Learned iterative solvers for the Helmholtz equation

Gabrio Rizzuti, Ali Siahkoohi, and Felix J. Herrmann

Georgia Institute of Technology

January 22, 2019

Abstract

We propose a 'learned' iterative solver for the Helmholtz equation, by combining traditional Krylov-based solvers with machine learning. The method is, in principle, able to circumvent the shortcomings of classical iterative solvers, and has clear advantages over purely data-driven approaches. We demonstrate the effectiveness of this approach under a 1.5-D assumption, when adequate a priori information about the velocity distribution is known.

Introduction

We explore the potential role of neural networks as a solver for the Helmholtz equation, here discretized by finite-differences. The underlying principle of this work is merging data-driven approaches, such as machine learning, with the physical/mathematical description of the problem. The advantage is reduced computational complexity, on the one hand, and simplified training phase, on the other. We are mainly inspired by the recent works of [1], which combines gradient-based optimization and deep learning for inverse problems, and [11], where finite-difference time-domain modeling is trained to reduce numerical dispersion.

The starting point of this paper is the classical Krylov-space iterative solver. Notoriously, Krylov-based solvers for the wave equation are inefficient when no preconditioning is applied, since Krylov-space approximations—due to operator locality—have adequate support only for an exceedingly large number of iterations [4]. Building a preconditioner for the Helmholtz operator, however, is no trivial matter because of its indefiniteness [see, for example, 3, 9]

The basic idea, here, is to intersperse Krylov-based iterations with neural net corrections. (Note that this can also be interpreted as a form of nonlinear preconditioning.) We are motivated by recent advancements in the multiscale behavior of convolutional neural nets [CNN, 8], whose guiding principles are not too dissimilar from, e.g., multigrid methods [12]. We refer to [6, 13] for the particular architectures of the neural networks used in this paper.

In abstract, we seek to approximate the nonlinear modeling function $F : M \to U$, which maps a squared slowness model $m \in M$ to the associated solution $u \in U$ of the Helmholtz equation, for a fixed frequency and source location. Training data, in a supervised setting, consist of model/solution pairs (m,u), where m belongs to a certain set M_0. In the unsupervised case, we only have $m \in M_0$. Based on the available data, training selects an approximation $F_\theta : M \to U$, where $\theta \in \Theta$ is a neural net parameter. The broad challenge is to study how the F_θ behaves for models not contemplated in the training phase.

Method

This paper constitutes a preliminary proof-of-concept, and, as such, we adopt a simplifying 1.5-D assumption for Helmholtz. In this setting, the operator reads:

$$H[m] = -k_z^2 - \Delta, \tag{1}$$

where $k_z = \sqrt{\omega^2 m - k_x^2}$. Quantities k_x and k_z denote horizontal and vertical wavenumbers, ω is a (fixed) frequency, and $m = m(z)$ is the depth-dependent squared slowness model. The operator Δ is a finite-difference discretization of the Laplacian. The forward modeling map is then defined by:

$$F : M \to U, \qquad F(m) = (H[m])^{-1} s, \tag{2}$$

for a fixed source function s. Domain and codomain in (2) are, respectively, certain model and wavefield sets.

As a candidate for an approximation of the forward map F, we define (by induction) the following structure:

$$F_{\boldsymbol{\theta}} = F_{(\theta_1,\ldots,\theta_p)} : M \to U, \qquad F_{(\theta_1,\ldots,\theta_p)}(m) = F^q_{\mathrm{Kr}}(m, N_{\theta_p} \circ F_{(\theta_1,\ldots,\theta_{p-1})}(m)), \tag{3}$$

where $\circ$ symbolizes map composition. For $p = 1$,

$$F_{\theta_1}(m) = F^q_{\mathrm{Kr}}(m, N_{\theta_1}(F^q_{\mathrm{Kr}}(m, u_0))), \tag{4}$$

where the starting guess u_0 is a fixed field (typically, the null field). The evaluation of the mapping defined in (3) simply amounts to a sequence of (q) Krylov iterations (F_{Kr}) and neural net corrections (N_θ).

Plain Krylov iterations are encoded by the mappings:

$$F^q_{\mathrm{Kr}} : M \times U \to U, \qquad F^q_{\mathrm{Kr}}(m,u) \leftarrow \begin{array}{l} q \text{ iters applied to eq.: } H[m]v = s, \\ \text{with starting field } u. \end{array} \tag{5}$$

In the following, for example, we will stick to a conjugate-gradient method (CG) applied to the normal equation associated to (1). However, many different choices are also available, such as GMRES [10]. Neural net corrections are indicated by $N_{\theta_l} : U \to U$, each of which adopts the architecture described in [13]: in summary, a stack of convolutional layers comprising downsampling, a sequence of residual neural nets [5], and upsampling. For each single net N_{θ_l}, the parameter θ_l encompasses the many convolutional stencils and biases employed across the different layers forming N_{θ_l}.

The training phase is determined by the choice of a loss function. We first define the least-squares (squared) norm functional:

$$L : U \times U \to \mathbb{R}, \qquad L(u,v) = ||u-v||_2^2.$$

We distinguish the supervised or unsupervised setting: in the supervised case, where model/solution pairs (m_i, u_i) are available, training corresponds to the following optimization problem:

$$\min_{\boldsymbol{\theta}} \sum_i L(u_i, F_{\boldsymbol{\theta}}(m_i)), \tag{6}$$

while in the unsupervised case we resort to the following measure of the residual:

$$\min_{\boldsymbol{\theta}} \sum_i L(s, H[m_i] F_{\boldsymbol{\theta}}(m_i)). \tag{7}$$

In the following numerical experiments, we will use the Adam optimizer [7].

Example

We test the ideas enunciated in the previous section with a simple example. We consider a 1-D three-layered velocity profile defined on the depth interval $z \in [0, 500]$ m. All the layers have the same thickness. Velocity values are, respectively, $v_1 = 2000$ km/s, $v_2 = 3000$ km/s, and $v_3 = 4000$ km/s. We randomly generate different realizations of the profile, by perturbing layer interfaces according to a normal distribution with zero mean and a standard deviation of 25 m. For each of these models, we also solve for the corresponding Helmholtz equation (via direct methods). We consider a frequency $\omega = 2\pi f$, with $f = 60$ Hz, and $k_x = \omega p$ with ray parameter $p = \sin(\pi\alpha/180)/v_1$ and angle $\alpha = 10°$. The point source is located at $z = 0$ m. The resulting training data are shown in Figure 1.

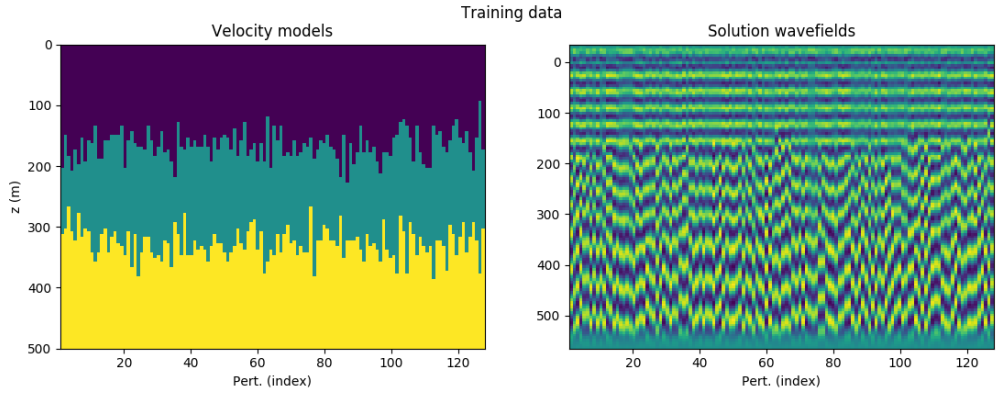

Figure 1 *Velocity model perturbations and corresponding solution wavefields (real part)*

Note that an iterative CG scheme will take about 300 iterations to converge to the solution with a normalized residual tolerance of 10^{-4}.

Results

We train a structured net (3) with $p = 3$ corrections and $q = 20$ Krylov iterations. We will use the unsupervised criterion (7), based on residual minimization. We then compare direct solutions (as in Figure 1), solutions obtained by the trained net, and the result after 80 CG iterations (with no correction involved). In Figure 2, we report such comparison on the training set and a limited test set not included in the training phase (drawn from the same distribution described earlier).

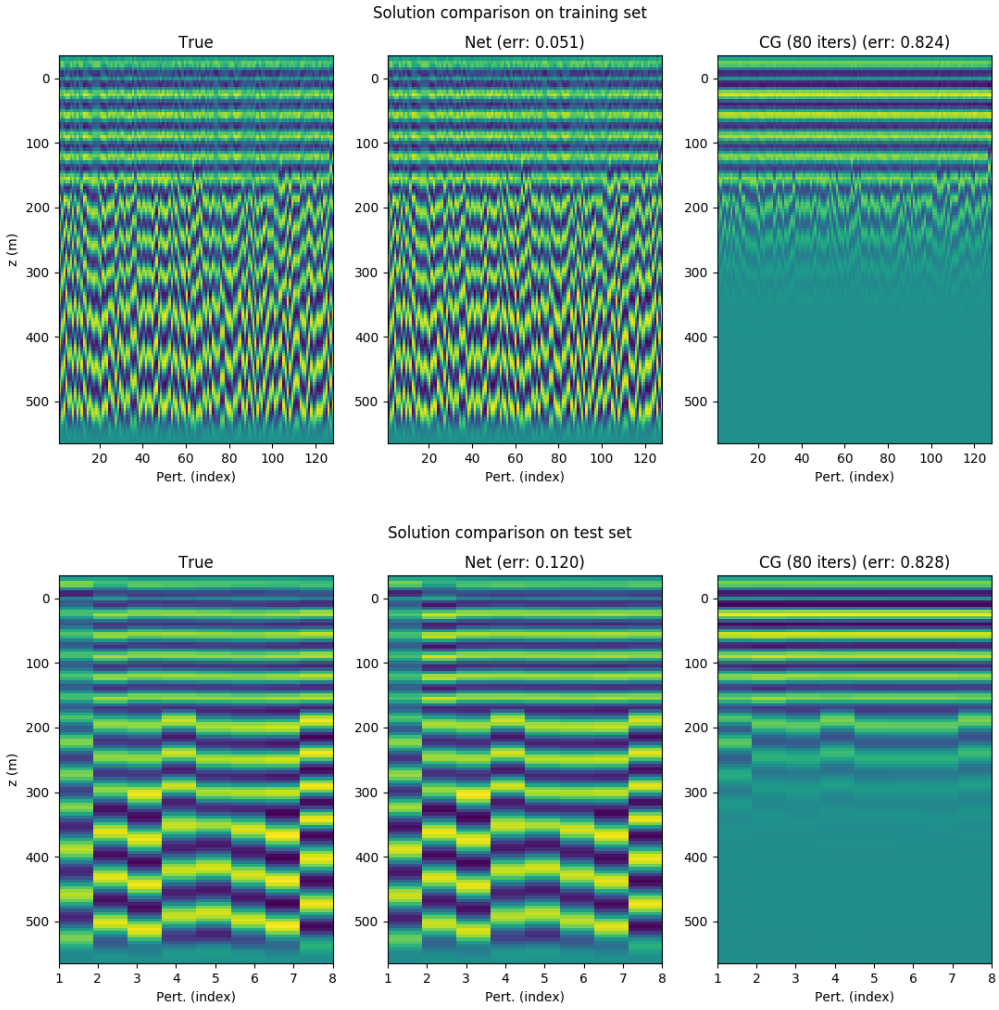

Figure 2 *Comparison of the solutions obtained via direct methods, learned iterative solver, and CG iterative scheme: training set (top row), test set (bottom row)*

Qualitatively, the modeling obtained from the trained net compares favorably with the true solution. Figure 2 also displays the shortcomings of conventional iterative solvers. For a yet more exhaustive comparison, we measure how many iterations are required by the standard CG scheme to reduce the residual least-squares norm to the same values corresponding to the learned iterative solutions. The results corresponding to the test set are summarized in Table 1.

Summary and conclusions

We tested the capabilities of a learned iterative solver for the Helmholtz equation, stemming from the combination of classical iterative solvers and machine learning. The main idea is to improve convergence by 'propagating' an approximated wavefield, obtained from a limited number of iterations, with the aid of a convolutional neural net.

Pert. (index)	1	2	3	4	5	6	7	8
Residual	0.044	0.038	0.024	0.025	0.017	0.023	0.028	0.028
CG iters	187	201	222	217	302	277	217	214

Table 1 *Number of CG iterations required by the standard iterative solver to reduce the (normalized) residual value to the same levels obtained by the learned method (on the test set displayed in Figure 2)*

The results showed in this paper seem to suggest that the learned iterative method can be effective when adequate a priori information about the model space M is assumed. Quite interestingly, true field solutions might not be necessary for successful training.

Future work will involve the study of different Krylov-based methods and net corrections (e.g. by exploiting ideas from multigrid). The extension to 2-D is straightforward. Parallel to this work, one could also train a mapping as a function of source location, for a fixed velocity model. In general, we are also interested in exploring the role of transfer learning, in order to finely tune a pre-trained net when a more specific setting is encountered.

References

[1] Adler, J. and Öktem, O. [2017] Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, **33**(12), 4007.
[2] Adler, J. and Öktem, O. [2018] Learned Primal-dual Reconstruction. *arXiv preprint*.
[3] Erlangga, Y.A., Oosterlee, C.W. and Vuik, C. [2006] A novel multigrid based preconditioner for heterogeneous Helmholtz problems. *J. Sci. Comput.*, **27**(4), 1471–1492.
[4] Ernst, O.G. and Gander, M.J. [2012] Why it is Difficult to Solve Helmholtz Problems with Classical Iterative Methods. *Numerical Analysis of Multiscale Problems*, **83**, 325–363.
[5] He, K., Zhang, X., Ren, S. and Sun, J. [2015] Deep Residual Learning for Image Recognition. *arXiv preprint*.
[6] Johnson, J., Alahi, A. and Fei-Fei, L. [2016] Perceptual Losses for Real-Time Style Transfer and Super-Resolution. *arXiv preprint*.
[7] Kingma, D.P. and Ba, J. [2014] Adam: A Method for Stochastic Optimization. *arXiv preprint*.
[8] Krizhevsky, A., Sutskever, I. and Hinton, G.E. [2012] Imagenet classification with deep convolutional neural networks. In: *Advances in neural information processing systems*. 1097–1105.
[9] Rizzuti, G. and Mulder, W.A. [2016] Multigrid-based 'shifted-Laplacian' preconditioning for the time-harmonic elastic wave equation. *J. Comput. Phys.*, **317**(15), 47–65.
[10] Saad, Y. [2003] *Iterative Methods for Sparse Linear Systems*. SIAM, 2 edn.
[11] Siahkoohi, A., Louboutin, M., Kumar, R. and Herrmann, F.J. [2018] Deep-convolutional neural networks in prestack seismic: Two exploratory examples. *SEG Technical Program Expanded Abstracts 2018*, 2196–2200.
[12] Trottenberg, U., Oosterlee, C.W. and Schüller, A. [2001] *Multigrid*. Academic Press.
[13] Zhu, J.Y., Park, T., Isola, P. and Efros, A.A. [2017] Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*.