Accelerated full seismic waveform modeling and inversion with U-shaped Neural Operators

Yan Yang,1* Angela F. Gao,2 Kamyar Azizzadenesheli3, Robert W. Clayton1, and Zachary E. Ross1

1Seismological Laboratory, California Institute of Technology
2Dept. of Computing and Mathematical Sciences, California Institute of Technology
3Nvidia Corporation, Santa Clara, CA

Abstract

Full seismic waveform modeling is a powerful tool for determining earth structure models and unraveling earthquake rupture processes, but it is computationally expensive. We introduce a scheme to vastly accelerate these calculations with a recently developed machine learning paradigm called neural operator. Once trained, these models can simulate a full wavefield at negligible cost. We use a U-shaped neural operator to learn a general solution operator to the 2D elastic wave equation from an ensemble of numerical simulations performed with random velocity models and source locations. We show that full waveform modeling with neural operators is nearly two orders of magnitude faster than conventional numerical methods. Importantly, the trained model enables accurate simulation for arbitrary velocity models, source locations, and mesh discretization, even when distinctly different from the training dataset. The method also allows for efficient full-waveform inversion with automatic differentiation.
Introduction
The seismic wave equation connects displacement fields to external forces and the density and elastic structure in the Earth. Solutions to the wave equation form the basis of ground shaking simulations of large earthquakes \(^1–^3\) and full waveform inversion for Earth’s structure \(^4–^6\). Due to the highly heterogeneous nature of the Earth, as exemplified by subduction zones and sedimentary basins, there are no exact analytical solutions for these wavefields. Instead, approximate solutions are made possible by discretizing derivatives, which can be numerically computed through iterative time stepping or in the frequency domain. Finite difference methods (FDM) have been popular since the early 80s due to their relatively straightforward formulation \(^7–^9\). The spectral-element method (SEM), a particular case of finite element methods (FEM), which was introduced to seismology in early 2000s, combined the flexibility of FEMs with the accuracy of spectral approaches \(^10–^13\). These numerical solvers impose a tradeoff between resolution and computation speed, with the computational cost proportional to the fourth power of frequency. Thus, the cost of wave simulation is a major barrier to using full-waveform techniques for seismic inversion and updating models of the subsurface with new data.

A number of machine learning-based methods have been proposed in the past few years to provide a faster alternative for tackling seismological problems such as signal denoising \(^14,15\), event detection \(^16–^18\), and phase association \(^19,20\). Deep neural networks have also recently been used to solve partial differential equations (PDEs), such as the Eikonal equation and wave equation \(^21–^25\). These approaches to solving PDEs offer not only speedup in computational capabilities, but also low-memory overhead, differentiability, and on-demand solutions. Such advantages facilitate deep learning being used for seismic inversion \(^26–^28\). However, one major limitation of these approaches is that the solutions generated by these models are dependent on the specific spatial and temporal discretization in the numerical simulation training set.

Recently, a new paradigm named ‘neural operator’ was developed to address the mesh-dependent shortcoming of classical neural networks by creating a single deep learning model that can be applied to different discretizations \(^29–^32\). This is made possible because neural operators can provably learn mappings between infinite-dimensional function spaces \(^33\) and therefore are suitable for learning general solution operators to PDEs, which are valid even when the PDE constants (e.g. elastic properties) are varied. Since first introduced \(^29\), a variety of neural operator models have been developed. In particular, the Fourier neural operator (FNO) is a model that uses the fast Fourier transform as an integral operator, and has been shown to outperform other neural networks or operators in terms of efficiency and accuracy \(^34\). The FNO has been applied to many types of scientific problems including weather forecasting \(^35\), CO\(_2\) sequestration \(^36\), and coastal flooding \(^37\).

Within the domain of seismology, neural operators were also recently used to learn general solution operators to the 2-D acoustic wave equation, a simplified case of the elastic wave equation \(^38\). This pilot study demonstrated that it was possible for a single FNO model to predict a complete wavefield given an arbitrary velocity model and mesh discretization. The success of this limited case highlights the potential of these methods, however, extending the methods from the acoustic wave equation to the elastodynamic case requires substantially increased model complexity. By comparison with neural networks, FNO is not considered to be a deep architecture, and is most analogous to the fully-connected neural networks employed heavily until the 2010s. A U-shaped neural operator (U-NO) was recently proposed to enable very deep neural operators and facilitate fast training, data efficiency, and hyperparameter selection robustness \(^39\).

In this paper, we apply the U-NO architecture to full seismic waveform modeling. We train a U-NO model to learn a general solution operator to the 2D elastic wave equation and demonstrate that
the trained model enables fast and accurate simulation for arbitrary source locations, velocity structures and mesh discretization. The trained U-NO also allows for efficient full-waveform inversion with automatic differentiation.

**Results**

**Neural operator development and training**

We developed a framework that applies U-NO to the 2D elastic wave equation. The architecture is depicted schematically in Figure 1. U-NO takes the source location and P- and S-wave velocities as inputs, which are then passed through a point-wise lifting operator. A sequence of non-linear integral operators (encoders) are applied that gradually contract the physical domain size, while simultaneously increasing the number of channels in the co-domain. These operators are followed by a sequence of non-linear integral operators (decoders) that progressively expand the physical domain, and decrease the number of channels. Finally, a point-wise projection operator leads to the output function. For the integral operators, we use FNO, which performs the integration in the frequency domain through fast convolutions. The output of the U-NO model is the complete horizontal and vertical displacement wavefield function over the medium domain, which can be queried at any mesh points desired, regardless of the input mesh used.

We generate a training dataset of FDM simulations using source locations, S-velocity models (Vs), and Vp-Vs ratios (Vp/Vs) that are all randomly generated. The U-NO is trained in a supervised manner with the objective of learning the general solution operator to the wave equation for arbitrary inputs (see Materials and Methods for details). Once completely trained, the U-NO model can be evaluated on a new input with very little computational cost (0.02 sec compared to the FDM runtime of 1.23 sec). The number of training simulations is the main factor in the computational cost, and we show that the U-NO model trained on a dataset of 5000 simulations can already predict the major phase arrivals of a new arbitrary velocity model. Increasing the dataset size from 5000 to 20000 provides better fit to the amplitudes (Figure 2). With a training dataset of 20000 simulations, the validation and training loss are very close, indicating there is no overfitting of the training data.

**Generalizability to arbitrary velocity structure or discretization**

The U-NO model is trained on random velocity models generated with the von Karman correlation function, which can best mimic the Earth’s heterogenous velocity distribution. We show by example that the U-NO model, although trained on random velocity models with some certain parameters, is applicable to arbitrary velocity models. These outcomes are in fact expected from theoretical grounds because von Karman random fields are able to approximate any physical function to arbitrary accuracy.

Our first example is with velocity models from a von Karman-type random distribution, but with a different covariance function than the one used for the training data. We increase the roughness of the velocity structure by a factor of four by decreasing the correlation length of Vs and Vp/Vs to only one-fourth that of the training data. As shown in Figure 3, the wavefield snapshot has more coda than with the smoother models because of the scattering from increased heterogeneity. However, the coda waves are well modeled by U-NO when compared to the ground truth simulation by FDM.

The velocity models used in the training data do not have coherent structures with discontinuities as in the real Earth, but wavefields for such models can still be simulated with our method. Figure 4 shows a simple model with a dipping ‘slab’ embedded in a homogeneous background. The slab has 20% higher Vs and 5% lower Vp/Vs. The wavefield snapshot shows that the reflections from
the high velocity anomaly are clearly predicted by U-NO. A more complex example is shown in Figure 5, where a random subpanel of the Marmousi model, a 2D synthetic structure model with complex vertical and horizontal variation used in exploration studies 43, is used. The reflected and refracted waves are very complicated due to the presence of folding and faulting, but the U-NO predictions still closely approximate the numerical solutions (Figure 5).

One of the most important advantages of a neural operator compared with a neural network is its mesh-free nature, since it intrinsically learns the mapping between function spaces. A model trained on any particular mesh can be evaluated on any other mesh, even at finer spacing. The example in Figure 6 shows the U-NO trained on a grid of 64*64 nodes applied to an input velocity model with 160*160 nodes. Here the input velocity model and the output wavefield can be seen in a much higher resolution, yet U-NO provides comparable prediction with the FDM solver. Note that if the resolution is increased by a factor of 2, a grid-based numerical solver like FDM takes about 6 times greater computational time; however, the evaluation using U-NO takes only about 2.5 times longer, providing additional computational efficiency.

We evaluate the overall generalization performance of the trained U-NO by performing a thousand random realizations on each of these cases. The distribution of the relative L2-norm misfit and cross-correlation coefficient are plotted in Figure 7. In the case of the Marmousi model, the extended tail of the histogram is attributed to the model's imbalanced complexity. In general, however, we see a very high cross-correlation coefficient (>0.95) between U-NO prediction and ground truth, confirming its robust generalizability.

**Application to full-waveform inversion**

One of the most important applications of wavefield simulations is in full-waveform inversion (FWI), which uses the full recorded waveform to image the Earth’s interior. The adjoint-state method is the traditional approach to compute the gradients of an objective function with respect to parameters of interest 4,5. Neural operators are differentiable by design, which enables gradient computation with reverse-mode automatic differentiation. It has been shown that automatic differentiation and the adjoint approach are mathematically equivalent 26. Hence, the trained U-NO model allows for convenient FWI and the associated speed and accuracy should depend only on the forward modeling part.

We demonstrate the inversion performance using the velocity structure of random subpanels in the Marmousi model 43. The synthetic waveform data are simulated with FDM 40 using 14 sources distributed in a ring shape. In Figure 8, we use the true source location, receivers on all 64*64 grids and noise-free waveform data; the goal here is not to demonstrate resolution, but rather the computational accuracy of the method. We then invert for Vp and Vs simultaneously by starting with homogeneous initial Vp and Vs models and forward propagating the wavefield with the U-NO for each source. The misfit is defined by the mean square error between the forward modeled and true wavefield. The gradient of the misfit with respect to Vp and Vs can be computed through automatic differentiation. Vp and Vs are then iteratively updated with gradient descent for 100 iterations using the Adam optimizer 44 with a learning rate of 0.01. Each iteration takes only about 1.4 sec by taking advantage of U-NO forward computation. The results in Figure 8 show a relative L2-norm misfit between the true and inverted model of only 3%. This successful inversion, in turn, further validates the accuracy of forward modeling with U-NO.

Besides the fact that the inversion target velocity model is quite different from the smooth random fields in the training dataset, this experiment itself is difficult due to conventional problems in full-waveform inversion, such as cycle skipping (multiple local maxima in the least-squares misfit
function). We also show that if we only use 64 receivers on the surface, the inversion results in the region with ray path coverage is still recovered reasonably well (Figure 9). Inversion with a biased homogenous initial model is also capable of producing relatively accurate results (see Supplementary Materials).

Discussion
We use the relative L2 loss between the FDM and U-NO predictions to evaluate the performance of the trained model for generalization. The relative L2 loss is defined as the L2-norm of the difference between the prediction and ground truth divided by the L2-norm of the ground truth. This ratio is used to evaluate the performance of the trained model. When using the same mesh discretization as the training data, the relative L2 loss is around 10–20 percent, but this number rises to 30–40 percent when the tests are performed on finer grids (Figure 7). These values are misleading, however, because the relative L2 loss imposes equal weights to the entire sparse matrix of waveforms that is dominated by small amplitudes close to 0. Alternatively, the cross-correlation coefficient is a quantity that is more sensitive to the seismic phases with amplitudes larger than background noise. A cross-correlation coefficient larger than 0.95 suggests the coherence of the U-NO prediction is excellent, even for the scenarios with large relative L2 loss (Figure 7). In addition, the FWI results confirm that the large L2 loss is not so important since even challenging models can still be properly recovered (Figure 8).

The major limitation of the approach is that the training process is compute- and memory-intensive. The cost of training for the 2D case is tractable on a single GPU. In the example presented here, evaluation using the trained U-NO takes 3 times the GPU memory of the FDM approach. For the extension from 2D to 3D modeling, the computation and memory will increase due to the larger dataset and the larger number of parameters to learn. Therefore, the next step is to enhance data compression and parallelization to accelerate the training process and reduce the storage. Since this is a learning-based approach, the model performance can be improved by fine-tuning the model parameters and increasing the size of the training dataset. More importantly, any future advancements made in neural operator model architectures will be able to be directly incorporated into the system as they occur. For example, the improvement from linear layers of FNO to U-NO enables faster training convergence. As a result, we should only take current performance metrics as a starting point.

Besides the more than an order of magnitude higher speed, the most important advantage of the neural operator-based full waveform modeling is its generalizability to arbitrary velocity models or discretization. This is because the neural operator learns a general solution operator to the wave equation instead of a specific instance of input velocity models. Once the neural operator is trained, it can be used by the entire seismology community for any region of a similar size without the need for retraining. Since the full waveform modeling with neural operator has easily accessible gradients for convenient FWI, we anticipate that this approach will eventually make FWI as affordable as travel time tomography.

Methods
Neural operator learning
Neural operators can make finite-dimensional neural network parameters independent of a mesh. Operators are mapping between function spaces, and the purpose of operator training is to find the map (operator) given a dataset of input-output pairs. Let $A$ denote the input coefficients (e.g., source properties and elastic structure parameters), $U$ denote the output solution to the seismic wave equation (e.g., displacement). The solution operator can be expressed by


\[ U(x) = \int G(x, y) A(y) dy, \]

where \( G \) is the Green’s function. To learning the mapping function \( G \) through data-driven methods, we start with the dataset \( \{ a_i(x), u_i(x) \} \) that are drawn from \( A \) and \( U \). The neural operator generalizes equation (1) to the sequence:

\[
\begin{align*}
    v_0(x) &= P(a(x)) \\
    v_{l+1}(x) &= \sigma(W_l v_l(x) + \int \kappa_l(x, y) v_l(y) dy), \quad l = 0, \ldots, L - 1 \\
    u(x) &= Q(v_L(x))
\end{align*}
\]

where \( a \) and \( u \) are data points in the input function space \( A \), and output function space \( U \), respectively, \( v_i \) is the input function at the \( l \)th layer, \( P \) is the point-wise lifting operator working on the input, \( Q \) is the point-wise projection operator to output, \( W_l \) is a linear transformation, \( \sigma \) is a point-wise non-linear activation function, \( \kappa_l \) is a kernel function that acts along with the integral as a global linear operator.

Due to the expense of evaluating integral operators, neural operators may lack the efficiency of convolutional or recurrent neural networks in finite-dimensional settings. The Fourier neural operator (FNO) was proposed to mitigate this difficulty through the fast Fourier transform \(^{34}\). The kernel integral operator in (2) can be replaced by a convolution operator defined in Fourier space:

\[
\int \kappa_l(x, y) v_l(y) dy = \mathcal{F}^{-1}(\mathcal{F}(\kappa_l) \cdot \mathcal{F}(v_l))
\]

where \( \mathcal{F} \) and \( \mathcal{F}^{-1} \) denote Fourier Transform and its inverse. However, FNO imposes that each layer is a mapping between functions spaces with identical domain spaces, which may cause a large memory usage. The U-shaped neural operator (U-NO), an analogy to the U-net architectures, was proposed to allow progressively transforming the input function space with respect to a sequence of varying domains \(^{39,45}\). After the lifting operator \( P \), a sequence of \( L_1 \) non-linear integral operators \( G_l \) is applied to \( v_0 \) and map the input to a set of functions with decreasing dimensional domain. Then a sequence of \( L_2 \) non-linear integral operators \( G_l \) is applied to \( v_{L_1+1} \) and map the input to a set of functions with increasing dimensional domain before the projection operator \( Q \). Skip connections \(^{45}\) are included to add vector-wise concatenation of \( v_{L_1+i} \) and \( v_{L_L-i} \). The contracting and expanding paths are symmetric. The architecture of the U-NO used in this study is illustrated in Figure 1, and we refer the interested readers to the references \(^{29-32,34,39}\) for more details.

**Numerical simulation**

We set up a training dataset of random source locations, S-wave velocity (Vs) models, and P- to S-wave velocity ratios (Vp/Vs). We define the velocity model on a \( 64 \times 64 \) mesh with 0.16 km grid spacing. The source is set as an isotropic explosive source randomly distributed on the mesh. The Vs has an average background of 3 km/s and perturbed by random fields with a von Kármán covariance function with the following parameters: Hurst exponent \( \kappa = 0.5 \), correlation length \( a_x = a_y = 8 \) grids, and the fractional magnitude of the fluctuation \( \varepsilon = 10\% \) background velocity. The power spectral density function of the von Kármán type random field follows a power law (fractal randomness) and can accurately represent the distribution of Earth’s heterogeneity \(^{41}\). The Vp/Vs is simplified to an average background of 1.732 perturbed by a smooth Gaussian random field with the following parameters: correlation length \( \lambda = 32 \) grids, standard deviation \( \sigma = 2\% \) background. A total of 20,000 random sets of models are generated and each of them is input to a GPU based 2D finite difference code in Cartesian coordinates to simulate the 2-D displacement field \(^{40}\). For simulation, the top boundary is set with a free-surface boundary condition and the other three edges have absorbing boundary conditions. A total of 4-sec wavefield with a time step
of 0.01 sec is simulated. Each simulation takes about 1.23 sec with a GPU memory usage of 0.3 GB.

**U-NO model training**

We describe the detailed parameters used in U-NO below following the notations in Figure 1. The goal is to learn an operator mapping from the input function $a$ with size $X_{in} \times Y_{in} \times T_{in}$ to the output function $u$ with size $X_{out} \times Y_{out} \times T_{out} \times C_{out}$. Here we set $X_{in} = Y_{in} = X_{out} = Y_{out} = 64$ for the training mesh, $T_{in} = 3$ representing source, Vs, and Vp/Vs distribution on the mesh, $T_{out} = 128$ for 32 Hz data output, and $C_{out} = 2$ representing two displacement components (horizontal and vertical). This work applies the U-NO architecture designed for mapping between 3-D spatio-temporal function domains ($x, y, t$) without any recurrent composition in time 39. The fourth dimension $C_{out}$ of the output function $u$ can be created in the last step through the projection operator $Q$. Constructing the operator to learn the mapping between 3-D spatio-temporal function domains:

$$G: \{a: [0,1]^2 \times [0, T_{in}] \rightarrow \mathbb{R}^{d_A}\} \rightarrow \{u: [0,1]^2 \times [0, T_{out}] \rightarrow \mathbb{R}^{d_U}\},$$

(4)

The operators $\{G_i\}_{i=0}^L$ as shown in Figure 1. that are used to construct the U-NO are defined as:

$$G_i: \{v_i: [0, \alpha_i]^2 \times T_i \rightarrow \mathbb{R}^{d_{v0}}\} \rightarrow \{v_{i+1}: [0, c_i^2 \alpha_i]^2 \times c_i^2 T_i \rightarrow \mathbb{R}^{c_i^2 d_{v0}}\},$$

(5)

where $[0, \alpha_i]^2 \times T_i$ is the domain of the input function $v_i$ to the operator $G_i$, and $c_i^2$, $c_i^t$, and $c_i^t$ are the expansion or contraction factors for spatial domain, temporal domain, and co-domain for $i$th operator, respectively. Note that $T_0 = [0, T_{in}]$, $\alpha_0 = \alpha_{L+1} = 1$, and $T_{L+1} = [0, T_{out}]$. In this work we set $L=8$, $c_{1,2} = \frac{3}{4}$, $c_{3,4} = \frac{1}{2}$, $c_{5,6} = 2$, $c_{7,8} = \frac{4}{3}$, $c_{1}^t = \frac{3}{4}$, $c_{2}^t = \frac{2}{3}$, $c_{3,4}^t = \frac{1}{2}$, $c_{5,6}^t = 2$, $c_{7,8}^t = \frac{3}{2}$, $c_{8}^t = \frac{4}{3}$, $c_{1,2,3,4}^t = 2$, $c_{5,6,7,8}^t = \frac{1}{2}$. The lifting operator $P$ to convert the input to a higher dimension channel space is set as a fully connected neural network with channel number $d_0 = 16$. The projection operator $Q$ to the output domain is also a fully connected neural network. The activation function used in each FNO block is the Gaussian Error Linear Units (GELUs) 46.

With the simulation dataset and the U-NO design, we train the U-NO model in a supervised manner. We divide the training dataset into 90% training and 10% validation. The model is trained with a batch size of 8. After hyperparameter tuning, the loss function we use in model training is the 90% relative L1 loss plus 10% relative L2 loss. The incorporation of L1-norm loss is more robust to outliers. We use Adam optimizer 44 with learning rate of $10^{-3}$ and a weight decay of $10^{-5}$. We trained for 100 epochs, which takes approximately 40 minutes per epoch using a single NVIDIA Tesla V100 GPU with 24GB memory usage. A 70% of loss decrease is achieved in the first 10 epochs. Once the U-NO model is trained, the model parameters require GPU usage of 3.8 GB and the time for an evaluation on a new source and velocity model takes only 0.02 sec with GPU usage of 0.9 GB.

**References**

1. Graves, R. W. & Pitarka, A. Broadband ground-motion simulation using a hybrid approach. *Bulletin of the Seismological Society of America* **100**, 2095–2123 (2010).

2. Rodgers, A. J. *et al.* Broadband (0-5 Hz) fully deterministic 3D ground-motion simulations of a magnitude 7.0 Hayward fault earthquake: Comparison with empirical ground-motion
models and 3D path and site effects from source normalized intensities. *Seismological Research Letters* **90**, 1268–1284 (2019).

3. Graves, R. & Pitarka, A. Kinematic ground-motion simulations on rough faults including effects of 3D stochastic velocity perturbations. *Bulletin of the Seismological Society of America* **106**, 2136–2153 (2016).

4. Fichtner, A., Kennett, B. L. N., Igel, H. & Bunge, H. P. Full seismic waveform tomography for upper-mantle structure in the Australasian region using adjoint methods. *Geophysical Journal International* **179**, 1703–1725 (2009).

5. Tape, C., Liu, Q., Maggi, A. & Tromp, J. Adjoint tomography of the southern california crust. *Science* **325**, 988–992 (2009).

6. Gebraad, L., Boehm, C. & Fichtner, A. Bayesian Elastic Full-Waveform Inversion Using Hamiltonian Monte Carlo. *Journal of Geophysical Research: Solid Earth* **125**, (2020).

7. Olsen, K. B. Site Amplification in the Los Angeles Basin from Three-Dimensional Modeling of Ground Motion. *Bulletin of the Seismological Society of America* **90**, S77–S94 (2000).

8. Kelly, K. R., Ward, R. W., Treitel, S. & Alford, R. M. Synthetic seismograms: a finite-difference approach. *Geophysics* **41**, 2–27 (1976).

9. Igel, H., Nissen-Meyer, T. & Jahnke, G. Wave propagation in 3D spherical sections: effects of subduction zones. *Physics of the Earth and Planetary Interiors* **132**, 219–234 (2002).

10. Fichtner, A., Igel, H., Bunge, H. P. & Kennett, B. L. N. Simulation and inversion of seismic wave propagation on continental scales based on a spectral-element method. *Journal of Numerical Analysis, Industrial and Applied Mathematics* **4**, 11–22 (2009).

11. Liu, Q. & Gu, Y. J. Seismic imaging: From classical to adjoint tomography. *Tectonophysics* **566–567**, 31–66 (2012).

12. Komatitsch, D. & Tromp, J. Spectral-element simulations of global seismic wave propagation-I. Validation. *Geophysical Journal International* **149**, 390–412 (2002).
13. Komatitsch, D. & Tromp, J. Spectral-element simulations of global seismic wave propagation-II. Three-dimensional models, oceans, rotation and self-gravitation. *Geophysical Journal International* **150**, 303–318 (2002).

14. Zhu, W., Mousavi, S. M. & Beroza, G. C. Seismic Signal Denoising and Decomposition Using Deep Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing* **57**, 9476–9488 (2019).

15. Yang, L., Liu, X., Zhu, W., Zhao, L. & Beroza, G. C. Toward improved urban earthquake monitoring through deep-learning-based noise suppression. *Science Advances* **8**, 3564 (2022).

16. Zhu, W. & Beroza, G. C. PhaseNet: A deep-neural-network-based seismic arrival-time picking method. *Geophysical Journal International* **216**, 261–273 (2019).

17. Mousavi, S. M., Ellsworth, W. L., Zhu, W., Chuang, L. Y. & Beroza, G. C. Earthquake transformer—an attentive deep-learning model for simultaneous earthquake detection and phase picking. *Nature Communications* **11**, 1–12 (2020).

18. Ross, Z. E., Meier, M. A. & Hauksson, E. P Wave Arrival Picking and First-Motion Polarity Determination With Deep Learning. *Journal of Geophysical Research: Solid Earth* **123**, 5120–5129 (2018).

19. Ross, Z. E., Yue, Y., Meier, M. A., Hauksson, E. & Heaton, T. H. PhaseLink: A Deep Learning Approach to Seismic Phase Association. *Journal of Geophysical Research: Solid Earth* **124**, 856–869 (2019).

20. Zhu, W., Tai, K. S., Mousavi, S. M., Bailis, P. & Beroza, G. C. An End-To-End Earthquake Detection Method for Joint Phase Picking and Association Using Deep Learning. *Journal of Geophysical Research: Solid Earth* **127**, 1–13 (2022).

21. Siahkoohi, A., Louboutin, M. & Herrmann, F. J. Neural network augmented wave-equation simulation. (2019) doi:10.48550/arxiv.1910.00925.
22. Moseley, B., Nissen-Meyer, T. & Markham, A. Deep learning for fast simulation of seismic waves in complex media. *Solid Earth* **11**, 1527–1549 (2020).

23. Moseley, B., Markham, A. & Nissen-Meyer, T. Fast approximate simulation of seismic waves with deep learning. (2018).

24. Moseley, B., Markham, A. & Nissen-Meyer, T. Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations. (2021).

25. Smith, J. D., Azizzadenesheli, K. & Ross, Z. E. EikoNet: Solving the Eikonal Equation with Deep Neural Networks. *IEEE Transactions on Geoscience and Remote Sensing* **59**, 10685–10696 (2021).

26. Zhu, W., Xu, K., Darve, E. & Beroza, G. C. A general approach to seismic inversion with automatic differentiation. *Computers & Geosciences* **151**, 104751 (2021).

27. Zhang, X. & Curtis, A. Bayesian Geophysical Inversion Using Invertible Neural Networks. *Journal of Geophysical Research: Solid Earth* **126**, e2021JB022320 (2021).

28. Rasht-Behesht, M., Huber, C., Shukla, K. & Karniadakis, G. E. Physics-Informed Neural Networks (PINNs) for Wave Propagation and Full Waveform Inversions. *Journal of Geophysical Research: Solid Earth* **127**, 1–21 (2022).

29. Lu, L., Jin, P. & Karniadakis, G. E. DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *Nature Machine Intelligence* **3**, 218–229 (2019).

30. Bhattacharya, K., Hosseini, B., Kovachki, N. B. & Stuart, A. M. Model Reduction and Neural Networks for Parametric PDEs. *SMAI Journal of Computational Mathematics* **7**, 121–157 (2020).

31. Li, Z. *et al.* Multipole Graph Neural Operator for Parametric Partial Differential Equations. *Advances in Neural Information Processing Systems* **2020-December**, (2020).
32. Li, Z. et al. Neural Operator: Graph Kernel Network for Partial Differential Equations. (2020) doi:10.48550/arxiv.2003.03485.

33. Kovachki, N. et al. Neural Operator: Learning Maps Between Function Spaces. http://arxiv.org/abs/2108.08481 (2021).

34. Li, Z. et al. Fourier Neural Operator for Parametric Partial Differential Equations. (2020) doi:10.48550/arxiv.2010.08895.

35. Pathak, J. et al. FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators. (2022) doi:10.48550/arxiv.2202.11214.

36. Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A. & Benson, S. M. U-FNO—An enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources* **163**, 104180 (2022).

37. Jiang, P. et al. Digital Twin Earth – Coasts: Developing a fast and physics-informed surrogate model for coastal floods via neural operators. (2021) doi:10.48550/arxiv.2110.07100.

38. Yang, Y. et al. Seismic Wave Propagation and Inversion with Neural Operators. *The Seismic Record* **1**, 126–134 (2021).

39. Rahman, M. A., Ross, Z. E. & Azizzadenesheli, K. U-NO: U-shaped Neural Operators. (2022) doi:10.48550/arxiv.2204.11127.

40. Li, D., Helmberger, D., Clayton, R. W. & Sun, D. Global synthetic seismograms using a 2-D finite-difference method. *Geophysical Journal International* **197**, 1166–1183 (2014).

41. Nakata, N. & Beroza, G. C. Stochastic characterization of mesoscale seismic velocity heterogeneity in Long Beach, California. *Geophysical Journal International* **203**, 2049–2054 (2015).

42. Mai, P. M. & Beroza, G. C. A Spatial Random Field Model to Characterize Complexity in Earthquake Slip. *Journal of Geophysical Research: Solid Earth* **107**, ESE 10-1-ESE 10-21 (2002).
43. Versteeg, R. The Marmousi experience: Velocity model determination on a synthetic complex data set. https://doi.org/10.1190/1.1437051 13, 927–936 (2012).

44. Kingma, D. P. & Ba, J. L. Adam: A Method for Stochastic Optimization. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2014) doi:10.48550/arxiv.1412.6980.

45. Ronneberger, O., Fischer, P. & Brox, T. U-net: Convolutional networks for biomedical image segmentation. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9351, 234–241 (2015).

46. Hendrycks, D. & Gimpel, K. Gaussian Error Linear Units (GELUs). (2016) doi:10.48550/arxiv.1606.08415.

Figures

Figure 1. Overview of our method for solving the elastic wave equation with neural operators. The inputs, $a$, to the U-NO model are the P- and S-wave velocity model and the source location (indicated by the white star). The outputs, $u$, are the horizontal and vertical displacements. Orange
circles denote point-wise operators, rectangles denote general operators, and smaller blue circles denote concatenations in function space. Typical roughness of the velocity models in the training dataset is shown in input and output images.

**Figure 2 Model performance as a function of the number of training samples.** Left panel: relative L2 loss curves for the training and validation data. Right panel: Example of simulated waveform comparison between FDM (black solid) and U-NO (red dashed).
Figure 3 Model generalization experiments 1: random fields of Vs and Vp/Vs model with 4-times roughness of the training data. Relative L2 loss of the U-NO simulation is 0.182.
Figure 4 Model generalization experiments 2: a homogeneous background model embedded with a ‘slab’ with 20% higher $V_s$ and 5% lower $V_p/V_s$. Relative L2 loss of the U-NO simulation is 0.090.
Figure 5 Model generalization experiments 3: a random subpanel from the Marmousi model. The velocity perturbation range is normalized to 30% of the average velocity. Relative L2 loss of the U-NO simulation is 0.225.
Figure 6 Model generalization experiments 4: Vp and Vs model mesh discretization is increased from 64*64 to 160*160. Relative L2 loss of the U-NO simulation is 0.385.
**Figure 7 Output evaluation.** Distribution of Relative L2 loss (top) and correlation coefficient (bottom) between the U-NO predictions and ground truth. From left to right, the columns are corresponding to the experiments in Figure 3, 5, 6. The red and black dashed lines mark the mean and standard deviation of the histograms.

**Figure 8 Full waveform inversion.** The inversion for a random subpanel from the Marmousi model. 14 sources are placed in a ring shape and receivers are placed at every node of the 64*64 grid. From left to right, the columns represent true velocity model, initial model for inversion, inverted model without regularization, and inverted model with damping and smoothing. The top and bottom rows are Vs and Vp models, respectively.
Figure 9 Full waveform inversion. Same as Figure 8, but the receivers are only placed on the 64 grids on the surface (blue line on the top of each subpanel). The gray shaded areas mask the areas without ray path coverage.
Figure S1 Cross-correlation of the waveform simulated by FDM and U-NO. Velocity model and source location are the same as those in Figure 3. The horizontal and vertical displacement waveforms at each grid are cross-correlated and the maximum cross-correlation value and its associated time shift is shown.
Figure S2 Same as Figure S1, but for the generalization example in Figure 4
Figure S3 Same as Figure S1, but for the generalization example in Figure 5.
**Figure S4** Same as Figure S1, but for the generalization example in Figure 6.
Figure S5 Full-waveform inversion example. The true model is the same as Figure 8 but we use an initial model with homogeneous velocity 10% higher than the average value of the true model for inversion. The misfit curve of the inversion is shown in the lower right panel.
Figure S6 Same as Figure S5 but for an initial model with homogeneous velocity 10% lower than the average value of the true model.