Geoscience Frontiers 11 (2020) 1993–2001

HOSTED BY

ELSEVIER

Contents lists available at ScienceDirect

Geoscience Frontiers

journal homepage: www.elsevier.com/locate/gsf

Research Paper

Physics informed machine learning: Seismic wave equation

Sadegh Karimpouli[a], Pejman Tahmasebi[b,*]

[a] Mining Engineering Group, Faculty of Engineering, University of Zanjan, Zanjan, Iran
[b] Department of Petroleum Engineering, University of Wyoming, Laramie, WY, 82071, USA

A R T I C L E I N F O

Handling Editor: Masaki Yoshida

Keywords:
Gaussian process (GP)
Physics informed machine learning (PIML)
Seismic wave
Optimization

A B S T R A C T

Similar to many fields of sciences, recent deep learning advances have been applied extensively in geosciences for both small- and large-scale problems. However, the necessity of using large training data and the 'black box' nature of learning have limited them in practice and difficult to interpret. Furthermore, including the governing equations and physical facts in such methods is also another challenge, which entails either ignoring the physics or simplifying them using unrealistic data. To address such issues, physics informed machine learning methods have been developed which can integrate the governing physics law into the learning process. In this work, a 1-dimensional (1D) time-dependent seismic wave equation is considered and solved using two methods, namely Gaussian process (GP) and physics informed neural networks. We show that these meshless methods are trained by smaller amount of data and can predict the solution of the equation with even high accuracy. They are also capable of inverting any parameter involved in the governing equation such as wave velocity in our case. Results show that the GP can predict the solution of the seismic wave equation with a lower level of error, while our developed neural network is more accurate for velocity (P- and S-wave) and density inversion.

1. Introduction

Recent advances in recording data and computational power made a revolution in machine learning applications in many fields of sciences such as machine vision (Aslam et al., 2019; Nasirahmadi et al., 2019; Wang et al., 2020), speech recognition (Lee et al., 2019; Vishal and Aggarwal, 2019), geosciences (Waldeland et al., 2018; Xiong et al., 2018; Kamrava et al., 2019, 2019b; Karimpouli et al., 2019; Karimpouli and Tahmasebi, 2019a; Kong et al., 2019; Bai and Tahmasebi, 2020; Rouet-Leduc et al., 2020; Tang et al., 2020) and medical engineering (Bernal et al., 2019; Guo et al., 2019; Maitra et al., 2019) to name a few examples. Convolutional neural networks and their derivatives are one of those highly effective networks with high accuracy in both classifications (Garcia-Garcia et al., 2017), estimation problems (Kamrava et al., 2019a; Karimpouli and Tahmasebi, 2019b), the problems related to uncertainty quantification in complex (geo)-materials (Kamrava et al., 2020a), and also the problems related to fracture modeling and phase transition in complex materials (Kamrava et al., 2020b). A comprehensive review of the applications of the AI methods in geosciences at both small- and large-scale systems has been reviewed recently and can be found elsewhere (Tahmasebi et al., 2020). Thus, such methods are not reviewed here. These networks perform very well when a vast amount of data is available. However, in most engineering applications, if not all, data acquisition is an expensive and time-demanding task. Thus, one has to draw conclusions from partial information. In this case, deep networks fail to converge mostly due to the complexity of the system and unoptimized parameters.

In most cases, deep learning algorithms are assumed as a *black box*, usually without any contribution of prior knowledge of the system. This knowledge could be significant, varying from physical laws governing the time-dependent dynamic of a system to empirical relations derived from the experimental analysis. Prior information is assumed as tool to regularize the solution by constraining it into a physically meaningful space (Raissi et al., 2019). These constraints lead to rejecting nonrealistic solutions, known as *nonuniqueness* issue in inverse problems, and steer the learning process to the right solution. Algorithms that encode this information into their learning processes are called as *physics-constrained* (Zhu et al., 2019) or *Physics-Informed* (Raissi et al., 2019) methods, which are used either for data-driven estimation and/or parameter inversion. The engaged governing equations conduct quickly the learning process to the right solution and enable it to learn the whole rule between inputs and outputs even in a small data scheme.

* Corresponding author.
E-mail addresses: s.karimpouli@znu.ac.ir (S. Karimpouli), ptahmase@uwyo.edu (P. Tahmasebi).
Peer-review under responsibility of China University of Geosciences (Beijing).

https://doi.org/10.1016/j.gsf.2020.07.007
Received 30 April 2020; Received in revised form 14 June 2020; Accepted 15 July 2020
Available online 5 August 2020

Among all data-driven methods, Gaussian Process (GP) and physics-informed machine learning (PIML) are used here as intelligent *meshless* methods for solving the seismic wave equation. A Gaussian process is a collection of random variables, any Gaussian process finite number of which have a joint Gaussian distribution (Rasmussen and Williams, 2006). The GP, as a Bayesian procedure, assumes prior distributions over functions and estimates posterior covariance function by applying the governed physical lows (e.g., differential equation). It inherently quantifies the uncertainty using a fully probabilistic workflow and a powerful training procedure. The GPs could be categorized either as a kernel machine with regularization approaches (Vapnik, 2013) or one-layer feed-forward Bayesian neural networks with an infinite number of hidden neurons (Neal, 2012). Some researchers attempted to combine probabilistic models and machine learning and proposed the GP as a non-parametric Bayesian machine learning approach to solve differential equations (Lawrence, 2005; Titsias and Lawrence, 2010; Alvarez et al., 2013). In contrast to previous models, other researchers placed the GP prior to the solution of a differential equation where optimal model parameters and hyper-parameters are all learned directly from the data (Raissi et al., 2017a, b). They used such a GP model to solve ordinary and partial differential, integro-differential, and fractional-order operators.

In the second group, the PIMLs are a class of artificial neural networks (ANN), where the learning process is reinforced by physical governing laws. Although some researchers solved differential equations with ANN (Meade and Fernandez, 1994; Lagaris et al., 1998), recent studies instead have tried to revise the loss function based on the observed data as well as the governing equations, initial and boundary conditions (Raissi et al., 2019; Zhu et al., 2019; Haghighat et al., 2020). Toward this goal, PIML has been used for data-driven solution and discovery of a collection of classical problems in fluids, quantum mechanics, reaction-diffusion systems, and the propagation of nonlinear shallow-water waves (Raissi et al., 2019). They extended their framework for learning fluid velocity and pressure field from flow visualizations (Raissi et al., 2020). In a similar fashion, PIML is used in solid mechanics to predict 2D displacement of a plate under a nonuniform force (Haghighat et al., 2020). The physical laws in linear elasticity were encoded in a loss function to induce physical aspects of the problem. They found that PIML can predict a wide range of parameters, converges quickly when the network is re-training via transfer learning and can be used for sensitivity analysis and surrogate modeling.

In this study, we also have implemented PIML for solving the seismic wave equation as a fundamental equation, which has many applications in geosciences such as earthquake seismology, seismic exploration, rock physics and etc. In this work, we have compared two physics-informed deep learning methods, namely GP and PIML, for solving a 1D form of a linear differential equation of seismic wave and simultaneously for inversion of the wave velocity.

2. Methodology

2.1. Gaussian Process

Suppose $u(x)$ is a function of spatial or temporal variable x and a differential equation operator $\mathscr{L}_x^\alpha$ with a-priori unknown parameters of α acts on it. Thus, a differential equation is expressed as:

$$\mathscr{L}_x^\alpha u(x) = v(x) \tag{1}$$

where $v(x)$ is the forcing term. Thus, $u(x)$ is an unknown solution of differential equations, which is targeted to be found via GP. To this end, $u(x)$ is assumed to be a GP with mean 0 and covariance function of $k_{uu}(x, x^{'}; \beta)$ (Rasmussen and Williams, 2006):

$$u(x) \sim GP(0, k_{uu}(x, x^{'}; \beta)) \tag{2}$$

where β is the hyper-parameter of the kernel k_{uu}. The key point of GP is that it is closed under linear transformations such as summation, multiplication, differentiation, and integration. For example, since $u(x)$ is a GP, the obtained function under differential equation operator $\mathscr{L}_x^\alpha$ is still GP (Dondelinger et al., 2013; Cockayne et al., 2017; Raissi et al., 2017a):

$$v(x) \sim GP(0, k_{vv}(x, x^{'}; \alpha, \beta)) \tag{3}$$

where

$$k_{vv}(x, x^{'}; \alpha, \beta) = \left(\mathscr{L}_x^\alpha \mathscr{L}_{x'}^\alpha\, k_{uu}(x, x^{'}; \beta)\right) \tag{4}$$

and

$$k_{vu}(x, x^{'}; \alpha, \beta) = \left(\mathscr{L}_x^\alpha\, k_{uu}(x, x^{'}; \beta)\right) \tag{5}$$

$$k_{uv}(x, x^{'}; \alpha, \beta) = \left(\mathscr{L}_{x'}^\alpha\, k_{uu}(x, x^{'}; \beta)\right) \tag{6}$$

We assume a squared exponential although there are many covariance functions (Rasmussen and Williams, 2006):

$$k_{uu}(x, x^{'}; \beta) = \sigma_u^2 \exp\left[-\frac{1}{2}\sum_{i=1}^{N} \lambda_i (x_i - x'_i)^2\right] \tag{7}$$

where x (and/or x') is a N-dimensional vector in space/time and $\beta = \{\sigma_u^2,\ \lambda_i(i = 1 \ldots N)\}$ is a collection of hyper-parameters: signal variance and weights. This is a stationary covariance function in which the covariance is almost unity between variables whose corresponding inputs are very close, and decreases as their distance in the input space increases (Rasmussen and Williams, 2006).

Both parameter α and hyper-parameters β can be obtained by minimizing the negative log marginal likelihood $\mathscr{NLML}$ (Rasmussen and Williams, 2006) via an optimization process (Raissi et al., 2017b):

$$\mathscr{NLML} = -\log p(\mathbf{y}|\alpha, \beta, \sigma_u^2, \sigma_v^2) = \frac{1}{2}\log|\mathbf{K}| + \frac{1}{2}\mathbf{y}^T\mathbf{K}^{-1}\mathbf{y} + \frac{N}{2}\log 2\pi \tag{8}$$

where $\mathbf{y} = \begin{bmatrix}\mathbf{y}_u \\ \mathbf{y}_v\end{bmatrix}$, $p(\mathbf{y}|\alpha, \beta, \sigma_u^2, \sigma_v^2) = \mathscr{N}(0, \mathbf{K})$ and $\mathbf{K}$ is expressed as:

$$\mathbf{K} = \begin{bmatrix} k_{uu}(\mathbf{X}_u, \mathbf{X}_u; \beta) + \sigma_{n_u}^2 I_{n_u} & k_{vu}(\mathbf{X}_v, \mathbf{X}_u; \alpha, \beta) \\ k_{uv}(\mathbf{X}_u, \mathbf{X}_v; \alpha, \beta) & k_{vv}(\mathbf{X}_v, \mathbf{X}_v; \alpha, \beta) + \sigma_{n_v}^2 I_{n_v} \end{bmatrix} \tag{9}$$

where $\sigma_{n_u}^2$ and $\sigma_{n_v}^2$ are the noise variances to cover noisy data.

After training the model and obtaining the optimized parameters and hyper-parameters, one can predict $u(x)$ and $v(x)$ for new points x using posterior distributions:

$$p(u(x)|\mathbf{y}) = \mathscr{N}\left(\overline{u}(x), S_u^2\right) \tag{10}$$

$$p(v(x)|\mathbf{y}) = \mathscr{N}\left(\overline{v}(x), S_v^2\right) \tag{11}$$

where

$$\overline{u}(x) = \mathbf{z}_u^T\mathbf{K}^{-1}\mathbf{y}, \quad S_u^2 = k_{uu}(x, x) - \mathbf{z}_u^T\mathbf{K}^{-1}\mathbf{z}_u \tag{12}$$

$$\overline{v}(x) = \mathbf{z}_v^T\mathbf{K}^{-1}\mathbf{y}, \quad S_v^2 = k_{vv}(x, x) - \mathbf{z}_v^T\mathbf{K}^{-1}\mathbf{z}_v \tag{13}$$

and

$$\mathbf{z}_u^T = [k_{uu}(x, \mathbf{X}_u) k_{uv}(x, \mathbf{X}_v)] \tag{14}$$

$$\mathbf{z}_v^T = [k_{vu}(x, \mathbf{X}_u) k_{vv}(x, \mathbf{X}_v)] \tag{15}$$

The $\overline{u}(x)$ and $\overline{v}(x)$ are the predictions in each new point and S_u^2 and S_v^2 are the uncertainties of each prediction. These variances are accounted as a direct consequence of a Bayesian procedure, which can be used to quantify the uncertainty and to find new positions of data acquisition.

2.2. Physics informed machine learning

Suppose a differential equation such the one shown in Eq. (1). A conventional neural network $\mathscr{NN}(x; W, b)$ is used to approximate $u(x)$, where W, b are weights and biases of the network. Therefore, the network inputs are N-dimensional vector of x and the output is $u(x)$. In a feed-forward network, the value of each layer is computed from the last one as:

$$q^i = \mathscr{AF}^i\left(W^i x^i + b^i\right),\ i = 1\ldots L \tag{16}$$

where $\mathscr{AF}$ is the activation function and q vector contains the values of i-th layer. The process of a neural network is an optimization problem where weights and biases of the network are updated by minimizing a loss function. For example, Mean Square Error (MSE) is a well-known loss function and is defined as:

$$MSE = \frac{1}{N_u}\sum_{i=1}^{N_u}\left(u - u^*\right)^2 \tag{17}$$

where u and u^* are the ground truth and output value of the network. The updated weights and biases are obtained using an optimization function such as *adam* (Kingma and Ba, 2014):

$$(W_{new}, b_{new}) \leftarrow (W_{current}, b_{current}) - \xi\frac{\partial MSE}{\partial(W_{current}, b_{current})} \tag{18}$$

where ξ is the learning rate.

To induce physical differential equations into the network, one needs first to compute differentiation of $u(x)$ in the network. To this end, the network inputs must be space or time variables while differentiation of output is physically meaningful. Due to graph-based implementation of feed-forward networks, partial differentiation of $u(x)$ is naturally achieved at machine precision. Thus, differential equations and physics laws are involved in the loss function to penalize the network learning procedure. In fact, the outputs of the network must meet the physics informed loss function to achieve the best results. Consequently, the mean square error of the loss function can be written based on the network output (MSE_u), differential equation (MSE_v) and initial (MSE_0) and boundary (MSE_b) conditions (Raissi et al., 2019):

$$MSE = MSE_u + MSE_v + MSE_0 + MSE_b \tag{19}$$

and

$$MSE_u = \frac{1}{N_u}\sum_{i=1}^{N_u}\left(u - u^*\right)^2 \tag{20}$$

$$MSE_v = \frac{1}{N_v}\sum_{i=1}^{N_v}\left(v(x) - L_x^\alpha u(x)^*\right)^2 \tag{21}$$

$$MSE_0 = \frac{1}{N_0}\sum_{i=1}^{N_0}\left(u(x_0) - u_0^*\right)^2 \tag{22}$$

$$MSE_b = \frac{1}{N_b}\sum_{i=1}^{N_b}\left(u(x_b) - u_b^*\right)^2 \tag{23}$$

where $u(x_0)$, $u(x_b)$ and u_0^*, u_b^* are the initial and boundary values of the function and network, respectively.

3. Results

3.1. Seismic wave equation

The 1D time-dependent seismic wave equation in an isotropic and homogeneous medium can be expressed by (Sheriff and Geldart, 1995):

$$\frac{\partial^2 u(x,t)}{\partial t^2} - V^2\frac{\partial^2 u(x,t)}{\partial x^2} = 0 \tag{24}$$

where $u(x,t)$ is the displacement in point x at time t. The general solution to this differential equation can be expressed as $u(x,t) = A\cos[2\pi(t - x/V) - \varphi]$, where A and φ are the amplitude and phase of the wave. In this study, we assume a wave with unite amplitude, zero phase, and a velocity of 2 km/s, which leads to:

$$u(x,t) = \cos[2\pi(t - x/2)],\ 0 \le x \le 1,\ 0 \le t \le 1 \tag{25}$$

where the units of x and t are km and s. We aim to use physics informed machine learning algorithms (i.e., GP and PIML) to train a machine by limited numbeq2rs of training points from $u(x,t)$ for both inversion of the velocity V and prediction of $u(x,t)$.

3.2. GP results

According to Section 2.1 as well as seismic wave equation shown in Eq. (24), $\alpha = V^2$ and $v = 0$, and the corresponding covariance function is:

$$k_{uu}(x, x', t, t'; \lambda_1, \lambda_2) = \sigma_u^2 e^{-\frac{1}{2}\left(\lambda_1(x-x')^2 + \lambda_2(t-t')^2\right)} \tag{26}$$

Other covariance functions (k_{uv}, k_{vu}, k_{vv}) are computed based on Eqs. (4)–(6). Fifty numbers of training data ($n_u = n_v = 50$) are randomly selected from both x and t domain ($[0,1]^2$), and $y_u = u(x,t)$ are computed from Eq. (25). According to Eq. (24), $y_v = 0$ for all data. We used a L-BFGS-B (Zhu et al., 1997) as the optimization method to minimize the negative log marginal likelihood (i.e., Eq. (8)). Results showed that in noise-free data the inverted velocity is 2.0382 km/s. Fig. 1a illustrates the predicted surface of the wave as well as the spatial location of training data. The results are also compared with the original data. Fig. 1b shows absolute differences between the original and predicted values ($|u(x,t) - u^*(x,t)|$). Also, variances of predictions are shown in Fig. 1c.

To add more complexity, we produced noisy data and generated $y_u = u(x,t) + \varepsilon_u$ and $y_v = v(x,t) + \varepsilon_v$, where ε_u and ε_v are the noise values. Both noises are assumed to have a normal distribution with zero mean and unite variance ($\varepsilon \sim \mathscr{N}(0,1)$). Results show that the inverted velocity is 1.9109 km/s, which is less accurate than the noise-free data. Fig. 1d–f illustrate the predicted surface, errors, and variances of predictions.

3.3. PIML results

The utilized network in this study is illustrated in Fig. 2. As can be seen, the inputs are x and t based on our 1D assumption of the wave equation, i.e. Eq. (25), and the only output is $u(x,t)$, which is the solution of the differential equation, i.e. Eq. (24). The hidden neurons consist of 2 layers with 40 neurons in each. We will come back to this point shortly. Finally, our PIML network is constructed and the following loss function is implemented based on which the first and second derivatives of the network output with regard to x and t (Eqs. (19)–(23)) are computed:

$$Loss\ \ function = \frac{1}{N_u}\sum_{i=1}^{N_u}\left(u(x,t) - u^*(x,t)\right)^2 + \frac{1}{N_v}\sum_{i=1}^{N_v}\left(\frac{\partial^2 u^*(x,t)}{\partial t^2} - V^2\frac{\partial^2 u^*(x,t)}{\partial x^2}\right)^2 \tag{27}$$

where $u^*(x,t)$ is the output of the network.

The velocity value is introduced as an unknown parameter to the

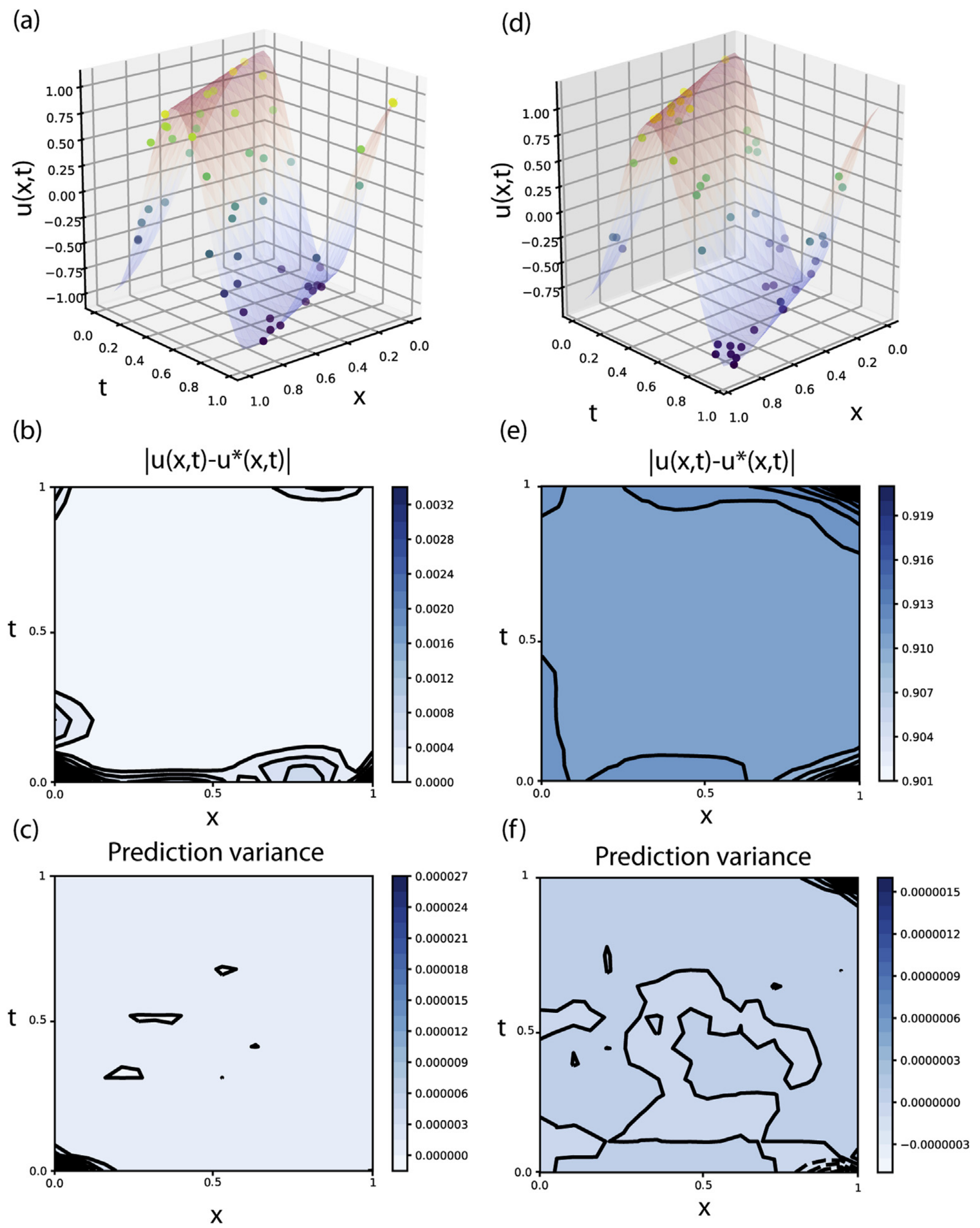

Fig. 1. The GP evaluation: (a, d) Predicted $u(x,t)$ and (b, e) the absolute error of predictions and (c, f) variance of predictions for noise-free and noisy data, respectively.

network, which allows the network to optimize it during the learning procedure. This means that the velocity inversion and displacement prediction are conducted simultaneously.

To find the optimum numbers of hidden layers and neurons, we generated 1000 numbers of training data in a regular pattern in x, t domain. Then, for 2, 4 and 6 hidden layers each with 20, 30 and 40 neurons, we trained the network and the obtained MSE values as well as inverted velocity are compared. These results are reported in Table 1, which indicates that the best performance can be obtained when the number of layers and neurons are selected 4 and 20, respectively.

After finding the optimal architecture of the PINN, we randomly generated 1000 noise-free samples from $u(x,t)$ and trained the network using *adam* optimization function. We also used *tanh* as the activation function, 100 number of batch size, 0.001 for learning rate, 500 patience epochs and 7000 epochs. Fig. 3a shows the MSEs during the training phase. As can be seen, after 5000 epochs, the loss function reaches a plateau. Fig. 4a also illustrates the predicted wave as well as the training data. To evaluate the performance of the conducted predictions, Fig. 4b shows the absolute difference between the original and predicted values. The inverted velocity for noise-free data is 1.9994 km/s. We also generated noisy data by adding noises to the training data ($y_u = u(x,t) + \varepsilon_u$). The noises ε_u were sampled from a normal distribution with zero mean and unite variance ($\varepsilon \sim \mathcal{N}(0,1)$). Figs. 3b, 4c and 4d show the MSE components, predicted wave and absolute difference between the original and predicted values. The inverted velocity for noisy data is 1.9991 km/s.

4. Discussion

Deep learning methods are accounted as data driven methods with many cons and pros, but being a *black box* is one the most negative points in scientific computations. In physics informed deep learning methods, however, physical laws and empirical relations are involved in the learning process, which makes these methods to be more scientifically

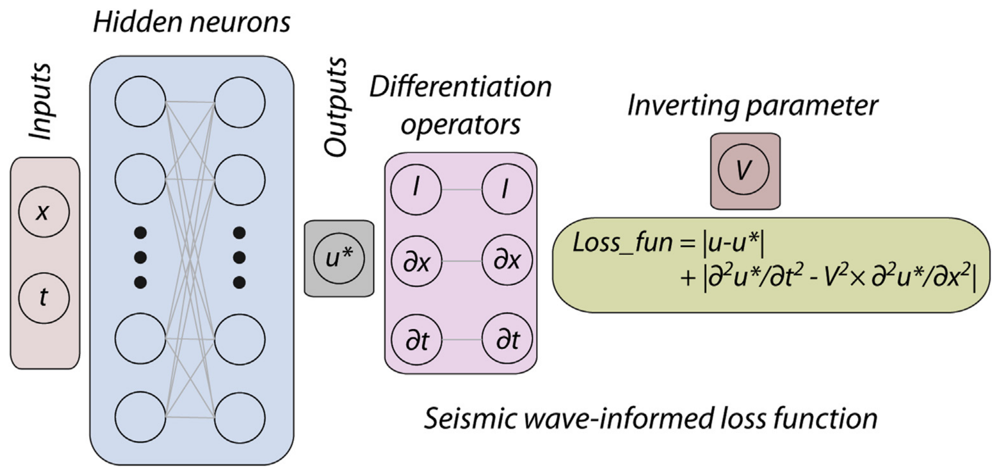
Hidden neurons
Inputs
x
t
Outputs
u*
Differentiation operators
I
∂x
∂t
Inverting parameter
V
Loss_fun = |u-u*| + |∂²u*/∂t² - V²× ∂²u*/∂x²|
Seismic wave-informed loss function

Fig. 2. Architecture of the proposed PIML for solving 1D seismic wave equation.

Table 1
MSE and inverted velocity values for networks with different hidden layers and neurons (bold numbers are selected as optimal values).

		Number of Neurons		
Layer		20	30	40
2	MSE	2.84×10^{-4}	4.33×10^{-4}	4.31×10^{-4}
	V	1.998	1.9968	1.9972
4	MSE	**2.44×10^{-5}**	8.19×10^{-5}	5.19×10^{-5}
	V	**1.9937**	1.9771	1.9884
6	MSE	1.67×10^{-4}	4.81×10^{-5}	6.04×10^{-4}
	V	1.9646	1.9768	1.9727

oriented. Some of these methods such as GP are even tractable analytically. Using the results of our study, we demonstrated such methods by evaluating their abilities and flexibilities to be used for solving seismic wave equation as an example in geosciences.

Prediction of a field, function or parameter is one the main abilities of deep learning methods. For example, here we aimed to predict displacements of points in x direction along time t; i.e. ($u(x,t)$); where a wave is propagated through a 1D medium with 2 km/s velocity. Unlike many of deep leaning methods, the GP can predict $u(x,t)$ using a small dataset (e.g., here $n_u = n_v = 50$) with a promising level of error either in noise-free or noisy data as shown in Fig. 1. Adding noise results a higher level of error in prediction which is expected. One can compare Fig. 1b and e where the prediction error increases by including a noise to training data. A critical issue with the GP method is minimizing log marginal likelihood - Eq. (9) - where inverting the covariance matrix and an optimization process are needed. When the number of training data is increased, a direct inversion of **K** is not computationally feasible. Instead, it is suggested to use Cholesky decomposition of **K**, since it is faster and numerically more stable (Rasmussen and Williams, 2006). It is also highly probable that **K** becomes ill-positioned during optimization when the parameters are changed. The optimization may converge to a local minimum which is also another issue. In fact, these issues correspond to the optimization problem, which is still assumed as an open problem (Raissi et al., 2017b). To have the best results, the problem is solved using a variant of hyper-parameter initializations and the solution is the smallest log marginal likelihood.

On the other hand, the optimization process in PINN is more stable than GP, but PINN cannot predict $u(x,t)$ with very small training data like GP. For example, even with using 1000 samples, the prediction error of PINN is higher than GP (compare Figs. 1b and 4b), though both of them are very small. To provide a more quantitative comparison, histograms of such errors are plotted in Fig. 5. Givens the data presented here and those in Figs. 1b and 4b, high error points are distributed in boundaries of prediction surface. Based on mean values of errors, the error levels of both methods are comparable, while the error by GP is smaller than PINN more than two times. Adding noise to the training data hence increases the prediction error in both methods, which is due to random nature of noises (see Figs. 1e and 4d).

An inherent property of the GP method, as a Bayesian procedure, is computing the variances of predictions which is known as *uncertainty quantification*. Fig. 1c and f illustrate the variances of prediction for noise-free or noisy data, which are indicators of how certain these predictions are. In practice, the results of uncertainty quantification can be used for decision making process in a real project.

The other interesting point of these methods is their ability to invert the parameters of the governing equations while training, which is known as *inversion* or *inverse problem*. Table 2 summarizes the inverted velocities by two methods. According to these results, the PINN is more reliable than the GP, where the difference between the true and inverted velocities are 0.03% and 0.04% for noise-free and noisy data, respectively. These errors are 1.91% and 4.49% in the case of the GP method.

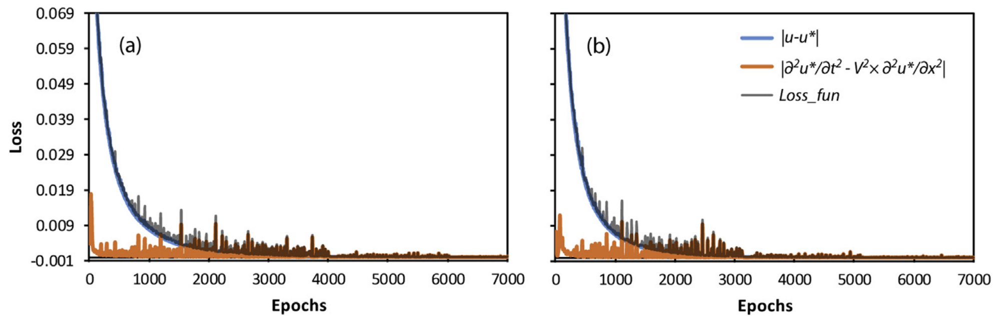

Fig. 3. MSE values of different components of the loss function during training for (a) noise free and (b) noisy data.

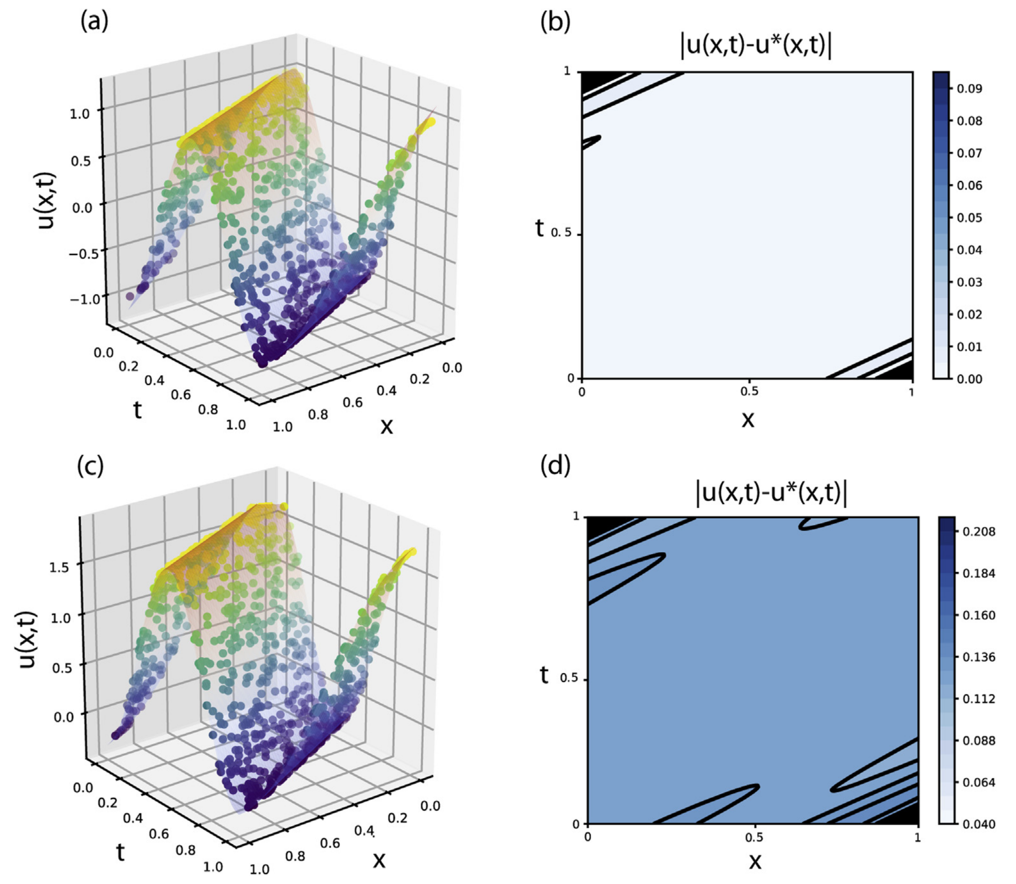

Fig. 4. The PINN evaluation: (a) Predicted $u(x,t)$ and (b) the absolute error of predictions for noise free data. (c) Predicted $u(x,t)$ and (d) the absolute error of predictions for noisy data.

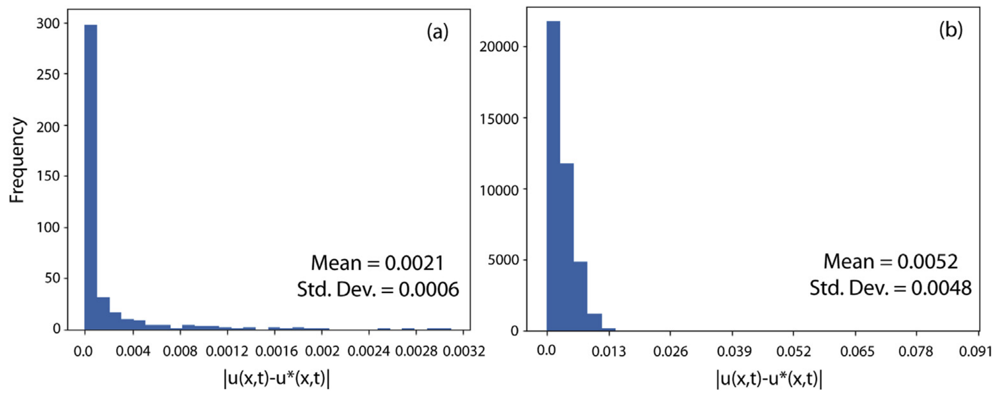

Fig. 5. Histograms of prediction errors generated by (a) GP, and (b) PINN methods.

The PINN methods is also more flexible than GP with regard to involving more physical relations. For example, in this case, one could benefit from linear elastic relations for both P- and S-waves to invert other parameters such as Lame parameters and/or density. Since we use 1D equations, it is not possible to invert Lame parameters, but both P- and S-waves as well as density could still be inverted. To this end, elastic relations could be written as:

$$\varepsilon_{xx}(x,t) = \frac{\partial u_x(x,t)}{\partial x} \tag{28}$$

$$\sigma_{xx}(x,t) = V_P^2 \rho \varepsilon_{xx}(x,t) \tag{29}$$

$$\varepsilon_{xy}(x,t) = \frac{\partial u_y(x,t)}{\partial x} \tag{30}$$

$$\sigma_{xy}(x,t) = V_S^2 \rho \varepsilon_{xy}(x,t) \tag{31}$$

Table 2
Inverted velocities (the true velocity is 2 km/s).

	GP		PINN	
	Noise free	Noisy	Noise free	Noisy
Inverted Velocity	2.0382	1.9109	1.9994	1.9991
Difference (%)	1.91	4.49	0.03	0.04

where $\varepsilon_{xx}(x,t)$ and $\sigma_{xx}(x,t)$ are strain and stress fields in x plane corresponding to displacement in x direction $u_x(x,t)$. Similarly, $\varepsilon_{xy}(x,t)$ and $\sigma_{xy}(x,t)$ are strain and stress fields in x plane corresponding to displacement in y direction; i.e. $u_y(x,t)$. V_P, V_S and ρ are P- and S-waves velocities and the density of the medium, respectively.

To involve these relations into the learning process, one needs to compute the strain by computing differential of estimated $u(x,t)$ (in both x and y directions) using Eqs. (28) and (30). Since velocities and density are unknowns, stress must be estimated by the neural network and, then, tied to strain by Eqs. (29) and (31). The wave equation is similar to Eq. (24), but for each of P- and S-wave. This leads to simultaneous inversion of P- and S-wave velocity and as well as density as shown in Fig. 6.

To estimate the stress values, either one neural network with two outputs or two neural networks each with one output might be used. Previous studies (Haghighat et al., 2020) showed that one output per network is more effective. Here, we used four individual networks for each of $u_x(x,t)$, $\sigma_{xx}(x,t)$, $u_y(x,t)$ and $\sigma_{xy}(x,t)$ as illustrated in Fig. 6. Thus, the loss function is defined as:

$$\begin{aligned}Loss\,function=&\frac{1}{N_u}\sum_{i=1}^{N_u}\left(u_x(x,t)-u_x^*(x,t)\right)^2+\frac{1}{N_u}\sum_{i=1}^{N_u}\left(\sigma_{xx}(x,t)-\sigma_{xx}^*(x,t)\right)^2\\&+\frac{1}{N_u}\sum_{i=1}^{N_u}\left(u_y(x,t)-u_y^*(x,t)\right)^2+\frac{1}{N_u}\sum_{i=1}^{N_u}\left(\sigma_{xy}\left(x,t\right)-\sigma_{xy}^*\left(x,t\right)\right)^2\\&+\frac{1}{N_u}\sum_{i=1}^{N_u}\left(\sigma_{xx}^*(x,t)-V_P^2\rho\frac{\partial u_x^*(x,t)}{\partial x}\right)^2+\frac{1}{N_u}\sum_{i=1}^{N_u}\left(\sigma_{xy}^*(x,t)-V_S^2\rho\frac{\partial u_y^*\left(x,t\right)}{\partial x}\right)^2\\&+\frac{1}{N_v}\sum_{i=1}^{N_v}\left(\frac{\partial^2u_x^*(x,t)}{\partial t^2}-V_P^2\frac{\partial^2u_x^*(x,t)}{\partial x^2}\right)^2+\frac{1}{N_v}\sum_{i=1}^{N_v}\left(\frac{\partial^2u_y^*\left(x,t\right)}{\partial t^2}-V_S^2\frac{\partial^2u_y^*(x,t)}{\partial x^2}\right)^2\end{aligned}\tag{32}$$

We supposed the medium with P- and S-wave velocity of 2 and 1.2 km/s and a density of 2.2 g/cm^3 and used the same wave relation as in Eq. (25) for both P- and S-wave. Therefore, the stress field is defined as $2\pi(V\rho)\sin[2\pi(t-x/V)]$. Using similar arguments as the previous PINN, this network was also trained and tested. Fig. 7 shows evolution of the loss function and its components during the training, which indicates that the training process has been stopped at 5348-th epoch using early

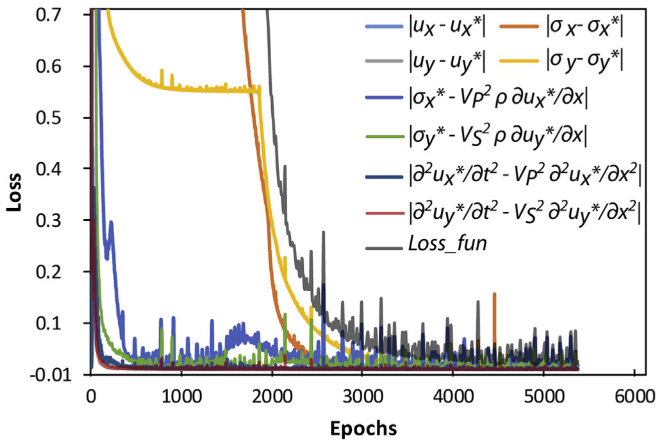

Fig. 7. MSE values of different components of the loss function during training.

stopping procedure. Fig. 8 shows the results of the PINN for prediction of $u_x(x,t)$, $\sigma_{xx}(x,t)$, $u_y(x,t)$ and $\sigma_{xy}(x,t)$ and their corresponding errors.

According to this PINN, P- and S-wave velocities and density values have been inverted as 1.9998 km/s, 1.2001 km/s and 2.1983 g/cm^3 which are very close to their original values (2 km/s, 1.2 km/s and 2.2 g/cm^3). It can also be concluded from these results that not only PINN optimization process is more stable and the inverted values are more accurate, but it is also more flexible then GP for involving more physical relation into learning process.

The problem solved in this study is a 2D problem: 1D wave equation in space and time (x,t). To solve more complex problems in 3D, the networks and loss functions must be extended to, for example, more inputs and outputs. In fact, there is no limitations for extending the methods beyond 2D problems except that the computational time may increase for higher-dimensional problems. In this study, a system with Intel Core i7 CPU, 32 GB RAM and GTX1050Ti GPU was used. The computation time for GP depends on the number of input data and the utilized optimization algorithm. In our study, however, it took few seconds to complete the process. The run time for training of our large PINN (Fig. 6) was less than 19 min.

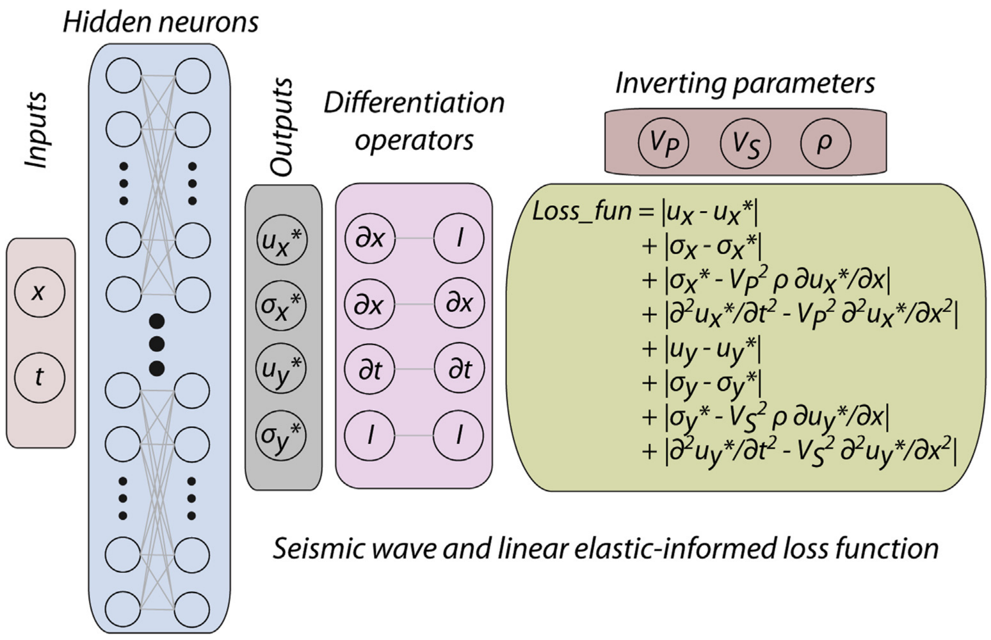

Fig. 6. Architecture of the PINN for solving 1D seismic wave equation involving with linear elastic equations for inversion of P- and S-wave velocities and density.

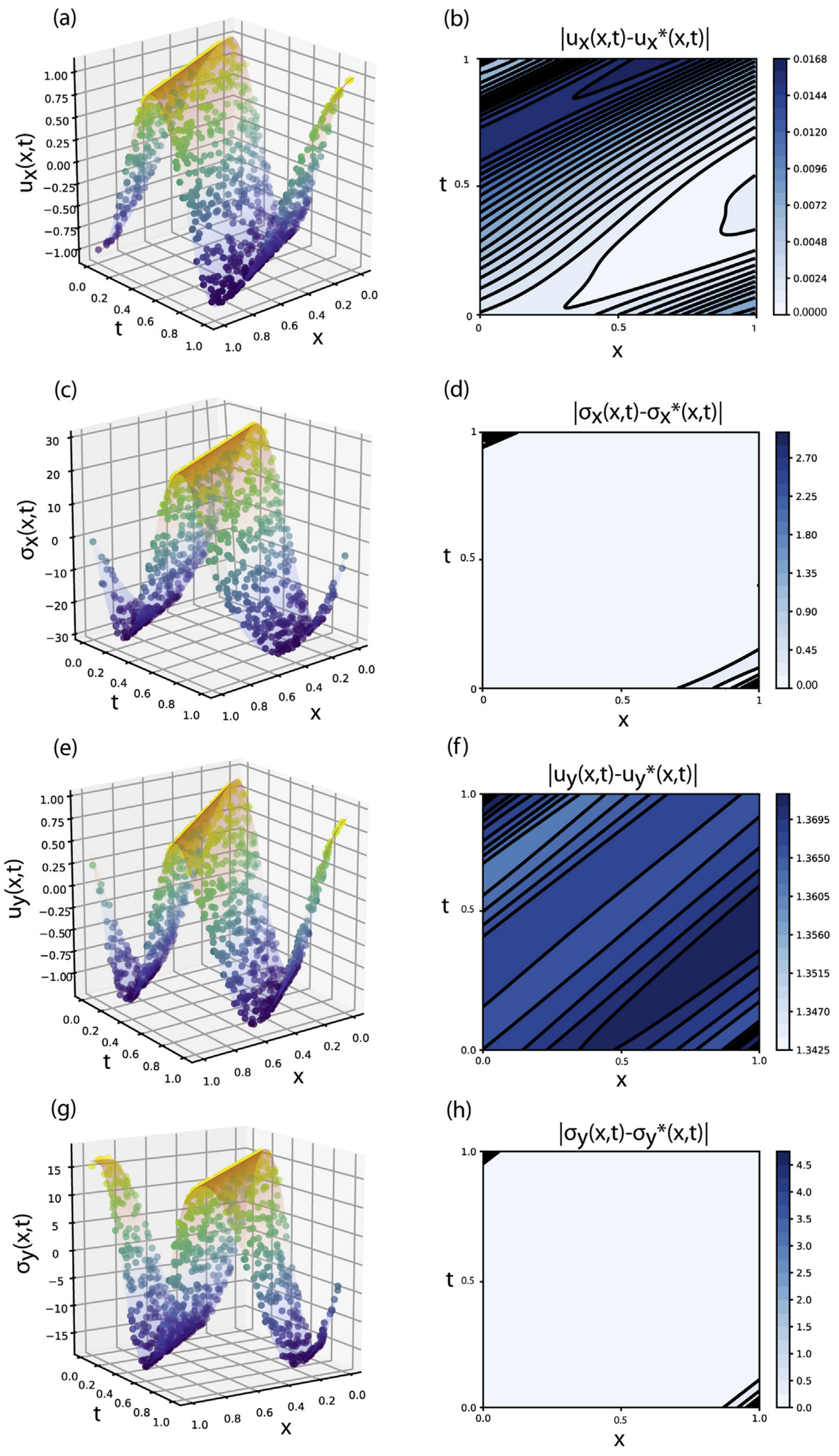

Fig. 8. Predicted values and the absolute error of predictions for (a, b) $u(x,t)$, (c, d) $\sigma_x(x,t)$, (e, f) $u_y(x,t)$ and (g, h) $\sigma_y(x,t)$.

5. Conclusions

Physics informed machine learning methods belong to those data driven methods which are trained not only by the observed data, but also by the induced physics laws. In this study, we implemented the Gaussian process and physics informed machine learning to solve differential equation of a 1D seismic wave as one of the most popular equations in geosciences. In overall, the following points are concluded by this work:

(1) Both GP and PINN can solve the differential equation and predict the $u(x,t)$ with a small error. However, the error level of the GP is smaller than PINN.

(2) The prediction error is increased by adding noise into the training data, but both methods are robust to the random noise.
(3) In GP, minimizing the log marginal likelihood is likely to result in an ill-positioned covariance matrix or local minima.
(4) Due to probabilistic nature of GP, variance of predictions is also computed which can be used for uncertainty quantification.
(5) Both GP and PINN can invert the velocity of wave as an unknown parameter of the equation accurately. However, PINN is more accurate than the GP.
(6) The PINN is flexible enough to add more physical relations and, therefore, can invert more unknown parameters.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Alvarez, M.A., Luengo, D., Lawrence, N.D., 2013. Linear latent force models using Gaussian processes. IEEE Trans. Pattern Anal. Mach. Intell. 35 (11), 2693–2705. https://doi.org/10.1109/TPAMI.2013.86.

Aslam, M., Khan, T.M., Naqvi, S.S., Holmes, G., Naffa, R., 2019. On the application of automated machine vision for leather defect inspection and grading: a survey. IEEE Access 7, 176065–176086. https://doi.org/10.1109/ACCESS.2019.2957427.

Bai, T., Tahmasebi, P., 2020. Hybrid geological modeling: combining machine learning and multiple-point statistics. Comput. Geosci. 104519. https://doi.org/10.1016/j.cageo.2020.104519.

Bernal, J., Kushibar, K., Asfaw, D.S., Valverde, S., Oliver, A., Martí, R., Lladó, X., 2019. Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. Artif. Intell. Med. 95, 64–81. https://doi.org/10.1016/j.artmed.2018.08.008.

Cockayne, J., Oates, C., Sullivan, T., Girolami, M., 2017. Probabilistic numerical methods for PDE-constrained Bayesian inverse problems. AIP Conf. Proc. 1853, 060001. https://doi.org/10.1063/1.4985359.

Dondelinger, F., Husmeier, D., Rogers, S., Filippone, M., 2013. ODE parameter inference using adaptive gradient matching with Gaussian processes. In: Sixteenth International Conference on Artificial Intelligence and Statistics. Scottsdale, AZ, USA, pp. 216–228.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J., 2017. A Review on Deep Learning Techniques Applied to Semantic Segmentation. https://doi.org/10.1007/978-1-4471-4640-7. arXiv:1704.06857.

Guo, Z., Li, X., Huang, H., Guo, N., Li, Q., 2019. Deep learning-based image segmentation on multimodal medical imaging. IEEE Trans. Radiat. Plasma Med. Sci. 3, 162–169. https://doi.org/10.1109/TRPMS.2018.2890359.

Haghighat, E., Raissi, M., Moure, A., Gomez, H., Juanes, R., 2020. A Deep Learning Framework for Solution and Discovery in Solid Mechanics: Linear Elasticity. arXiv: 2003.02751.

Kamrava, S., Sahimi, M., Tahmasebi, P., 2020a. Quantifying accuracy of stochastic methods of reconstructing complex materials by deep learning. Phys. Rev. E 101, 043301. https://doi.org/10.1103/PhysRevE.101.043301.

Kamrava, S., Tahmasebi, P., Sahimi, M., 2019a. Linking morphology of porous media to their macroscopic permeability by deep learning. Transport Porous Media. https://doi.org/10.1007/s11242-019-01352-5.

Kamrava, S., Tahmasebi, P., Sahimi, M., 2019b. Enhancing images of shale formations by a hybrid stochastic and deep learning algorithm. Neural Netw. 118, 310–320. https://doi.org/10.1016/J.NEUNET.2019.07.009.

Kamrava, S., Tahmasebi, P., Sahimi, M., Arbabi, S., 2020b. Phase transitions, percolation, fracture of materials, and deep learning. Phys. Rev. E 102 (1), 011001. https://doi.org/10.1103/PhysRevE.102.011001.

Karimpouli, S., Tahmasebi, P., 2019a. Segmentation of digital rock images using deep convolutional autoencoder networks. Comput. Geosci. 126, 142–150. https://doi.org/10.1016/J.CAGEO.2019.02.003.

Karimpouli, S., Tahmasebi, P., 2019b. Image-based velocity estimation of rock using Convolutional Neural Networks. Neural Netw. 111, 89–97. https://doi.org/10.1016/J.NEUNET.2018.12.006.

Karimpouli, S., Tahmasebi, P., Saenger, E.H., 2019. Coal cleat/fracture segmentation using convolutional neural networks. Nat. Resour. Res. 29, 1675–1685. https://doi.org/10.1007/s11053-019-09536-y.

Kingma, D.P., Ba, J., 2014. Adam: A Method for Stochastic Optimization. arXiv: 1412.6980.

Kong, Q., Trugman, D.T., Ross, Z.E., Bianco, M.J., Meade, B.J., Gerstoft, P., 2019. Machine learning in seismology: turning data into insights. Seismol Res. Lett. 90, 3–14. https://doi.org/10.1785/0220180259.

Lagaris, I.E., Likas, A., Fotiadis, D.I., 1998. Artificial neural networks for solving ordinary and partial differential equations. IEEE Trans. Neural Network. 9, 987–1000. https://doi.org/10.1109/72.712178.

Lawrence, N., 2005. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. J. Mach. Learn. Res. 6, 1783–1816.

Lee, M., Lee, J., Chang, J.-H., 2019. Ensemble of jointly trained deep neural network-based acoustic models for reverberant speech recognition. Digit. Signal Process. 85, 1–9. https://doi.org/10.1016/j.dsp.2018.11.005.

Maitra, S., Ghosh, R., Ghosh, K., 2019. Applications of deep learning in medical imaging. In: Balas, V.E., Roy, S.S., Sharma, D., Samui, P. (Eds.), Handbook of Deep Learning Applications. Springer International Publishing, Cham, pp. 111–127. https://doi.org/10.1007/978-3-030-11479-4_6.

Meade, A.J., Fernandez, A.A., 1994. The numerical solution of linear ordinary differential equations by feedforward neural networks. Math. Comput. Model. 19, 1–25. https://doi.org/10.1016/0895-7177(94)90095-7.

Nasirahmadi, A., Sturm, B., Edwards, S., Jeppsson, K.-H., Olsson, A.-C., Müller, S., Hensel, O., 2019. Deep learning and machine vision approaches for posture detection of individual pigs. Sensors 19 (17), 3738. https://doi.org/10.3390/s19173738.

Neal, R.M., 2012. Bayesian Learning for Neural Networks, vol. 118. Springer Science & Business Media.

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2019. 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. https://doi.org/10.1016/j.jcp.2018.10.045.

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2017a. Inferring solutions of differential equations using noisy multi-fidelity data. J. Comput. Phys. 335, 736–746. https://doi.org/10.1016/j.jcp.2017.01.060.

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2017b. Machine learning of linear differential equations using Gaussian processes. J. Comput. Phys. 348, 683–693. https://doi.org/10.1016/j.jcp.2017.07.050.

Raissi, M., Yazdani, A., Karniadakis, G.E., 2020. Hidden fluid mechanics: learning velocity and pressure fields from flow visualizations. Science 367 (6481), 1026–1030. https://doi.org/10.1126/science.aaw4741.

Rasmussen, C.E., Williams, C.K., 2006. Gaussian Processes for Machine Learning. MIT Press, Massachusetts.

Rouet-Leduc, B., Hulbert, C., McBrearty, I.W., Johnson, P.A., 2020. Probing slow earthquakes with deep learning. Geophys. Res. Lett. 47, e2019GL085870. https://doi.org/10.1029/2019GL085870.

Sheriff, R.E., Geldart, L.P., 1995. Exploration Seismology. Cambridge University Press, UK.

Tahmasebi, P., Kamrava, S., Bai, T., Sahimi, M., 2020. Machine learning in geo- and environmental sciences: from small to large scale. Adv. Water Resour. 142, 103619. https://doi.org/10.1016/j.advwatres.2020.103619.

Tang, V., Seetharaman, P., Chao, K., Pardo, B.A., van der Lee, S., 2020. Automating the detection of dynamically triggered earthquakes via a deep metric learning algorithm. Seismol Res. Lett. 91, 901–912. https://doi.org/10.1785/0220190165.

Titsias, M., Lawrence, N.D., 2010. Bayesian Gaussian process latent variable model. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS) 2010. Chia Laguna Resort, Sardinia, Italy, pp. 844–851.

Vapnik, V., 2013. The Nature of Statistical Learning Theory. Springer Science & Business Media, p. 768.

Vishal, P., Aggarwal, K.R., 2019. A hybrid of deep CNN and bidirectional LSTM for automatic speech recognition. J. Intell. Syst. 29 (1), 1261–1274. https://doi.org/10.1515/jisys-2018-0372.

Waldeland, A.U., Jensen, A.C., Gelius, L.-J., Solberg, A.H.S., 2018. Convolutional neural networks for automated seismic interpretation. Lead. Edge 37, 529–537. https://doi.org/10.1190/tle37070529.1.

Wang, J., Tchapmi, L.P., Ravikumar, A.P., McGuire, M., Bell, C.S., Zimmerle, D., Savarese, S., Brandt, A.R., 2020. Machine vision for natural gas methane emissions detection using an infrared camera. Appl. Energy 257, 113998. https://doi.org/10.1016/j.apenergy.2019.113998.

Xiong, W., Ji, X., Ma, Y., Wang, Y., AlBinHassan, N.M., Ali, M.N., Luo, Y., 2018. Seismic fault detection with convolutional neural network. Geophysics 83 (5), O97–O103. https://doi.org/10.1190/geo2017-0666.1.

Zhu, C., Byrd, R.H., Lu, P., Nocedal, J., 1997. Algorithm 778: l-BFGS-B: fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math Software 23 (4), 550–560. https://doi.org/10.1145/279232.279236.

Zhu, Y., Zabaras, N., Koutsourelakis, P.-S., Perdikaris, P., 2019. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. J. Comput. Phys. 394, 56–81. https://doi.org/10.1016/j.jcp.2019.05.024.