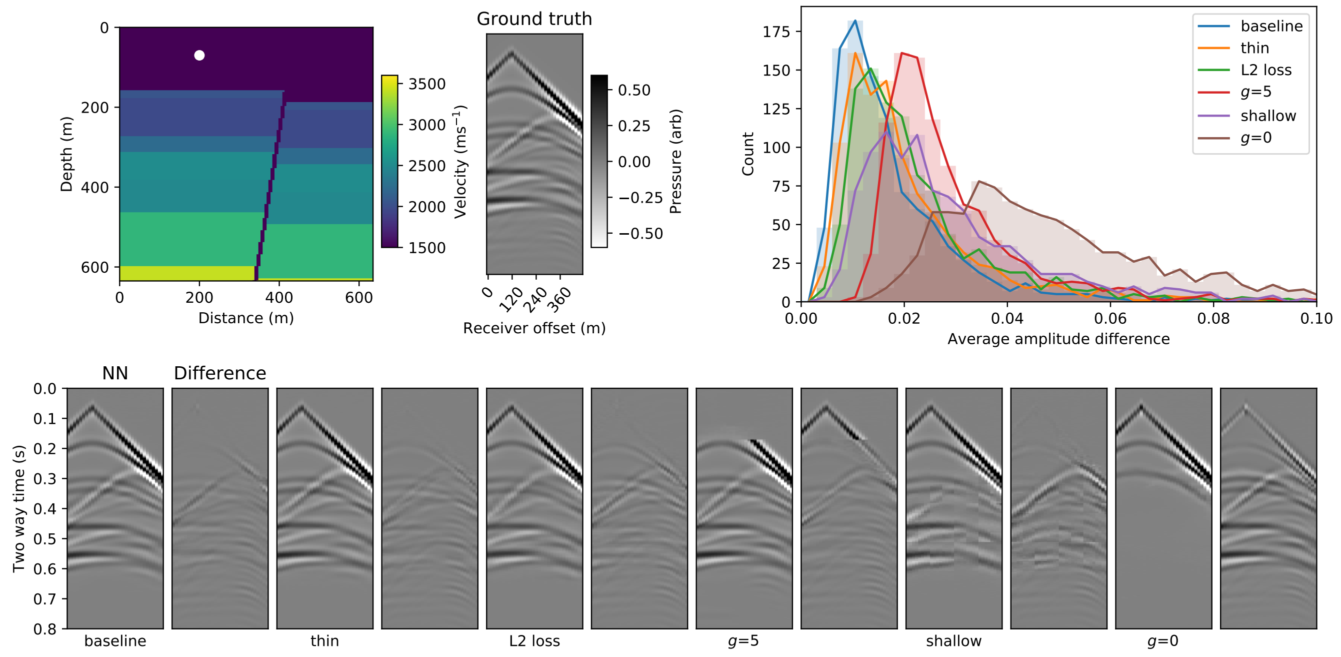

Figure 12. Comparison of different network designs and training hyperparameters on simulation accuracy. Top right shows a randomly selected velocity model and source location from the test set and its corresponding ground truth FD simulation. Bottom compares simulations and their difference to the ground truth when using our proposed conditional autoencoder (baseline), when halving the number of hidden channels for all layers (thin), when using an L2 loss function during training (L2 loss), when using gain exponents of $g = 0$ and $g = 5$ in the loss function and when removing 2 layers from the encoder and 8 layers from the decoder (shallow). Top right: histogram of the average absolute amplitude difference between the ground truth FD simulation and the simulation from the different cases over the test set. A $t^{2.5}$ gain is applied for display.

4.3 Results

During training the loss over the training and validation datasets converge to similar values and we test the performance of the trained network using a test set of 1000 unseen examples. The output simulations for 8 randomly selected velocity models and source positions from this set are shown in Fig. 10. We observe that the network is able to simulate the recorded pressure response. The network is able to simulate the kinematics of the primary reflections and in most cases is able to approximate their relative amplitudes. The network is also able to generalise over different source locations. We demonstrate this capability further in Fig. 11, where we plot the network simulation when varying the source location over 2 velocity models from the test set.

We test the accuracy of the network simulation when using different network designs and training hyperparameters. Fig. 12 compares an example simulation from the test set when using our baseline conditional autoencoder network, when halving the number of hidden channels for all layers, when using an L2 loss function during training, when using gain exponents of

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

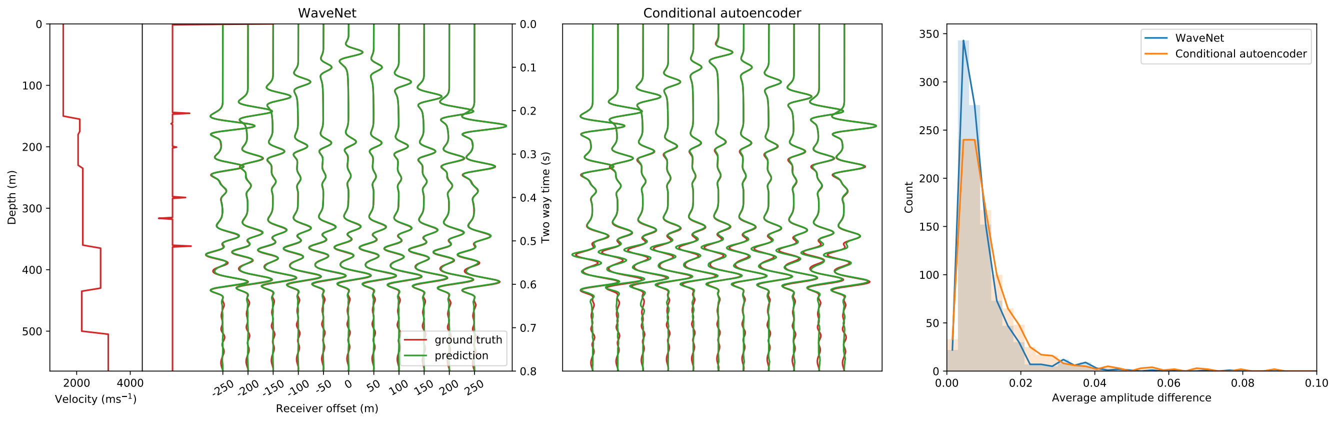

Figure 13. Comparison of the WaveNet and conditional autoencoder simulation accuracy. The left plot shows a velocity model, reflectivity series and ground truth receiver pressure responses for a randomly selected example in the horizontally layered velocity model test set in red. Green shows the WaveNet simulation. The middle plot shows the conditional autoencoder simulation for the same velocity model. The right plot shows the histogram of the average absolute amplitude difference between the ground truth FD simulation and the WaveNet and conditional autoencoder simulations over this test set. A $t^{2.5}$ gain is applied for display.

Layer	Type	in, out channels	kernel size	stride	padding						
1	Conv2d	(1,8)	(3,3)	(1,1)	(1,1)	14	Conv2d	(512,512)	(3,3)	(1,1)	(1,1)
2	Conv2d	(8,16)	(2,2)	(2,2)	0	15	Conv2d	(512,512)	(3,3)	(1,1)	(1,1)
3	Conv2d	(16,16)	(3,3)	(1,1)	(1,1)	16	ConvT2d	(512,256)	(2,4)	(2,4)	0
4	Conv2d	(16,32)	(2,2)	(2,2)	0	17	Conv2d	(256,256)	(3,3)	(1,1)	(1,1)
5	Conv2d	(32,32)	(3,3)	(1,1)	(1,1)	18	Conv2d	(256,256)	(3,3)	(1,1)	(1,1)
6	Conv2d	(32,64)	(2,2)	(2,2)	0	19	ConvT2d	(256,64)	(2,4)	(2,4)	0
7	Conv2d	(64,128)	(2,2)	(2,2)	0	20	Conv2d	(64,64)	(3,3)	(1,1)	(1,1)
8	Conv2d	(128,256)	(2,2)	(2,2)	0	21	Conv2d	(64,64)	(3,3)	(1,1)	(1,1)
9	Conv2d	(256,512)	(2,2)	(2,2)	0	22	ConvT2d	(64,8)	(2,4)	(2,4)	0
10	Conv2d	(512,1024)	(2,2)	(2,2)	0	23	Conv2d	(8,8)	(3,3)	(1,1)	(1,1)
11	Concat	(1024,1025)				24	Conv2d	(8,8)	(3,3)	(1,1)	(1,1)
12	ConvT2d	(1025,1025)	(2,2)	(2,2)	0	25	Conv2d	(8,1)	(1,1)	(1,1)	0
13	ConvT2d	(1025,512)	(2,4)	(2,4)	0						

Table 1. Conditional autoencoder layer parameters. Each entry shows the parameterisation of each convolutional layer. The padding column shows the padding on each side of the input tensor for each spatial dimension.

$g = 0$ and $g = 5$ in the loss function and when removing 2 layers from the encoder and 8 layers from the decoder. We plot the histogram of the average absolute amplitude difference between the ground truth FD simulation and the simulation from all of these cases over the test set. We observe that in all the cases the simulations are less accurate than our baseline approach. Without the gain in the loss function, the network only learns to simulate the direct arrival and the first few reflections in the receiver responses. With a gain exponent of $g = 5$, the network simulation is unstable and it fails to simulate the first 0.2 seconds of the receiver responses. When using the network with less layers the simulations have edge artefacts. The network with half the number of hidden channels is closest to the baseline accuracy. We also find that training a flat convolutional neural network with the same number of layers but without using a bottleneck design to reduce the velocity model to a 1x1x1024 latent vector does not converge.

We compare the accuracy of the conditional autoencoder to the WaveNet network. We plot the simulation from both networks for an example model in the horizontally layered velocity model test set in Fig. 13. We also plot the histogram of the average absolute amplitude difference between the ground truth FD simulation and the WaveNet and conditional autoencoder simulations over this test set. Both networks are able to simulate the receiver responses and the WaveNet simulation is slightly more accurate than the conditional autoencoder, though of course the latter is more general.

We measure the average time taken for the conditional autoencoder to generate 100 simulations using a single core of a 2.2 GHz Intel Core i7 processor to be 3.3 ± 0.1 s (22 times faster than FD modelling on the same core). Similar to the WaveNet network, this is an order of magnitude faster than FD modelling. Using PyTorch (Pytorch, 2018) and a Nvidia Tesla K80 GPU produces simulations with an average time of 0.180 ± 0.003 s (406 times faster than FD modelling). The conditional autoencoder network takes approximately 4 days to train on one Nvidia Titan V GPU. This is 8 times longer than training the WaveNet network, though we made little effort to optimise the training time. We also find that when using only 50,000 training examples the validation loss increases and the network overfits to the training dataset.

5 Discussion and Further Work

We present two neural networks which offer fast methods for simulating seismic waves in horizontally layered and faulted 2D acoustic media. Our WaveNet network is able to simulate the pressure response at multiple receiver locations in horizontally layered velocity models, whilst our conditional autoencoder is more general in that it is able to simulate the response from 2D faulted velocity models, conditioned on the source location.

We find that using causality in our WaveNet network generates more accurate simulations than when using a standard convolutional network without causality. This suggests that adding this physics constraint helps the network simulate the pressure responses. We also show that using exponential dilations appears to be a crictical design choice; using a convolutional network without dilations does not converge. This is likely because without dilations the network's field of view does not cover the entire input reflectivity series.

Our WaveNet network can also carry out seismic inversion, which offers a fast alternative to existing inversion algorithms. We note that seismic inversion is typically an ill-defined problem and it is likely that the predictions the inverse WaveNet architecture makes are biased towards the velocity models it is trained on. This uncertainty could be quantified, for example by using Bayesian deep learning methods (Gal, 2016). We have also yet to compare our inverse WaveNet network to existing seismic inversion techniques, or test it with real seismic data. An alternative method for inversion is to use our forward networks in existing seismic inversion algorithms based on optimisation, such as FWI (Virieux et al., 2017). Both our WaveNet and conditional autoencoder networks are fully differentiable and could therefore be used to generate fast approximate gradient estimates in these methods.

Our conditional autoencoder shows excellent generalisation; it is able to simulate seismic waves in faulted media with arbitrary layers, fault properties and an arbitrary location of the seismic source on the surface of the media. This is a significantly harder task than simulating the horizontally layered media with the WaveNet network. The network shows good accuracy in simulating the kinematics of reflections, but for velocity models with many layers or strong contrasts it sometimes struggles to accurately simulate their relative amplitudes. Our loss function or network design could be investigated further to reduce these amplitude differences.

Our ablation tests show that simulation accuracy is sensitive to the network design and training hyperparameters. Using an appropriate gain function in the loss function appears critical; with too little or too much gain convergence is unstable. Empirically our gain function with $g = 2.5$ performs well. When using less layers in the network we find the simulation has edge artefacts. This suggest that there is a minimum limit on the number of layers when using this encoder-decoder network design. We also find that using a flat network without a bottleneck design does not converge. Our hypothesis is that the bottleneck encourages a depth-to-time conversion by slowly reducing the spatial dimensions of the velocity model before expanding them into time.

We find that the WaveNet network has marginally better performance than the conditional autoencoder when simulating layered velocity models. This may be because the WaveNet network is more physically-constrained for this task; it uses causality and has 18 times less free parameters. It is an open question how best to represent causality in networks for simulating more arbitrary Earth models. We find that both networks have more difficulty simulating multiple reverberations, perhaps because they have more complex wave physics than the primary reflections. Other components such as Long Short-Term Memory (LSTM) or Recursive Neural Network (RNN) cells could be tested inside the networks for improving its prediction of these signals.

A key future challenge is to generalise our networks to simulate more complex Earth models. Whilst our generalisation from horizontally layered velocity models to faulted models is promising, we have yet to consider arbitrary Earth models. Furthermore we focus on acoustic simulation and do not consider elastic or viscoelastic simulation. To generalise further, our conditional autoencoder network requires more free parameters, more time to train and more training examples than the WaveNet network. Elastic and aniostropic parameters would need to be added as additional inputs to the networks and ground truth simulations with more complex Earth models would need to be used to train them. Novel network designs which incorporate physics constraints directly in their architecture or in their loss function may help; it may also be useful to use attention-like mechanisms to help the network focus on the relevant part of the velocity model to carry out simulation, rather than using convolutional layers with full fields of view (Vaswani et al., 2017).

Another key challenge is to move from 2D simulation to 3D simulation. This is challenging because the computation cost of generating training data in 3D is significantly higher than in 2D. However, though large, this could be an amortized cost;

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

the network would only need to be trained once and after this inference steps are still likely to be fast. Whilst we extract only the wavefield at each receiver location to train our network, using the entire wavefield from FD simulation during training may help reduce the number of training simulations.

6 Conclusions

We have presented two deep neural networks for carrying out fast and largely accurate simulation of seismic waves. Both networks are 20 - 500 times faster than FD modelling and simulate seismic waves in horizontally layered and faulted 2D acoustic media. The first network uses a WaveNet architecture and simulates seismic waves in horizontally layered media. We showed that this network can also be used to carry out fast seismic inversion of the same media. Our second network is significantly more general than the first; it simulates seismic waves in faulted media with arbitrary layers, fault properties and an arbitrary location of the seismic source on the surface of the media. Our approaches could pave the way to real-time seismic simulation and benefit seismic inversion algorithms based on forward simulation. Our work suggests that deep learning is a valuable tool for both seismic simulation and inversion.

Code and data availability. All our training data used was generated synthetically, using the SEISMIC_CPML FD modelling library. Our WaveNet code is already publicly available on Github here: https://github.com/benmoseley/seismic-simulation-wavenet. We are happy to release the code to reproduce all our results on Github on publication of this paper.

Author contributions. TNM and AM were involved in the conceptualisation, supervision and review of the work. BM was involved in the conceptualisation, data creation, methodology, investigation, software, data analysis, validation and writing.

Competing interests. TNM is a Topical Editor for the Solid Earth Editorial Board.

Acknowledgements. The authors would like to thank the Computational Infrastructure for Geodynamics (www.geodynamics.org) for releasing the open-source SEISMIC_CPML FD modelling libraries.

We would also like to thank Tom Le Paine for his fast WaveNet implementation on GitHub from which our code was based on (github.com/tomlepaine/fast-wavenet).

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

References

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

Bohlen, T.: Parallel 3-D viscoelastic finite difference seismic modelling, Computers & Geosciences, pp. 887 – 899, 2002.

Cui, Y., Olsen, K. B., Jordan, T. H., Lee, K., Zhou, J., Small, P., Roten, D., Ely, G., Panda, D. K., Chourasia, A., Levesque, J., Day, S. M., and Maechling, P.: Scalable Earthquake Simulation on Petascale Supercomputers, in: 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–20, 2010.

Gal, Y.: Uncertainty in Deep Learning, Ph.D. thesis, University of Cambridge, 2016.

Guo, X., Li, W., and Iorio, F.: Convolutional Neural Networks for Steady Flow Approximation, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, pp. 481–490, 2016.

Hosseini, K., Sigloch, K., Tsekhmistrenko, M., Zaheri, A., Nissen-Meyer, T., and Igel, H.: Global mantle structure from multi-frequency tomography using P, PP and P-diffracted waves, Geophysical Journal International, https://doi.org/10.1093/gji/ggz394, https://doi.org/10.1093/gji/ggz394, ggz394, 2019.

Igel, H.: Computational Seismology: A Practical Introduction, Oxford University Press, Oxford, https://www.oxfordscholarship.com/10.1093/acprof:oso/9780198717409.001.0001/acprof-9780198717409, 2016.

Ioffe, S. and Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, arXiv e-prints, arXiv:1502.03167, 2015.

Kingma, D. P. and Ba, J.: Adam: A Method for Stochastic Optimization, ArXiv e-prints, 2014.

Komatitsch, D. and Martin, R.: An unsplit convolutional Perfectly Matched Layer improved at grazing incidence for the seismic wave equation, Geophysics, 72, SM155–SM167, 2007.

Komatitsch, D. and Tromp, J.: Introduction to the spectral element method for three-dimensional seismic wave propagation, Geophysical Journal International, 139, 1999.

Krischer, L. and Fichtner, A.: Generating Seismograms with Deep Neural Networks, AGU Fall Meeting Abstracts, 2017.

Krischer, L., Hutko, A., van Driel, M., Stähler, S., Bahavar, M., Trabant, C., and Nissen-Meyer, T.: On-Demand Custom Broadband Synthetic Seismograms, Seism. Res. Letters, 2017.

Leng, K., Nissen-Meyer, T., and van Driel, M.: Efficient global wave propagation adapted to 3-D structural complexity: a pseudospectral/spectral-element approach, Geophysical Journal International, 207, 2016.

Leng, K., Nissen-Meyer, T., van Driel, M., Hosseini, K., and Al-Attar, D.: AxiSEM3D: broad-band seismic wavefields in 3-D global earth models with undulating discontinuities, Geophysical Journal International, 217, 2125–2146, https://doi.org/10.1093/gji/ggz092, https://doi.org/10.1093/gji/ggz092, 2019.

Lerer, A., Gross, S., and Fergus, R.: Learning Physical Intuition of Block Towers by Example, in: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, pp. 430–438, 2016.

Long, G., Zhao, Y., and Zou, J.: A temporal fourth-order scheme for the first-order acoustic wave equations, Geophysical Journal International, 194, 1473–1485, 2013.

Lubrano-Lavadera, P., Drottning, Å., Lecomte, I., Dando, B., Kühn, D., and Oye, V.: Seismic Modelling: 4D Capabilities for CO2 Injection, Energy Procedia, 114, 3432 – 3444, 2017.

Moczo, P., Robertsson, J. O., and Eisner, L.: The Finite-Difference Time-Domain Method for Modeling of Seismic Wave Propagation, in: Advances in Wave Propagation in Heterogenous Earth, vol. 48 of *Advances in Geophysics*, pp. 421 – 516, Elsevier, 2007.

https://doi.org/10.5194/se-2019-157
Preprint. Discussion started: 13 November 2019

Moseley, B., Markham, A., and Nissen-Meyer, T.: Fast approximate simulation of seismic waves with deep learning, arXiv e-prints, arXiv:1807.06873, 2018.

Nair, V. and Hinton, G. E.: Rectified Linear Units Improve Restricted Boltzmann Machines, in: Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, pp. 807–814, Omnipress, USA, http://dl.acm.org/citation.cfm?id=3104322.3104425, 2010.

Perol, T., Gharbi, M., and Denolle, M.: Convolutional neural network for earthquake detection and location, Science Advances, 4, 2018.

Pytorch: https://www.pytorch.org, 2018.

Richardson, A.: Seismic Full-Waveform Inversion Using Deep Learning Tools and Techniques, ArXiv e-prints, 2018.

Rietmann, M., Messmer, P., Nissen-Meyer, T., Peter, D., Basini, P., Komatitsch, D., Schenk, O., Tromp, J., Boschi, L., and Giardini, D.: Forward and adjoint simulations of seismic wave propagation on emerging large-scale GPU architectures, International Conference for High Performance Computing, Networking, Storage and Analysis, SC, pp. 1–11, https://doi.org/10.1109/SC.2012.59, 2012.

Russell, B. H.: Introduction to Seismic Inversion Methods, Society of Exploration Geophysicists, https://doi.org/10.1190/1.9781560802303, https://library.seg.org/doi/abs/10.1190/1.9781560802303, 1988.

Schuster, G.: Seismic Inversion, Society of Exploration Geophysicists, 2017.

Siddiqui, N. A., Mathew, M. J., Menier, D., and Hassaan, M.: 2D and 3D seismic simulation for fault modeling: exploratory revision from the Gullfaks field, Journal of Petroleum Exploration and Production Technology, 7, 417–432, 2017.

Sun, H. and Demanet, L.: Low frequency extrapolation with deep learning, in: ERL Annual Founding Members Meeting 2018: Simulation, Inference, and Machine Learning for Applied Geophysics, 2018.

TensorFlow: https://www.tensorflow.org, 2015.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K.: WaveNet: A Generative Model for Raw Audio, ArXiv e-prints, 2016.

van Driel, M. and Nissen-Meyer, T.: Optimized viscoelastic wave propagation for weakly dissipative media, Geophysical Journal International, 199, 1078–1093, https://doi.org/10.1093/gji/ggu314, https://doi.org/10.1093/gji/ggu314, 2014a.

van Driel, M. and Nissen-Meyer, T.: Seismic wave propagation in fully anisotropic axisymmetric media, Geophysical Journal International, 199, 880–893, https://doi.org/10.1093/gji/ggu269, https://doi.org/10.1093/gji/ggu269, 2014b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I.: Attention Is All You Need, ArXiv e-prints, 2017.

Virieux, J., Asnaashari, A., Brossier, R., Metivier, L., Ribodetti, A., and Zhou, W.: 6. An introduction to full waveform inversion, pp. R1–1–R1–40, 2017.

Wu, Y., Lin, Y., and Zhou, Z.: Inversionet: Accurate and efficient seismic-waveform inversion with convolutional neural networks, pp. 2096–2100, 2018.

Xie, X.-B., Jin, S., and Wu, R.-S.: Wave-equation-based seismic illumination analysis, GEOPHYSICS, 71, S169–S177, 2006.

Yang, F. and Ma, J.: Deep-learning inversion: A next-generation seismic velocity model building method, Geophysics, 84, R583–R599, https://doi.org/10.1190/geo2018-0249.1, 2019.

Zhu, W., Sheng, Y., and Sun, Y.: Wave-dynamics simulation using deep neural networks, 2017.