Communications in Mathematics and Statistics (2023) 11:3–19
https://doi.org/10.1007/s40304-022-00329-z

Solving Traveltime Tomography with Deep Learning

Yuwei Fan[1] · **Lexing Ying**[1]

Received: 6 February 2022 / Revised: 7 July 2022 / Accepted: 5 October 2022 /
Published online: 24 February 2023

Abstract
This paper introduces a neural network approach for solving two-dimensional traveltime tomography (TT) problems based on the eikonal equation. The mathematical problem of TT is to recover the slowness field of a medium based on the boundary measurement of the traveltimes of waves going through the medium. This inverse map is high-dimensional and nonlinear. For the circular tomography geometry, a perturbative analysis shows that the forward map can be approximated by a vectorized convolution operator in the angular direction. Motivated by this and filtered back-projection, we propose an effective neural network architecture for the inverse map using the recently proposed BCR-Net, with weights learned from training datasets. Numerical results demonstrate the efficiency of the proposed neural networks.

Keywords Traveltime tomography · Eikonal equation · Inverse problem · Neural networks · Convolutional neural network

Mathematics Subject Classification 65J22 · 65N21 · 74J25

1 Introduction

Traveltime tomography is a method to determinate the internal properties of a medium by measuring the traveltimes of waves going through the medium. It is first motivated in global seismology in determining the inner structure of the Earth by measuring at different seismic stations the traveltimes of seismic waves produced by earthquakes [5, 59]. By now, it has found many applications, such as Sun's interior [42], ocean acoustics [53], and ultrasound tomography [36, 62] in biomedical imaging.

✉ Lexing Ying
lexing@stanford.edu

Yuwei Fan
ywfan1989@gmail.com

[1] Department of Mathematics, Stanford University, Stanford, CA 94305, USA

Background The governing equation of first-arrival traveltime tomography (TT) is the eikonal equation [9] and we consider the two-dimensional case in this paper for simplicity. Let $\Omega \subset \mathbb{R}^2$ be an open bounded domain with Lipschitz boundary $\Gamma = \partial\Omega$. Suppose that the positive function $m(x)$ is the slowness field, i.e., the reciprocal of the velocity field defined in Ω. The traveltime $u(x)$ satisfies the eikonal equation $|\nabla u(x)| = m(x)$. Since it is a special case of the Hamilton–Jacobi equation, the solution $u(x)$ can develop singularities and should be understood in the viscosity sense [35].

A typical experimental setup of TT is as follows: For each source point $x_s \in \Gamma$, one sets up the Soner boundary condition at x_s, i.e., zero value at x_s, and solves for the following the eikonal equation

$$\begin{aligned} |\nabla u^s(x)| &= m(x), \quad x \in \Omega, \\ u^s(x_s) &= 0, \end{aligned} \tag{1.1}$$

where the superscript s is to index the source point. Recording the solution of $u^s(\cdot)$ at receiver points $\{x_r\} \subset \Gamma$ produces the whole data set $\{u^s(x_r)\}_{s,r}$. In practice, x_r and x_s are samples from a discrete set of points on Γ. Here, we assume for now that they are placed everywhere on Γ, for the simplicity of presentation and mathematical analysis.

The forward problem is to compute $u^s(x_r)$ given the slowness field $m(x)$. On the other hand, the inverse problem, at the center of the first-arrival TT, is to recover $m(x)$ given $u^s(x_r)$.

Both the forward and inverse problems are computationally challenging, and a lot of efforts have been devoted to their numerical solutions. For the forward problem, the eikonal equation, as a special case of the Hamilton–Jacobi equation, can develop singular solutions. In order to compute the physically meaningful viscosity solution, special care such as up-winding is required. As the resulting discrete system is nonlinear, fast iteration methods such as fast marching method [54, 64] and fast sweeping method [38, 56, 70] have been developed. Among them, the fast sweeping methods have been successfully applied to many traveltime tomography problems [47]. The inverse problem is often computationally more intensive, due to the nonlinearity of the problem. Typical methods take an optimization approach with proper regularization [11] and require a significant number of iterations.

A deep learning approach Over the past decade or so, deep learning (DL) has become the dominant approach in computer vision, image processing, speech recognition, and many other applications in machine learning and data science [28, 33, 43, 45, 46, 51, 61, 65]. From a technical point of view, this success is a synergy of several key developments: neural networks (NNs) as a flexible framework for representing high-dimensional functions and maps, simple algorithms such as back-propagation (BP) and stochastic gradient descent (SGD) for tuning the model parameters, efficient general software packages such as TensorFlow and PyTorch, and unprecedented computing power provided by GPUs and TPUs.

In the past several years, deep neural networks (DNNs) have been increasingly used in scientific computing, particularly in solving PDE-related problems [4, 7, 21, 24, 30, 39, 44, 57], in two directions. In the first direction, as NNs offer a powerful tool for approximating high-dimensional functions [14], it is natural to use them as an ansatz for high-dimensional PDEs [10, 18, 30, 40, 60]. The second direction focuses on the low-dimensional parameterized PDE problems, by using the DNNs to represent the nonlinear map from the high-dimensional parameters of the PDE solution [6, 19–21, 31, 39, 48, 49].

As an extension of the second direction, DNNs have been widely applied to inverse problems [2, 22, 23, 34, 37, 41, 50, 58, 66]. For the inverse problem, DNNs can help in two critical ways: (1) due to its flexibility in representing high-dimensional functions, DNNs can potentially be used to approximate the full inverse map, thus avoiding the iterative solution process; (2) recent work in machine learning shows that DNNs often can automatically extract features from the data and offer a data-driven regularization prior.

This paper applies the deep learning approach to the first-arrival TT by representing the whole inverse map using an NN. The starting point is a perturbative analysis of the forward map, which reveals that for the circular tomography geometry, the forward map contains a one-dimensional convolution with multiple channels, after appropriate (polar) reparameterization. This observation motivates to represent the forward map from 2D coefficient $m(x)$ to the boundary data $\{u^s(x_r)\}_{s,r}$ (with sources $\{x_s\}$ and receivers $\{x_r\}$) by a one-dimensional convolution neural network (with multiple channels). Further, the one-dimensional convolution neural network can be implemented by the recently proposed multiscale neural networks [19, 21]. Following the idea of filtered back-projection [63], the inverse map can be approximated by the adjoint map followed by a pseudo-differential filtering step. This suggests an architecture for the inverse map by reversing the architecture of the forward map followed with a simple two-dimensional convolution neural network. For the test problems being considered, the resulting neural networks have 10^5 parameters when the data are of size 160×160 (a fully-connected layer results in $160^4 \approx 6 \times 10^8$ parameters), thanks to the convolutional structure and the compact multiscale neural network. This rather small number of parameters allows for rapid and accurate training, even on rather limited data sets.

The approach followed by this paper relies on the harmonic analysis and PDE theory of the wave equations. A more optimization-based deep learning approach for inverse problems is the unrolling method [2, 15, 26, 32, 52, 55, 69], where one writes the iterative solution algorithm as a ResNet and then trains the network parameters to minimize the reconstruction error. In many cases, this approach also leads to high quality reconstructions. There is also active work studying stability issues when applying deep learning to inverse problems [3, 13, 25, 29], which is particularly important for applications with ill-posed inverse problems.

Organization This rest of the paper is organized as follows: The mathematical background is given in Sect. 2. The design and architecture of the DNNs of the forward and inverse maps are discussed in Sect. 3. The numerical results in Sect. 4 demonstrate the numerical efficiency and the generalization of the proposed neural networks.

2 Mathematical Analysis of Traveltime Tomography

2.1 Problem Setup

This section describes the necessary mathematical insights that motivate the NN architecture design. Let us consider the so-called differential imaging setting, where a background slowness field $m_0(x)$ is known, and denote by u_0^s the solution of the eikonal equation associated with the field m_0:

$$\begin{aligned} |\nabla u_0^s(x)| &= m_0(x), \quad x \in \Omega, \\ u_0^s(x_s) &= 0. \end{aligned} \tag{2.1}$$

Then for a perturbation $\tilde{m}$ to the slowness field, the difference in the traveltime $\tilde{u}^s \equiv u^s - u_0^s$ naturally satisfies

$$\begin{aligned} |\nabla(u_0^s(x) + \tilde{u}^s(x))| &= m_0(x) + \tilde{m}(x), \quad x \in \Omega, \\ \tilde{u}^s(x_s) &= 0. \end{aligned} \tag{2.2}$$

The imaging data $d(x_s, x_r)$ consist of $\tilde{u}^s(x_r)$ over all x_s and x_r: $d(x_s, x_r) \equiv \tilde{u}^s(x_r)$.

To better understand the dependence of $\tilde{u}^s$ on $\tilde{m}$, we assume $\tilde{m}$ to be sufficient small and carry out a perturbative analysis. Squaring Eq. 2.2 and canceling the background using Eq. 2.1 result in

$$(\nabla\tilde{u}^s(x))^{\mathsf{T}}\nabla\tilde{u}^s(x) + 2(\nabla u_0^s(x))^{\mathsf{T}}\nabla\tilde{u}^s(x) = \tilde{m}(x)^2 + 2m_0(x)\tilde{m}(x). \tag{2.3}$$

Since $\tilde{m}(x)$ is sufficiently small, $\nabla\tilde{u}^s(x)$ is also a small quantity. Keeping only linear terms in $\tilde{m}$ and discarding the higher order ones yields

$$\nabla u_0^s(x)^{\mathsf{T}}\nabla\tilde{u}^s(x) \approx m_0(x)\tilde{m}(x), \tag{2.4}$$

which is an advection equation. Using $|\nabla u_0^s(x)^{\mathsf{T}}| = m_0(x)$, one can further simplify the upper equation as

$$\widehat{\nabla u_0^s(x)}^{\mathsf{T}}\nabla\tilde{u}^s(x) \approx \tilde{m}(x), \tag{2.5}$$

where $\widehat{\cdot}$ stands for the unit vector.

For simplicity, let $\mathcal{C}_0(x_s, x_r)$ be the unique characteristic of $u_0^s(x_r)$ that connects x_s and x_r. Then,

$$d(x_s, x_r) \equiv \tilde{u}^s(x_r) \approx \int_{\mathcal{C}_0(x_s,x_r)} \tilde{m}(x)\,\mathrm{d}x \equiv d_1(x_s, x_r), \tag{2.6}$$

where $d_1(x_s, x_r)$ is introduced to stand for the first-order approximation to $d(x_s, x_r)$. Particularly, if the background slowness field is a constant, then $\mathcal{C}_0(x_s, x_r)$ is a line

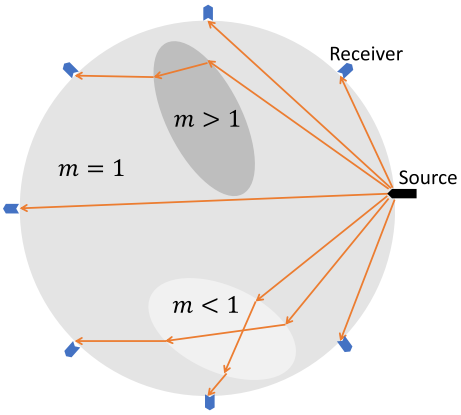


Fig. 1 Illustration of the problem setup. The domain is a unit disk and the light sources and the receivers are equidistantly placed on the boundary

segment with start and end points to be x_s and x_r, respectively, and

$$d_1(x_s, x_r) = |x_s - x_r| \int_0^1 \tilde{m}(x_s + \tau(x_r - x_s))\, \mathrm{d}\tau.$$

The most relevant geometry in traveltime tomography either for medicine and earth science is the circular geometry where Ω is modeled as a unit disk [11, 16, 68]. As illustrated in Fig. 1, the sources and receivers are placed on the boundary equidistantly. More precisely, $x_s = (\cos(s), \sin(s))$ with $s = \frac{2\pi k}{N_s}$, $k = 0, \ldots, N_s - 1$ and $x_r = (\cos(r), \sin(r))$ with $r = \frac{2\pi j}{N_r}$, $j = 0, \ldots, N_r - 1$, where $N_s = N_r$ in the current setup.

Often in many cases, the background slowness field $m_0(x)$ is only radially dependent, or even a constant [16, 68]. In what follows, $m_0(x)$ is assumed to be radially dependent, i.e., $m_0(x) = m_0(|x|)$.

2.2 Mathematical Analysis of the Forward Map

Since the domain Ω is a disk, it is convenient to rewrite the problem in the polar coordinates. Let $x_r = (\cos(r), \sin(r))$, $x_s = (\cos(s), \sin(s))$ and $x = (\rho\cos(\theta), \rho\sin(\theta))$, where $\rho \in [0, 1]$ is the radial coordinate and $r, s, \theta \in [0, 2\pi)$ are the angular ones.

Figure 2 presents an example of the slowness field and the measurement data. Notice that the main signal in $u^s(x_r)$ and $d(x_s, x_r)$ concentrates on the minor diagonal part. Due to the circular tomography geometry, it is convenient to "shear" the measurement data by introducing a new angular variable $h = r - s$, where the difference here is understood modulus 2π. As we shall see in the next section, this shearing step significantly simplifies the architecture of the NNs. Under the new parameterization, the measurement data are

$$d(s, h) \equiv d(x_s, x_{s+h}). \tag{2.7}$$

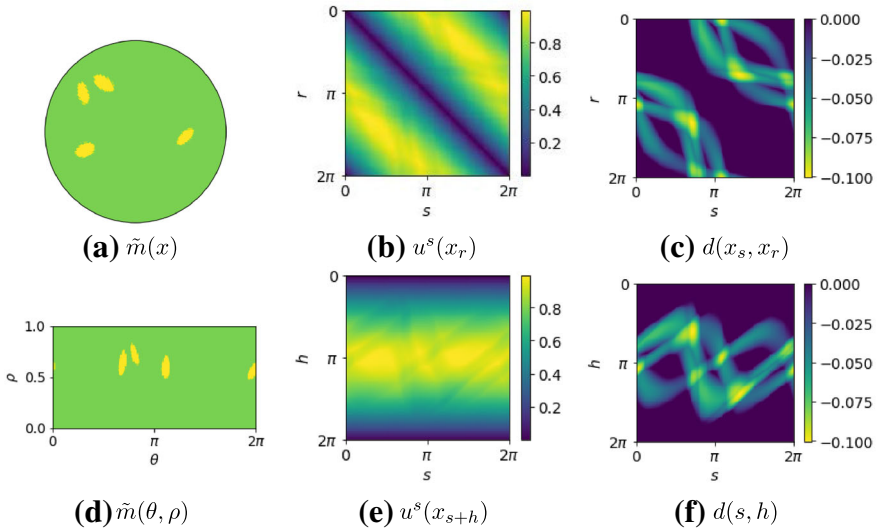


Fig. 2 Visualization of the slowness field and the measurement data. The upper figures are the perturbation of the slowness field $\tilde{m}(x)$ ($m_0 = 1$ and $\tilde{m} \leq 0$ in this sample), the measurement data $u^s(x_r)$ and the difference $d(x_s, x_r)$ with respect to the background measurement data. The lower-left figure is $\tilde{m}(x)$ in the polar coordinates and the lower-right two figures the "shear" of their corresponding upper figures

The same convention applies to its first-order approximation: $d_1(s, h) \equiv d_1(x_s, x_{s+h})$. By writing $\tilde{m}(\theta, \rho) \equiv \tilde{m}(\rho\cos(\theta), \rho\sin(\theta))$ in the polar coordinates, the linear dependence of $d_1(s, h)$ on $\tilde{m}$ in (2.6) states that there exists a kernel distribution $K(s, h, \theta, \rho)$ such that

$$d_1(s, h) = \int_0^1 \int_0^{2\pi} K(s, h, \theta, \rho)\tilde{m}(\theta, \rho)\, \mathrm{d}\rho\, \mathrm{d}\theta. \tag{2.8}$$

Convolution form of the map $m(\theta, \rho) \to d_1(s, h)$. Since the domain is a disk and m_0 is only radially independent, the whole problem is equivariant to rotation. In this case, the situation can be dramatically simplified. Precisely, we have the following proposition.

Proposition 2.1 *There exists a function $\kappa(h, \rho, \cdot)$ periodic in the last parameter such that*

$$d_1(s, h) = \int_0^1 \int_0^{2\pi} \kappa(h, \rho, s - \theta)\tilde{m}(\theta, \rho)\, \mathrm{d}\rho\, \mathrm{d}\theta. \tag{2.9}$$

Proof Let $\mathcal{C}_0(s, r) \equiv \mathcal{C}_0((\cos(s), \sin(s)), (\cos(r), \sin(r)))$ and we parameterize the characteristic $\mathcal{C}_0(s, r)$ as $p_{s,h}(\tau) \equiv (\theta_{s,h}(\tau), \rho_{s,h}(\tau))$, $\tau \in [0, 1]$ with $\rho_{s,h}(0) = \rho_{s,h}(1) = 1$ and $\theta_{s,h}(0) = s, \theta_{s,h}(1) = r$. Then, the relationship (2.6) between d_1 and

$\tilde{m}$ can be written as

$$d_1(s,h) = \int_0^1 \tilde{m}(\theta_{s,h}(\tau), \rho_{s,h}(\tau)) \| p'_{s,h}(\tau) \| \, \mathrm{d}\tau.$$

Since the background slowness m_0 is radially independent, the characteristic $\mathcal{C}_0(s,r)$ is rotation invariant in the sense that for any $\phi \in [0, 2\pi)$, if $(\theta_{s,h}(\tau), \rho_{s,h}(\tau))$ is a parameterization of $\mathcal{C}_0(s,r)$, then $(\theta_{s,h}(\tau) + \phi, \rho_{s,h}(\tau))$ is a parameterization of $\mathcal{C}_0(s+\phi, r+\phi)$. Hence, for any $\phi \in [0, 2\pi)$, if we rotate the system by a angular ϕ, then

$$\begin{aligned} d_1(s+\phi,h) &= \int_0^1 \tilde{m}(\theta_{s+\phi,h}(\tau), \rho_{s+\phi,h}(\tau)) \| p'_{s+\phi,h}(\tau) \| \, \mathrm{d}\tau \\ &= \int_0^1 \tilde{m}(\theta_{s,h}(\tau) + \phi, \rho_{s,h}(\tau)) \| p'_{s,h}(\tau) \| \, \mathrm{d}\tau. \end{aligned}$$

Writing this equation in the form of (2.8) directly yields $K(s+\phi, h, \theta+\phi, \rho) = K(s,h,\theta,\rho)$. Hence, there is a periodic $\kappa(s,h,\cdot)$ in the last parameter such that $K(s,h,\theta,\rho) = \kappa(h,\rho,s-\theta)$. This completes the proof. □

Proposition 2.1 shows that K acts on $\tilde{m}$ in the angular direction by a convolution, which is, in fact, the motivation behind shearing the measurement data d. This property allows us to evaluate the map $\tilde{m}(\theta,\rho) \to d(s,h)$ by a family of 1D convolutions, parameterized ρ and h.

Discretization All the above analysis is in the continuous space. One can apply a discretization on the eikonal equation Eq. 1.1 by finite difference and solve it by fast sweeping method or fast marching method. Here, we assume that the discretization of $\tilde{m}(\theta,\rho)$ is on a uniform mesh on $[0, 2\pi) \times [0, 1]$. More details of the discretization and the numerical solver will be discussed in the Sect. 4. With a slight abuse of notation, we use the same letters to denote the continuous kernels, variables and their discretization. Then, the discretization version of Eqs. 2.8,2.9 is

$$d(s,h) \approx \sum_{\rho,\theta} K(s,h,\theta,\rho)\tilde{m}(\theta,\rho) = \sum_{\rho} (\kappa(h,\rho,\cdot) * \tilde{m}(\cdot,\rho))(s). \tag{2.10}$$

3 Neural Networks for TT

In this section, we describe the NN architecture for the inverse map $d(s,h) \to \tilde{m}(\theta,\rho)$ based on the mathematical analysis in Sect. 2. To start, we first study the NN for the forward map and then the inverse map.

Forward map The perturbative analysis in Sect. 2.2 shows that, when $\tilde{m}$ is sufficiently small, the forward map $\tilde{m}(\theta,\rho) \to d(s,h)$ can be approximated by Eq. 2.10. In terms of the NN architecture, for small $\tilde{m}$, the forward map Eq. 2.10 can be approximated by a (non-local) convolution on the angular direction and a fully-connected operator

on the (h, ρ) direction. In the actual implementation, it can be represented by the convolution layer by taking h and ρ as the channel dimensions. For larger $\tilde{m}$, this linear approximation is no longer accurate. In order to extend the neural network for Eq. 2.10 to the nonlinear case, we propose to increase the number of convolution layers and nonlinear activation functions.

In the (h, ρ) direction, denote the number of channels by c, whose value is problem-dependent and will be discussed in the numerical part. In the angular direction, since the convolution between $\tilde{m}$ and d is global, in order to represent global interactions the window size of the convolution w must satisfy the following relationship

$$w N_{\text{cnn}} \geq N_\theta, \tag{3.1}$$

where N_{cnn} is the number of layers and N_θ is number of discretization points on the angular direction. A simple calculation shows that the number of parameters of the neural network is $O(wN_{\text{cnn}}c^2) \sim O(N_\theta c^2)$.

In a recent work [19], BCR-Net has been proposed as an alternative with built-in multiscale structure. At a high level, BCR-Net is motivated by the data-sparse non-standard wavelet representation of the pseudo-differential operators [8]. It processes the information at different scale separately and each scale can be understood as a local convolutional neural network. It has been demonstrated to require fewer number of parameters and provide better efficiency for such global interactions. Therefore, in our architecture, we replace the convolution layers with the BCR-Net.

Data: $c, N_{\text{cnn}} \in \mathbb{N}^+, \tilde{m} \in \mathbb{R}^{N_\theta \times N_\rho}$
Result: $d \in \mathbb{R}^{N_s \times N_h}$
`/* Resampling data to fit for BCR-Net. */`
$\xi = Conv1d[c, 1, id](\tilde{m})$ with ρ as the channel direction
`/* Use BCR-Net to implement the convolutional neural network. */`
$\zeta = BCR - Net[c, N_{\text{cnn}}](\xi)$
`/* Reconstruct the result from the output of BCR-Net. */`
$d = Conv1d[N_h, 1, id](\zeta)$

Algorithm 1: Neural network architecture for the forward map $\tilde{m} \to d$.

The resulting neural network architecture for the forward map is summarized in Algorithm 1 with an estimate of $O(c^2 \log(N_\theta) N_{\text{cnn}})$ parameters. The components are explained in the following.

- $\xi = Conv1d[c, w, \phi](m)$ mapping $m \in \mathbb{R}^{N_\theta \times N_\rho}$ to $\xi \in \mathbb{R}^{N_\theta \times c}$ is the one-dimensional convolution layer with window size w, channel number c, activation function ϕ and period padding on the first direction.
- The one-dimensional $\zeta = BCR - Net[c, N_{\text{cnn}}](\xi)$ maps $\xi \in \mathbb{R}^{N_\theta \times c}$ to $\zeta \in \mathbb{R}^{N_\theta \times c}$, where the number of channels and layers in the local convolutional neural network in each scale are c and N_{cnn}, respectively. The readers are referred to [19] for more details on the BCR-Net.

Inverse map The perturbative analysis in Sect. 2.2 shows that if $\tilde{m}$ is sufficiently small, the forward map can be approximated by $d \approx K\tilde{m}$, the operator notation of the

discretization Eq. 2.10. Here, $\tilde{m}$ is a vector indexed by (θ, ρ), d is a vector indexed by (s, h), and K is a matrix with row indexed by (s, h) and column indexed by (θ, ρ).

The filtered back-projection method [63] suggests the following formula to recover $\tilde{m}$:

$$\tilde{m} \approx (K^{\mathsf{T}}K + \epsilon I)^{-1} K^{\mathsf{T}} d, \tag{3.2}$$

where ϵ is a regularization parameter. The first piece $K^{\mathsf{T}}d$ can also be written as a family of convolutions

$$(K^{\mathsf{T}}d)(\theta, \rho) = \sum_h (\kappa(h, \rho, \cdot) * d(\cdot, h))(\theta). \tag{3.3}$$

The application of K^{T} to d can be approximated with a similar neural network to K in Algorithm 1. The second piece $(K^{\mathsf{T}}K + \epsilon I)^{-1}$ is a pseudo-differential operator in the (θ, ρ) space and it is implemented with several two-dimensional convolutional layers for simplicity. Then, the resulting architecture for the inverse map is summarized in Algorithm 2 and illustrated in Figure 3. The $Conv2d[c_2, w, \phi]$ used in Algorithm 2 is the two-dimensional convolution layer with window size w, channel number c_2, activation function ϕ and periodic padding on the first direction and zero padding on the second direction. The selection of the hyper-parameters in Algorithm 2 will be discussed in Sect. 4.

Data: $c, c_2, w, N_{\text{cnn}}, N_{\text{cnn2}} \in \mathbb{N}^+$, $d \in \mathbb{R}^{N_s \times N_h}$
Result: $\tilde{m} \in \mathbb{R}^{N_\theta \times N_\rho}$
/* Application of K^{T} to d */
$\zeta = Conv1d[c, 1, id](d)$ with h as the channel direction
$\xi = BCR - Net[c, N_{\text{cnn}}](\zeta)$
$\xi^{(0)} = Conv1d[N_\rho, 1, id](\xi)$
/* Application of $(K^{\mathsf{T}}K + \epsilon I)^{-1}$ */
for k *from* 1 *to* $N_{\text{cnn2}} - 1$ **do**
  $\xi^{(k)} = Conv2d[c_2, w, ReLU](\xi^{(k-1)})$
end
$\tilde{m} = Conv2d[1, w, id](\xi^{(N_{\text{cnn2}}-1)})$

Algorithm 2: Neural network architecture for the inverse problem $d \to \tilde{m}$.

4 Numerical Tests

This section reports the numerical performance of the proposed neural network architecture in Algorithm 2 for the inverse map $d \to \tilde{m}$.

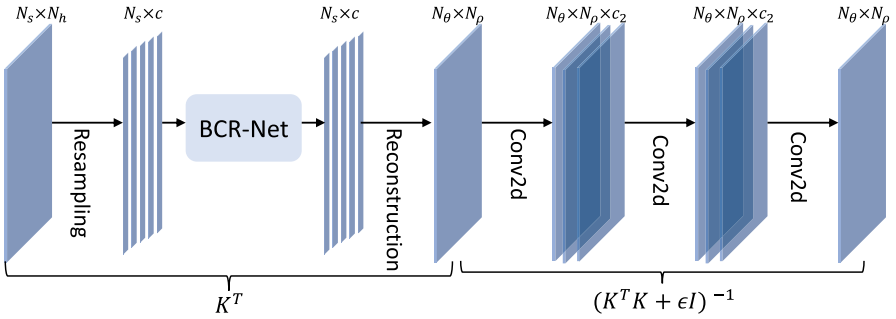


Fig. 3 Neural network architecture for the inverse map of TT

4.1 Experimental Setup

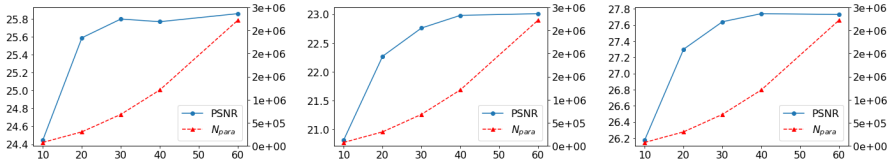
In order to solve the eikonal equation Eq. 1.1 on the unit disk Ω, we embed Ω into the square domain $[-1, 1]^2$ by specifying sufficiently large slowness values outside Ω. The domain $[-1, 1]^2$ is discretized with a uniform Cartesian mesh with 160 points in each direction by a finite difference scheme. The fast sweeping method proposed in [70] is used to solve the nonlinear discrete system. In the polar coordinates, the domain $(\theta, \rho) \in [0, 2\pi) \times [0, 1]$ is partitioned by uniformly Cartesian mesh with 160×80 points, i.e., $N_\theta = 160$ and $N_\rho = 80$. As $\tilde{m}(\theta, \rho)$ used in Algorithm 2 is in the polar coordinates while the eikonal equation is solved in the Cartesian ones, and the perturbation of the slowness field $\tilde{m}$ is treated as a piecewise linear function in the domain Ω and is interpolated on to the polar grid. The number of sources and receivers as $N_s = N_r = 160$, and hence $N_h = 160$.

The NN in Algorithm 2 is implemented with Keras running on top of TensorFlow [1]. All the parameters of the network are trainable and initialized by Xavier initialization [27]. The loss function is the mean squared error, and the optimizer is the Nadam [17].

During the training process, the batch size and the learning rate is firstly set as 32 and 10^{-3}, respectively, and the NN is trained 100 epochs. One then increases the batch size by a factor 2 till 512 with the learning rate unchanged, and then decreases the learning rate by a factor $10^{1/2}$ to 10^{-5} with the batch size fixed as 512. In each step, the NN is trained with 50 epochs. For the hyper-parameters used in Algorithm 2, $N_{\text{cnn}} = 6$, $N_{\text{cnn2}} = 5$, and $w = 3 \times 3$. The selection of the channel number c will be studied later.

4.2 Results

For a fixed $\tilde{m}$, $d(s, h)$ stands for the exact measurement data solved by numerical discretization of Eq. 1.1. The prediction of the NN from d is denoted by $\tilde{m}^{\text{NN}}$. The metric for the prediction is the peak signal-to-noise ratio (PSNR), which is defined as



(a) Negative inclusions **(b)** Positive inclusions **(c)** Mixture inclusions

Fig. 4 The test PSNR for different channel numbers c for the three types of data with $N_e = 4$

$$\text{PSNR} = 10\log_{10}\left(\frac{\text{Max}^2}{\text{MSE}}\right), \quad \text{Max} = \max_{i,j}(\tilde{m}_{i,j}) - \min_{i,j}(\tilde{m}_{i,j}), \quad \text{MSE}$$
$$= \frac{1}{N_\theta N_\rho}\sum_{i,j}|\tilde{m}_{i,j} - \tilde{m}_{i,j}^{\text{NN}}|^2. \tag{4.1}$$

For each experiment, the test PSNR is then obtained by averaging Eq. 4.1 over a given set of test samples. The numerical results presented below are obtained by repeating the training process five times, using different random seeds for the NN initialization.

The numerical experiments focus on the shape reconstruction setting [16, 67], where $\tilde{m}$ are often piecewise constant inclusions. The background slowness field is set as $m_0 \equiv 1$ and the slowness field $\tilde{m}$ is assumed to be the sum of N_e piecewise constant ellipses. As the slowness field m is positive, it is required that $\tilde{m} > -1$. For each ellipse, the direction is uniformly random over the unit circle, the position is uniformly sampled in the disk, and the width and height depend on the datasets. It is also required that each ellipse lies in the disk and there is no intersection between every two ellipses. Three types of data sets are generated to test the neural network.

- Negative inclusions. $\tilde{m}$, the perturbation of the slowness, is -0.5 in the ellipses and 0 otherwise, and the width and height of each ellipse are sampled from the uniform distributions $\mathcal{U}(0.1, 0.2)$ and $\mathcal{U}(0.05, 0.1)$, respectively.
- Positive inclusions. $\tilde{m}$ is 2 in the ellipses and 0 otherwise, and the width and height of each ellipse are sampled from $\mathcal{U}(0.2, 0.4)$ and $\mathcal{U}(0.1, 0.2)$, respectively.
- Mixture inclusions. The setup of each ellipse is either a negative one in the negative inclusions or a positive one in the positive inclusions.

For each type, we generate two datasets with the number of inclusions $N_e = 2$ and 4. For each test, 20480 samples $\{(\tilde{m}_i, d_i)\}$ are generated with 16384 used for training and the remaining 4096 for testing.

The first numerical study is concerned with the choice of channel number c in Algorithm 2. Figure 4 presents the test PSNR and the number of parameters with different channel number c for three types of data sets with $N_e = 4$. As the channel number c increases, the test PSNR first consistently increases and then saturates for all the three types of data. Notice that the number of parameters of the neural network is $O(c^2 \log(N_\theta) N_{\text{cnn}})$. The choice of $c = 30$ is a reasonable balance between accuracy and efficiency and the total number of parameters is 684K.

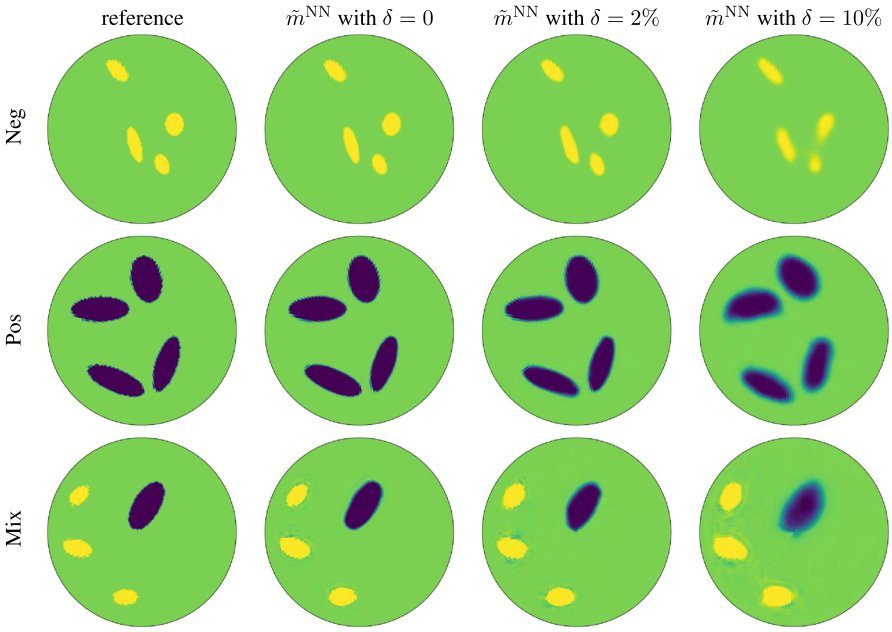


Fig. 5 NN prediction of a sample in the test data for negative (first row) / positive (second row) / mixture (third row) inclusions with $N_e = 4$ for different noise level $\delta = 0$, 2% and 10%

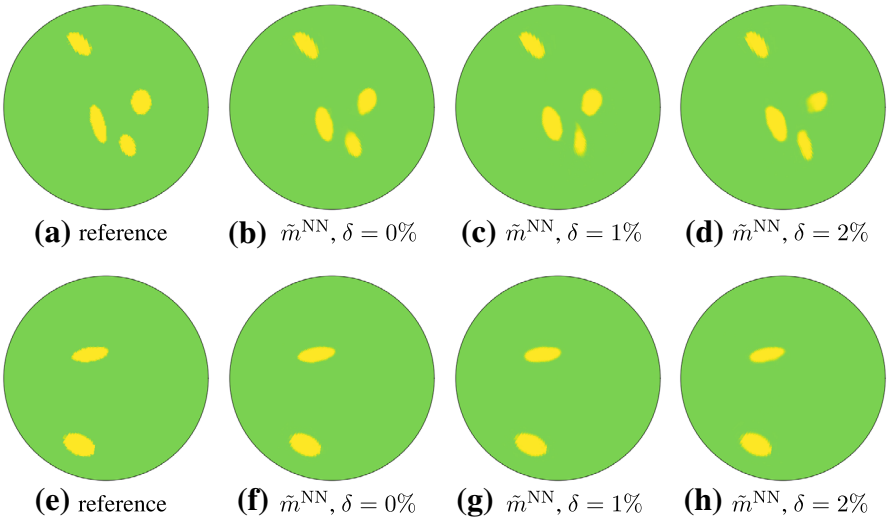


Fig. 6 NN generalization test for the negative inclusions. The upper (or lower) figures: the NN is trained by the data of the number of ellipses $N_e = 2$ (or 4) with noise level $\delta = 0$, 1% or 2% and test by the data of $N_e = 4$ (or 2) with the same noise level

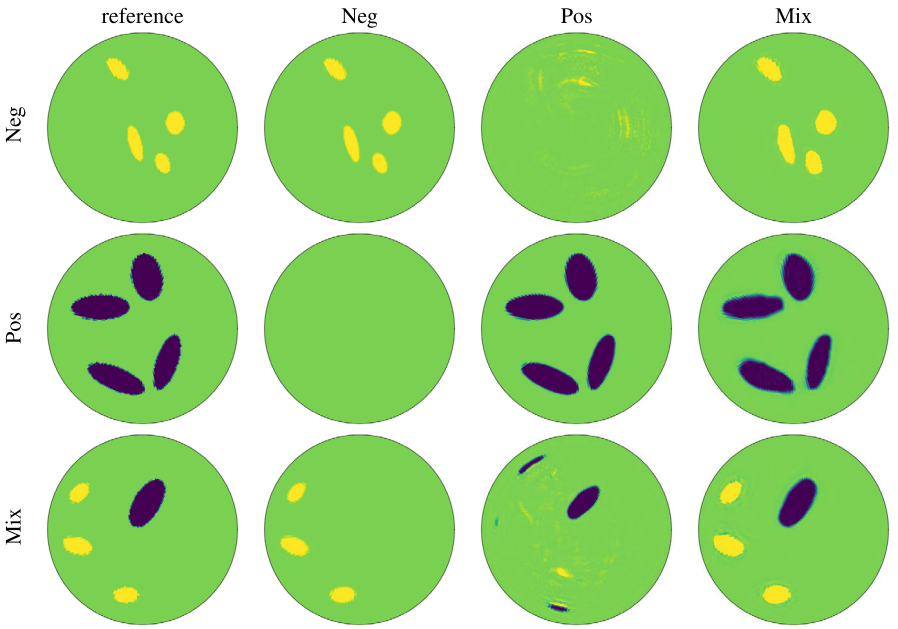


Fig. 7 NN generalization test for different types of data sets. The first column is the reference solution. In each column of the last three columns, the NN is trained with one data type (negative, positive, or mixed) and is tested on all three data types with $N_e = 4$ and without noise

To model the uncertainty in the measurement data, we introduce noises to the measurement data by defining $u^{s,\delta}(x_r) \equiv (1 + Z_i\delta)u^s(x_r)$, where Z_i is a Gaussian random variable with zero mean and unit variance and δ controls the signal-to-noise ratio. In terms of the actual data d of the differential imaging, $d^\delta(s,h) \equiv (1 + Z_i\delta)d(s,h) + Z_i\delta u_0^s(x_r)$. Notice that, since the mean of $\frac{\|u_0^s(x_r)\|}{\|d(s,h)\|}$ for all the samples lies in [15, 30] in these experiments, the signal-to-noise ratio for d is in fact more than $15 \cdot \delta$. For each noisy level $\delta = 0, 2\%, 10\%$, an independent NN is trained and tested with the noisy data set $\{(d_i^\delta, \tilde{m}_i)\}$.

Figure 5 collects, for different noise level δ, samples for all three data types: (1) negative inclusions with $N_e = 4$, (2) positive inclusions with $N_e = 4$, and (3) mixture inclusions with $N_e = 4$. The NN is trained with the datasets generated in the same way as the test data. When there is no noise in the measurement data, the NN consistently gives accurate predictions of the slowness field $\tilde{m}$, in the position, shape, and direction of the ellipse. For the small noise levels, for example, $\delta = 2\%$, the boundary of the shapes slightly blurs while the position and direction of the ellipse are still correct. As the noise level δ increases, the shapes become fuzzy but the position and number of shapes are always correct. This demonstrates the proposed NN architecture is capable of learning the inverse problem.

The next test is about the generalization of the proposed NN. We first train the NN by the data set of the negative inclusions with $N_e = 2$ (or 4) with noise level $\delta = 0$, 1% or 2% and test by the data of the negative inclusions with $N_e = 4$ (or 2) with

the same noise level. The results, presented in Figure 6, indicate that the NN trained by the data with two inclusions is capable of recovering the measurement data of the case with four inclusions and vice versa. This shows that the trained NN is capable of predicting beyond the training scenario.

The last test is about the prediction power of the NN on one data type while trained with another. In Figure 7, the first column is the reference solution. In each of the rest three columns, the NN is trained with one data type (negative, positive, or mixed) and is tested on all three data types, with $N_e = 4$ and without noise. The figures in the second column show that the NN trained by negative inclusions fails to capture the information of the positive inclusions, and vice versa, the third column demonstrates that the NN trained with positive inclusions fails for the negative inclusions. On the other hand, the NN trained with mixed inclusions is capable of predicting reasonably well for all three data types.

5 Discussions

This paper presents a neural network approach for the inverse problems of first-arrival traveltime tomography, by using the NN to approximate the whole inverse map from the measurement data to the slowness field. The perturbative analysis, which indicates that the linearized forward map can be represented by a one-dimensional convolution with multiple channels, inspires the design of the whole NN architectures. The analysis in this paper can also be extended to the three-dimensional TT problems by leveraging recent work such as [12].

Acknowledgements The work of Y.F. and L.Y. is partially supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program. The work of L.Y. is also partially supported by the National Science Foundation under award DMS-1818449.

References

1. Abadi, M. et al.: Tensorflow: a system for large-scale machine learning., Osdi, 16, 265–283, (2016)
2. Adler, J., Öktem, O.: Solving ill-posed inverse problems using iterative deep neural networks. Inverse Probl. **33**(12), 124007 (2017)
3. Antun, V., Renna, F., Poon, C., Adcock, B., Hansen, A.C.: On instabilities of deep learning in image reconstruction and the potential costs of AI. Proceed. Nat. Acad. Sci. **117**(48), 30088–30095 (2020)
4. Araya-Polo, M., Jennings, J., Adler, A., Dahlke, T.: Deep-learning tomography. Lead. Edge **37**(1), 58–66 (2018)
5. Backus, G., Gilbert, F.: The resolving power of gross Earth data. Geophys. J. Int. **16**(2), 169–205 (1968)
6. Bar, L., Sochen, N.: Unsupervised deep learning algorithm for PDE-based forward and inverse problems. arXiv preprint. (2019). arXiv:1904.05417
7. Berg, J., Nyström, K.: A unified deep artificial neural network approach to partial differential equations in complex geometries. Neurocomputing **317**, 28–41 (2018)
8. Beylkin, G., Coifman, R., Rokhlin, V.: Fast wavelet transforms and numerical algorithms I. Commun. Pure Appl. Math. **44**(2), 141–183 (1991)
9. Born, M., Wolf, E.: Principles of optics: electromagnetic theory of propagation, interference and diffraction of light, 3, Oxford: Pergamon, (1965)

10. Carleo, G., Troyer, M.: Solving the quantum many-body problem with artificial neural networks. Science **355**(6325), 602–606 (2017)
11. Chung, E., Qian, J., Uhlmann, G., Zhao, H.: An adaptive phase space method with application to reflection traveltime tomography. Inverse Probl. **27**(11), 115002 (2011)
12. Cohen, T.S., Geiger, M., Köhler, J., Welling, M.: Spherical CNNs. In: International conference on learning representations. https://openreview.net/forum?id=Hkbd5xZRb, (2018)
13. Colbrook, M.J., Antun, V., Hansen, A.C.: Can stable and accurate neural networks be computed?–on the barriers of deep learning and smale's 18th problem. arXiv preprint. (2021). arXiv:2101.08286
14. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Signals Syst. **2**(4), 303–314 (1989)
15. de Hoop, M.V., Lassas, M., Wong, C.A.: Deep learning architectures for nonlinear operator functions and nonlinear inverse problems. arXiv preprint. (2019). arXiv:1912.11090
16. Deckelnick, K., Elliott, C.M., Styles, V.: Numerical analysis of an inverse problem for the eikonal equation. Numerische Mathematik **119**(2), 245 (2011)
17. Dozat, T.: Incorporating Nesterov momentum into adam. In: International Conference on Learning Representations. (2016)
18. E Weinan, Y.B.: The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun. Math. Stat. **6**(1), 1–12 (2018)
19. Fan, Y., Bohorquez, C.O., Ying, L.: BCR-Net: a neural network based on the nonstandard wavelet form. J. Comput. Phys. **384**, 1–15 (2019)
20. Fan, Y., Feliu-Fabà, J., Lin, L., Ying, L., Zepeda-Núñez, L.: A multiscale neural network based on hierarchical nested bases. Res. Math. Sci. **6**(2), 21 (2019)
21. Fan, Yuwei., Lin, Lin., Ying, Lexing., Zepeda-Núñez, Leonardo.: A multiscale neural network based on hierarchical matrices. arXiv preprint. (2018). arXiv:1807.01883
22. Fan, Y., Ying, L.: Solving electrical impedance tomography with deep learning. arXiv preprint. (2019). arXiv:1906.03944
23. Fan, Y., Ying, L.: Solving optical tomography with deep learning. arXiv preprint. (2019). arXiv:1910.04756
24. Feliu-Faba, J., Fan, Y., Ying, L.: Meta-learning pseudo-differential operators with deep neural networks. arXiv preprint. (2019). arXiv:1906.06782
25. Genzel, M., Macdonald, J., März, M.: Solving inverse problems with deep neural networks–robustness included?. arXiv preprint. (2020). arXiv:2011.04268
26. Gilton, D., Ongie, G., Willett, R.: Neumann networks for linear inverse problems in imaging. IEEE Trans. Comput. Imag. **6**, 328–343 (2019)
27. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. 249–256. (2010)
28. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep learning, MIT press Cambridge, 1. (2016)
29. Gottschling, N.M., Antun, V., Adcock, B., Hansen, A.C.: The troublesome kernel: why deep learning for inverse problems is typically unstable. arXiv preprint. (2020). arXiv:2001.01258
30. Han, J., Jentzen, A., E, W.: Solving high-dimensional partial differential equations using deep learning. Proceed. Nat. Academy Sci. **115**(34), 8505–8510 (2018)
31. Han, J., Zhang, L., Car, R., E, W.: Deep potential: a general representation of a many-body potential energy surface. Commun. Comput. Phys. **23**(3), 629–639 (2018)
32. Hauptmann, A., Adler, J., Arridge, S., Öktem, O.: Multi-scale learned iterative reconstruction. IEEE Trans. Comput. Imag. **6**, 843–856 (2020)
33. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process. Mag. **29**(6), 82–97 (2012)
34. Hoole, S., Ratnajeevan, H.: Artificial neural networks in the solution of inverse electromagnetic field problems. IEEE Trans. Magn. **29**(2), 1931–1934 (1993)
35. Ishii, H.: A simple, direct proof of uniqueness for solutions of the Hamilton-Jacobi equations of eikonal type. Proceed. Am. Math. Soc. (1987)
36. Jin, X., Wang, L.V.: Thermoacoustic tomography with correction for acoustic speed variations. Phys. Med. Biol. **51**(24), 6437 (2006)
37. Kabir, H., Wang, Y., Yu, M., Zhang, Q.J.: Neural network inverse modeling and applications to microwave filter design. IEEE Trans. Microw. Theory Tech. **56**(4), 867–879 (2008)

38. Kao, Chiu-Yen., Osher, S., Tsai, Y.H.: Fast sweeping methods for static Hamilton-Jacobi equations. SIAM J. Numer. Anal. **42**(6), 2612–2632 (2005)
39. Khoo, Y., Lu, J., Ying, L.: Solving parametric PDE problems with artificial neural networks. arXiv preprint. (2017). arXiv:1707.03351
40. Khoo, Y., Lu, J., Ying, L.: Solving for high-dimensional committor functions using artificial neural networks. Res Math Sci **6**(1), 1 (2019)
41. Khoo, Y., Ying, L.: SwitchNet: a neural network model for forward and inverse scattering problems. arXiv preprint. (2018). arXiv:1810.09675
42. Kosovichev, A.G.: Tomographic imaging of the Sun's interior. Astrophys. J. Lett. **461**(1), L55 (1996)
43. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - vol. 1, NIPS'12, Curran Associates Inc., USA, 1097–1105, (2012)
44. Kutyniok, G., Petersen, P., Raslan, M., Schneider, R.: A theoretical analysis of deep neural networks and parametric PDEs. arXiv preprint. (2019). arXiv:1904.00377
45. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**, 436 (2015)
46. Leung, M.K.K., Xiong, H.Y., Lee, L.J., Frey, B.J.: Deep learning of the tissue-regulated splicing code. Bioinformatics **30**(12), i121–i129 (2014)
47. Leung, S., Qian, J.: An adjoint state method for three-dimensional transmission traveltime tomography using first-arrivals. Commun. Math. Sci. **4**(1), 249–266 (2006)
48. Li, Y., Lu, J., Mao, A.: Variational training of neural network approximations of solution maps for physical models. arXiv preprint. (2019). arXiv:1905.02789
49. Long, Z., Lu, Y., Ma, X., Dong, B.: PDE-net: Learning PDEs from data. In: Proceedings of the 35th International Conference on Machine Learning, Dy, Jennifer, Krause, Andreas, Proceedings of Machine Learning Research, 80, PMLR, Stockholmsmässan, Stockholm Sweden. pp 3208–3216 (2018). http://proceedings.mlr.press/v80/long18a.html
50. Lucas, A., Iliadis, M., Molina, R., Katsaggelos, A.K.: Using deep neural networks for inverse problems in imaging: beyond analytical methods. IEEE Signal Process. Mag. **35**(1), 20–36 (2018)
51. Ma, J., Sheridan, R.P., Liaw, A., Dahl, G.E., Svetnik, V.: Deep neural nets as a method for quantitative structure-activity relationships. J. Chem. Inf. Model. **55**(2), 263–274 (2015)
52. Monga, V., Li, Y., Eldar, Y.C.: Algorithm unrolling: interpretable, efficient deep learning for signal and image processing. IEEE Signal Process. Mag. **38**(2), 18–44 (2021)
53. Munk, W., Worcester, P., Wunsch, C.: Ocean acoustic tomography, Cambridge university press, (2009)
54. Popovici, A.M., Sethian, J.: Three dimensional traveltime computation using the fast marching method, Seg technical program expanded abstracts 1997. Society of Exploration Geophysicists. 1778–1781 (1997)
55. Putzky, Patrick: Welling, Max, Invert to learn to invert. Adv. Neural Inf. Process. Syst. **32**, 446–456 (2019)
56. Qian, J., Zhang, Y.T., Zhao, H.K.: Fast sweeping methods for eikonal equations on triangular meshes. SIAM J. Numer. Anal. **45**(1), 83–107 (2007)
57. Raissi, M., Karniadakis, G.E.: Hidden physics models: machine learning of nonlinear partial differential equations. J. Comput. Phys. **357**, 125–141 (2018)
58. Raissi, M., Perdikaris, P., Karniadakis, G.E.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
59. Rawlinson, N., Pozgay, S., Fishwick, S.: Seismic tomography: a window into deep Earth. Phys. Earth Planet. Inter. **178**(3–4), 101–135 (2010)
60. Rudd, K., Ferrari, S.: A constrained integration (CINT) approach to solving partial differential equations using artificial neural networks. Neurocomputing **155**, 277–285 (2015)
61. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015). (**ISSN=0893-6080**)
62. Schomberg, H.: An improved approach to reconstructive ultrasound tomography. J. Phys. D Appl. Phys. **11**(15), L181 (1978)
63. Schuster, G.T., Quintus-Bosz, A.: Wavepath eikonal traveltime inversion: theory. Geophysics **58**(9), 1314–1323 (1993)
64. Sethian, J.A.: Fast marching methods. SIAM Rev. **41**(2), 199–235 (1999)

65. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems 27, Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., Curran Associates, Inc., 3104–3112, (2014)
66. Tan, C., Lv, S., Dong, F., Takei, M.: Image reconstruction based on convolutional neural network for electrical resistance tomography. IEEE Sens. J. **19**(1), 196–204 (2018)
67. Üstündag, D.: Retrieving slowness distribution of a medium between two boreholes from first arrival traveltimes. Int. J. Geol 2, 1–8, (2008)
68. Yeung, T.S.A., Chung, E.T., Uhlmann, G.: Numerical inversion of 3d geodesic X-ray transform arising from traveltime tomography. arXiv preprint. (2018). arXiv:1804.10006
69. Zhang, H.M., Dong, B.: A review on deep learning in medical image reconstruction. J. Oper. Res. Soc. China (2020). https://doi.org/10.1007/s40305-019-00287-4
70. Zhao, H.: A fast sweeping method for eikonal equations. Math. Comput. **74**(250), 603–627 (2005)