A theory-guided deep learning formulation of seismic waveform inversion

Jian Sun, Zhan Niu, Kristopher A. Innanen, Junxiao Li and Daniel O. Trad, University of Calgary*

SUMMARY

Deep learning techniques appear to be poised to play very important roles in our processing flows for inversion and interpretation of seismic data. The most successful seismic applications of these complex pattern-identifying networks will, presumably, be those that also leverage the deterministic physical models on which we normally base our seismic methods. Motivated by the advantages of recurrent neural networks (RNN) for modelling dynamic processes, a theory-guided RNN is set up to address the forward and inverse problems. Starting from the mathematical form of the RNN, we deduce that, under certain assumptions, the RNN training process can be equivalent at leading order to gradient-based seismic full waveform inversion (FWI). Our numerical analysis shows the Adaptive Moment (or Adam) optimization with learning rate set to match the magnitudes of standard FWI updates appears to produce the most stable and well-behaved waveform inversion results, which is re-confirmed by a multidimensional 2D Marmousi experiment.

INTRODUCTION

Artificial intelligence (AI), primarily in the form of machine learning (ML) and deep learning technology, is having an enormous impact on applied seismology (Ruano et al., 2014; Chen, 2017; Duan et al., 2018; Chopra and Marfurt, 2018; Wrona et al., 2018; Zhang and Zhan, 2017; Shen et al., 2018; Jia et al., 2018; Halpert, 2018; Li et al., 2018) and its practitioners. However, although seismic geophysics is a highly data-rich discipline, much of its success has involved methods based on deterministic physical models. Since we are unlikely to abandon the successes derived from determinism, a realistic goal would be find a balance between what is new and powerful in pattern-identifying ML technologies, and what is reliable within the determinism of seismic physical models. For this reason, the AI paradigm referred to as *theory-guided data science* (Karpatne et al., 2017) would appear to be a good starting point. Mixing theoretical data-model relationships with those derived during network training has already met with early success in applied geophysics (Downton and Hampson, 2018). The purpose of this paper is to further expand on this idea, focusing on the area of waveform inversion.

One natural approach to combining deterministic and pattern-driven (e.g., ML) methods is to attempt to recreate one in terms of the other. A small number of heuristic attempts have been made to pose geophysical forward and inverse problems in terms of networks. Moseley et al. (2018) presented a fast approximate wave simulation using a deep WaveNet composed of causally-connected convolutional layers. Lewis and Vigh (2017) exploited a convolutional neural network (CNN) to generate the prior model for FWI optimization. CNN architectures have also been designed to solve velocity building or inversion tasks in fully end-to-end ways (Wang et al., 2018; Wu et al., 2018). In the latter cases, the requirements for exhaustive training datasets, and issues associated with over- and under-training, are currently difficult to quantify. This is the type of issue a theory-guided data science approach (Karpatne et al., 2018) directly attempts to address.

Theory guided data science has been outlined but not prescribed (Faghmous and Kumar, 2014; Karpatne et al., 2017, 2018), and while being largely in agreement with its tenets, the work of actually devising a theory-guided methodology must be done. In the approach we describe, which amounts to formulating waveform inversion as a specific example of a trainable recurrent neural network (RNN), it appears that several benefits, including and particularly a reduction of the reliance on complete training datasets, and reduction of long computational times, are possible. Our approach (Sun et al., 2018), which is similar to independent recent work (Richardson, 2018), is analyzed for robustness and response to hyperparameter tuning (primarily learning rate) in a 1D velocity inversion setting, and applied to a 2D synthetic Marmousi example.

THEORY

The RNN waveform inversion has as its associated forward problem the numerical solution of the wave equation (Carcione et al., 2002). In 2D acoustic media in with constant density, the wave equation in the time domain is discretized, such that under (for instance) a second-order finite difference in time and space, the wavefield at the current time, $t+\Delta t$, is expressed in terms of two previous instants, t and $t-\Delta t$, as

$$\begin{aligned} \mathrm{u}(\mathrm{r},t+\Delta t) =& v^2(\mathrm{r})\Delta t^2\left[\nabla^2\mathrm{u}(\mathrm{r},t) - s(\mathrm{r},t)\delta(\mathrm{r}-\mathrm{r_s})\right] \\ &+ 2\mathrm{u}(\mathrm{r},t) - \mathrm{u}(\mathrm{r},t-\Delta t). \end{aligned} \tag{1}$$

where ∇^2 is the spatial Laplacian operator, r represents position, u is the pressure or displacement, t is the time coordinate, and s is the source function. Per equation 1, wave propagation is then iteratively simulated using the source term s and the wave field at two previous time steps as inputs. This motivates the formulation of the seismic forward modeling problem in terms of a recurrent neural network, built such that each layer (or cell) represents the wavefield at one instant in time.

To build an architecture supportive of seismic inversion, a single RNN cell (Figure 1) is set up using the finite-difference operator, which takes the wavefield at one past instant as input, and produces the modeled shot record (i.e., the projection of the wavefield onto the measurement surface) at the current instant, saving the modeled wavefield of this block in memory for the next time step. The trainable weights in the RNN were chosen to be subsurface velocity parameters illustrated in purple in Figure 1.

The standard network training problem is to minimize the square of the misfit between the output layers of the network, which in this case are modelled seismic records, with the training data.

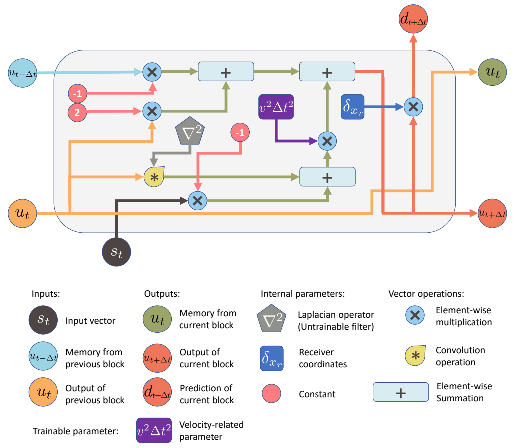

Figure 1: The single cell architecture of the waveform RNN.

This amounts to minimizing the misfit functional J:

$$J(v(\mathrm{r})) = \frac{1}{2n_s}\sum_{\mathrm{r_s}}\sum_{\mathrm{r_g}}\sum_{\mathrm{t}} \delta \mathrm{d_t}^2, \tag{2}$$

where n_s is the number of sources, and $\delta \mathrm{d_t} = \mathrm{d}_t - \tilde{\mathrm{d}}_t$, where d_t and $\tilde{\mathrm{d}}_t$ are training data and RNN-modelled data (respectively) at a fixed receiver, source position, and time. The updated function $v_{n+1}(\mathrm{r})$ is the sum of the current model and an update $\delta v_n(\mathrm{r})$:

$$v_{n+1}(\mathrm{r}) = v_n(\mathrm{r}) + \alpha\, \delta v_n(\mathrm{r}) \tag{3}$$

where α is referred to as the learning rate. With gradient-descent training, the update is based on the gradient of J:

$$\delta v_n(\mathrm{r}) = -g_n(\mathrm{r}) = -\frac{\partial J}{\partial v_n(\mathrm{r})}. \tag{4}$$

Hereafter neglecting the iteration index n, the gradient of J with respect to the RNN weights $v(\mathrm{r})$ is computed as

$$g(\mathrm{r}) = \sum_{\mathrm{t}}^{\mathrm{T}} \left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t)}\right] \frac{\partial \tilde{u}(\mathrm{r},t)}{\partial v(\mathrm{r})}, \tag{5}$$

The partial derivative $[\partial J/\partial \tilde{u}(\mathrm{r},t)]$ can be calculated in terms of the wavefield evaluated at the time increments within the RNN using the chain rule:

$$\begin{aligned}\left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t)}\right] &= \left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t+2\Delta t)}\right]\frac{\partial \tilde{u}(\mathrm{r},t+2\Delta t)}{\partial \tilde{u}(\mathrm{r}',t)} \\ &+ \left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t+\Delta t)}\right]\frac{\partial \tilde{u}(\mathrm{r},t+\Delta t)}{\partial \tilde{u}(\mathrm{r},t)} + \frac{\partial J}{\partial \tilde{u}(\mathrm{r},t)},\end{aligned} \tag{6}$$

where $0 \le t \le T$, and initial conditions for RNN backpropagation are assumed to be zeros, i.e., $[\partial J/\partial \tilde{u}(\mathrm{r},t)]_{t=T+1,T+2} = 0$.

The partial derivatives of $\tilde{u}(\mathrm{r},t+2\Delta t)$ and $\tilde{u}(\mathrm{r},t+\Delta t)$ with respect to $\tilde{u}(\mathrm{r},t)$, and of $\tilde{u}(\mathrm{r},t)$ with respect to $v(\mathrm{r})$ are

$$\frac{\partial \tilde{u}(\mathrm{r},t+2\Delta t)}{\partial \tilde{u}(\mathrm{r},t)} = -1, \tag{7a}$$

$$\frac{\partial \tilde{u}(\mathrm{r},t+\Delta t)}{\partial \tilde{u}(\mathrm{r},t)} = v^2(\mathrm{r})\Delta t^2 \nabla^2 + 2, \tag{7b}$$

$$\frac{\partial \tilde{u}(\mathrm{r},t)}{\partial v(\mathrm{r})} \approx \frac{2\Delta t^2}{v(\mathrm{r})}\frac{\partial^2 \tilde{u}(\mathrm{r},t)}{\partial t^2}. \tag{7c}$$

Substituting equations (7b) and (7a) into equation (6), we obtain

$$\begin{aligned}&\left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t)}\right] = \\ &v^2(\mathrm{r})\Delta t^2 \left(\nabla^2\left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t+\Delta t)}\right] - \frac{1}{n_s v^2(\mathrm{r})\Delta t^2}\sum_{\mathrm{r_s}}\sum_{\mathrm{r_g}}\delta \mathrm{d_t}\right) \\ &+2\left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t+\Delta t)}\right] - \left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t+2\Delta t)}\right].\end{aligned} \tag{8}$$

Equation 8 emphasizes that the partial derivative of the objective function with respect to the predicted wavefield $[\partial J/\partial \tilde{u}(\mathrm{r},t)]$ amounts to propagating the scaled data residual in reversed time. In other words, backpropagation through the network, a standard component of deep learning network training, amounts to backpropagation of the residual wavefield in time. Letting the operator BP represent the repetitive application of this template back to zero time, and substituting

$$\left[\frac{\partial J}{\partial \tilde{u}(\mathrm{r},t)}\right] = \mathrm{BP}\left(-\frac{1}{n_s v^2(\mathrm{r})\Delta t^2}\sum_{\mathrm{r_s}}\sum_{\mathrm{r_g}}\delta \mathrm{d_t}\right) \tag{9}$$

into equation (5), leads to

$$g(\mathrm{r}) \approx \sum_{\mathrm{t}}^{\mathrm{T}} \mathrm{BP}\left(-\frac{1}{n_s}\sum_{\mathrm{r_s}}\sum_{\mathrm{r_g}}\delta \mathrm{d_t}\right)\frac{2}{v^3(\mathrm{r})}\frac{\partial^2 \tilde{u}(\mathrm{r},t)}{\partial t^2}. \tag{10}$$

It is apparent from equation (10) that the RNN training amounts to the crosscorrelation, in time, of the second-order partial derivative of the forward-modelled wavefield and the back-propagated residuals. This process is, of course, a recapitulation of the gradient calculation in time-domain full-waveform inversion (see, for instance, the formulas of Yang et al., 2015). In other words, FWI can be correctly understood as a special case of theory-guided RNN training, in which the RNN is sufficiently well constrained in its *non-trainable* weights, i.e., the fixed model of scalar wave propagation, that a single training data set is used.

WAVEFORM RNN HYPERPARAMETER SELECTION

Tuning of the deep learning hyper-parameter (which lies between 0 and 1) is one of the more difficult, and problem dependent aspects of deep learning design. Successful tuning in general requires significant empirical analysis and a large number of trials. The RNN we have designed in this paper is, in any case, not a typical architecture, so existing guidelines cannot be assumed to hold a priori. In this section we set up a toy 1D wave inversion problem, both to illustrate the behaviour of the deep learning based waveform inversion, and to compare the performances of a variety of optimization algorithms and hyperparameters.

As a benchmark synthetic model, a 1D 4-layer profile, with velocities [2, 3, 4, 5]km/s (from shallowest to deepest) is selected, as illustrated in Figure 2a (black solid line). A 1D version of the second-order finite difference approach is used to synthesize the associated seismic record (i.e., the single trace), as plotted in Figure 2b.

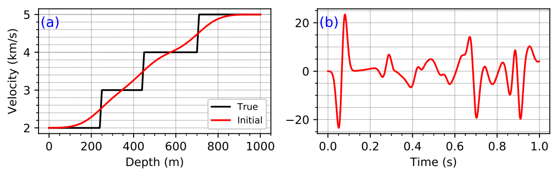

Figure 2: A depth-varying velocity profile. (a) Black: true velocity. Red: initial model. (b) Synthetic seismic trace.

Trials are carried out for hyperparameter tuning (i.e., learning rate selection) to find the best values and useful ranges. In Figure 3, performances of RNN inversion using variant gradient based algorithms with different learning rates are illustrated, where the first column delineates the final inversed results and the second column plots data errors versus iteration number. From the first row to the bottom row: gradient Descent, momentum, Adaptive gradient (Adagrad), Root-mean-square propagation (RMSprop), Adaptive moment (Adam).

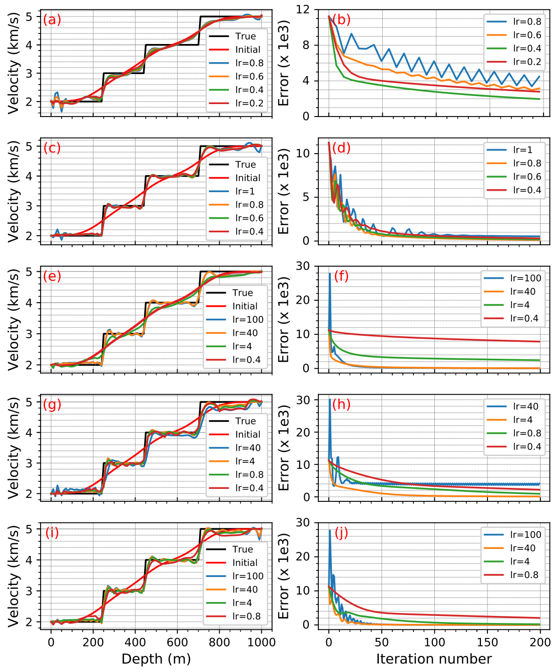

Figure 3: The RNN inversion using gradient-descent. (a) Inversion results; (b) Data error versus iteration.

Figure 3a-b shows that gradient descent waveform RNN training evidently requires a large number of iterations to converge, even after significant effort has been expended to determine the best learning rate. In the momentum algorithm, the adapted learning rate is similar to the gradient descent, and lies between $(0, 1]$. As expected, the speed of convergence and the final predicted results are stabilized by the accumulated gradients shown in Figure 3c-d. For Adagrad algorithm, following Duchi et al. (2011), we set the value of the hyperparameter β to a fixed value of 0.9, and the result is plotted in Figure 3e-f. Unlike momentum, the best learning rate for Adagrad in our tests is 40. We observe that this unusual best learning rate scope is caused by the scaling coefficient, i.e., magnitude differences brought by accumulated squared-norms of the gradients. A detailed analysis for this can be found in our report (Sun et al., 2018). Similarly, by setting the scaling coefficient to be in range of $(0, 1]$, the best learning rate for RMSprop and Adam can also be discovered as $(1, 10]$ and $(10, 100]$ respectively. The results shown in Figure 3h-j cross-examined our deduction.

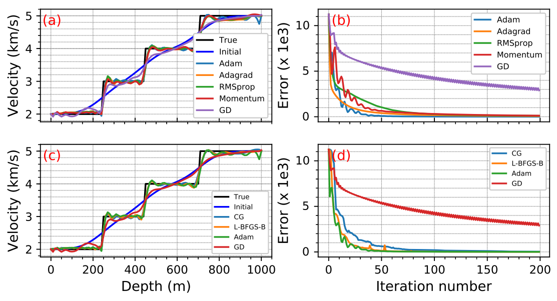

Figure 4: Performance comparison. (a)-(b): best performance comparison of gradient-based algorithms. (c)-(d): performance comparison of non-linear CG, l-BFGS, Adam.

To make fair comparisons between the gradient-based algorithms we have so far discussed, we select amongst each the parameters leading to their best performance; these are plotted in Figure 4a-b. We earlier established that the waveform RNN approach is approximately equivalent to a conventional gradient-based FWI algorithm. We therefore also implemented velocity inversion through RNN framework using non-linear CG and l-BFGS methods. The comparisons of Adam and these two second-order optimization algorithms are illustrated in Figure 4c-d. The results show that Adam, non-linear CG and l-BFGS are capable of retrieving velocity profiles in close agreement with the true model, within 200 iterations. In this 1D case, the convergence of Adam is equivalent to that of the l-BFGS implementation, which in turn is slightly faster than non-linear CG.

MULTIDIMENSIONAL WAVEFORM INVERSION WITH RNN TRAINING

To expand the numerical analysis of seismic inversion through waveform RNN training, we formulate a 2D scalar acoustic version, and apply it to synthetic data computed from the Marmousi model. A single smoothed velocity model is used to initialize all RNN training parameters. The true and initial velocity models are plotted in Figure 5a. We generate 12 shot gathers for source locations at regular 25m intervals at a depth of 40m from the top of the model.

Each of Adam, CG, and L-BFGS are employed to train the waveform RNN with data from the Marmousi model. The inversion results generated with nonlinear CG optimization are plotted at iterations $[400, 800, 1420]th$ in Figure 6. For consistent and maximally fair comparisons, and numbers which shed light on overall computational expense, we set *iteration* to mean the *number of forward modeling simulations*. Based on both this result, and our analysis of non-linear CG on the 1D case, we conclude that non-linear CG is stably convergent, but requires many iterations and is therefore prohibitive

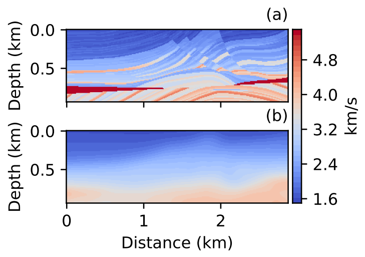

Figure 5: The synthetic 2D Marmousi model. (a) True velocity model; (b) initial velocity model.

in terms of computational cost. The counterpart waveform RNN with l-BFGS optimization on the 2D Marmousi model, at the $[200, 600, 1000]th$ iterations, are plotted in Figure 7. In comparison with the non-linear CG optimziation, the l-BFGS converges at a higher rate. However, the computational cost (1000 iterations) of the l-BFGS optimization is high, and the predicted velocity profiles are missing important features. The results at iterations $[25, 50, 100]th$ using Adam with learning rate 40 are plotted in Figure 8. Adam recovers almost all structural and layer information in shallow zones. After 100 iterations, accurate layer and structural information in deep zones.

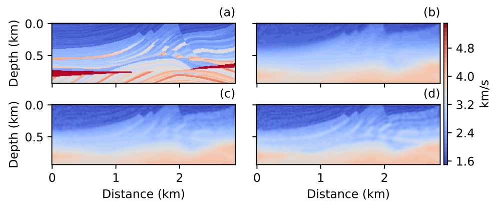

Figure 6: Inversion using the nonlinear CG algorithm. (a) True model. (b) Inversion at 400th iteration. (c) Inversion at 800th iteration. (d) Inversion at 1420th iteration.

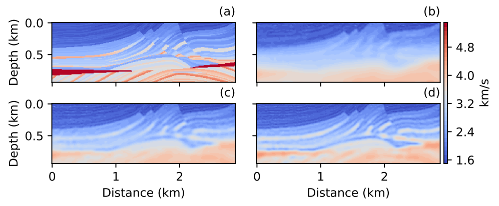

Figure 7: Inversion using the l-BFGS algorithm. (a) True model. (b) Inversion at 200th iteration. (c) Inversion at 600th iteration. (d) Inversion at 1000th iteration.

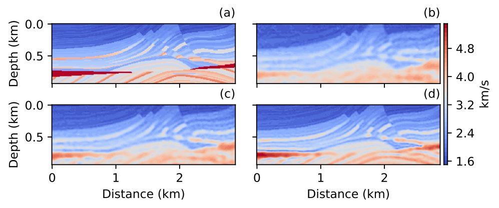

Figure 8: The inversion using the Adam algorithm. (a) True model. (b) Inversion at 25th iteration. (c) Inversion at 50th iteration. (d) Inversion at 100th iteration.

To characterize the relative computational cost of each algorithm, we plot data error versus iteration number in Figure 9. Compared to the nonlinear CG and l-BFGS algorithms, Adam is much faster, and exhibits a more stable path to convergence.

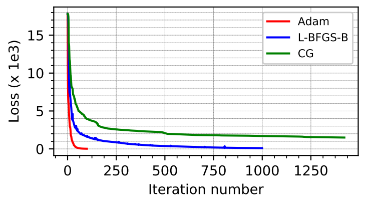

Figure 9: Loss values associated with Adam, CG, L-BFGS algorithms.

CONCLUSIONS

Deep learning is now being widely applied across scientific disciplines, including the data-rich problem of seismic analysis and inversion. Our viewpoint is that the approaches which make maximal use physical models which explain seismic data will be most likely to succeed in our field; the technical term for this is *theory-guided* (or *physics-informed* in turbulence modeling). When the forward propagation of information through the network mimics the forward propagation of a wavefield through a heterogeneous medium, the associated RNN training process is a form of waveform inversion. Derivation of the gradient for a least-squares based training of the waveform RNN can be shown to be equivalent to the that of FWI under certain assumptions. In other words, it is correct to refer to FWI as a strongly constrained, theory-designed deep learning network. Best learning rate ranges for gradient-based algorithms are theoretically analyzed and experimentally investigated. Different gradient-based and non-linear algorithms are inter-compared and cross-compared their internal optimization is achieved in the 1D seismic inversion sense. To further examine its capacity and feasibility, the waveform RNN is then applied to a multidimensional 2D Marmousi model using Adam, CG, and L-BFGS methods. The results indicate that the computational cost of non-linear CG is much higher than for the others. The l-BFGS has better convergence properties, but it also exacts a high computational cost in recovering detailed information. The gradient-based Adam converges more rapidly and recovers detailed velocity information at both shallow and deep zones.

We view the research in this paper as the first step in creating a flexible class of waveform inversion methods based on deep learning, which sit on a spectrum between strongly data-driven and strongly theory-guided methods. Our main line of ongoing research involves finding algorithms which sit in the centre of this spectrum, permitting, when data completeness allows, weights to be trained which are associated with the physics model as well as the parameter values.

ACKNOWLEDGMENTS

We thank the sponsors of CREWES for continued support. This work was funded by CREWES industrial sponsors and NSERC (Natural Science and Engineering Research Council of Canada) through the grant CRDPJ 461179-13.