Computational Geosciences (2023) 27:737–751
https://doi.org/10.1007/s10596-023-10232-3

ORIGINAL PAPER

Hard enforcement of physics-informed neural network solutions of acoustic wave propagation

Harpreet Sethi[1] · Doris Pan[2] · Pavel Dimitrov[2] · Jeffrey Shragge[1] · Gunter Roth[2] · Ken Hester[2]

Received: 25 July 2022 / Accepted: 9 June 2023 / Published online: 23 July 2023

Abstract

Simulating the temporal evolution of wavefield solutions through models with heterogeneous material properties is of practical interest for many scientific applications. The acoustic wave equation (AWE) is often used for studying wave propagation in both fluids and solids and is crucial for many applications including seismic imaging and inversion and non-destructive testing. Because analytical AWE solutions rarely exist for complex heterogeneous media, methods for generating numerical AWE solutions are very desirable. Traditional numerical solvers require discrete model representations with many restrictions placed on the shape and spacing of grid elements. This work uses a relatively new class of numerical solvers known as physics-informed neural networks (PINNs) that provide a mesh-free alternative for generating AWE solutions using a deep neural-network framework. We encapsulate a time-domain AWE formulation within a loss function that is used to train network parameters. The initial conditions are implemented by enforcing hard constraints on the neural network instead of including them as separate loss-function terms. We also use a Fourier neural network (FNN) to alleviate the spectral bias commonly observed when using fully connected neural network in the conventional PINN approach. Numerical tests on both 2D homogeneous and heterogeneous velocity models confirm the accuracy of our approach. We observe that using FNNs helps in the convergence of AWE solutions especially for heterogeneous models. We compare PINN-based solutions with those computed by the highly accurate conventional pseudo-spectral method, and observe that the normalized energy differences between the two sets of solutions were less than 4% for all numerical tests.

Keywords Acoustic · Wave propagation · Neural networks · Physics informed

Mathematics Subject Classification (2010) 86–08 · 86–10

1 Introduction

Numerical solutions of wave equations are of scientific importance for wide range of physical problems including in the fields of seismology, acoustics, fluid dynamics, medical science, and electromagnetics. Various numerical approaches including finite-difference, finite-element, finite-volume, and spectral methods have been developed to compute wavefield solutions; however, existing numerical methods require explicitly solving the governing wave equation in an step-wise fashion and can suffer from numerical errors caused by discrete meshing. A further challenge is the significant computational cost associated with obtaining multi-scale solutions that require discretizing the computational grid very finely, which results in large memory and/or hardware storage requirements.

Deep neural networks (DNNs) recently have gained significant attention for their ability to learn complex dynamical

✉ Harpreet Sethi
hsethi@mines.edu

Doris Pan
dorisp@nvidia.com

Pavel Dimitrov
pdimitrov@nvidia.com

Jeffrey Shragge
jshragge@mines.edu

Gunter Roth
gunterr@nvidia.com

Ken Hester
khester@nvidia.com

[1] Department of Geophysics, Colorado School of Mines, 1500 Illinois Street, Golden, CO 80401, USA

[2] NVIDIA Corporation, 2788 San Tomas Expressway, Santa Clara, CA 95050, USA

systems. Raissi et al. [1] propose a general framework, known as physics-informed neural networks (PINNs), to solve both forward and inverse problems involving systems of nonlinear partial differential equations (PDEs). The PINN framework involves approximating the PDE solution variables with a DNN. The physical laws governing the PDE as well as the initial and boundary conditions (ICs and BCs) are embedded as terms in the loss function used to regularize the network. Standard GPU-accelerated computational packages (e.g., TensorFlow, PyTorch, Keras) then can be used to find the set of solution variables that optimally minimizes the loss function and thereby satisfies the PDE, ICs, and BCs.

PINNs offer many practical advantages over traditional numerical methods. First, they represent a mesh-free numerical approach and are less prone to discretization errors. The partial derivative operators involved in the PDEs are computed using automatic differentiation [2], which is exact and free of truncation errors when compared to numerical differentiation methods using approximate stencils derived from Taylor-series or other expansion approaches. Once the network is trained, the complete spatial-temporal solution domain can be simulated in a single step, which is important for imaging and inversion applications that use the time history of propagating wavefields (e.g., seismic imaging for subsurface geological characterization or non-destructive testing).

The PINN framework has been used for solving numerous PDEs such as the Navier-Stokes equation for modeling fluid [3, 4] and cardiovascular [5] flow, the Black-Scholes equation for option pricing in computational finance [6], the Schrödinger equation for monitoring pulse evolutions in optical fiber [7], the non-linear equations of motion for structural seismic response modeling [8], the eikonal equation for computing activation times for a conduction velocity field [9], seismic traveltime equations [10], the Burgers equation for shallow-water modeling [1], and the Allen-Cahn equation for reaction-diffusion systems [1].

The application of PINNs for generating full-wavefield solutions of the acoustic wave equation (AWE) has been studied by different authors. Moseley et al. [11] used PINNs to investigate acoustic wave propagation; however, this approach used wavefield snapshots from a finite-difference method as well as physics-based constraints to train their network. Karimpouli and Tahmasebi [12] and Rasht-Behesht et al. [13] studied the use of PINNs for the forward and inverse seismic AWE applications and demonstrated that PINNs can learn wavefield solutions with fewer training data and that they are capable of estimating the acoustic velocity distribution. Alkhadhr et al. [14] solved the 1D acoustic wave equation for sinusoidal spatial source functions for medical ultrasound applications. Song et al. [15] formulated the problem in frequency domain to study PINNs for modeling seismic wavefields in transversely isotropic media with vertical symmetry axis (VTI) using an acoustic approximation. They computed scattered pressure wavefields based on a perturbation from an isotropic reference solution instead of directly simulating the full pressure wavefield itself. Song et al. [16] proposed to use adaptive sinusoidal activation functions to improve the multi-frequency PINN solutions.

A key challenge for using PINNs for generating AWE solutions is to train the network without using training data. In practice, assuming a known AWE solution at a certain time step arguably defeats the purpose of using PINNs for solving AWE. Because of the multi-scale and oscillatory nature of AWE solutions, it becomes challenging to train the network especially at higher frequencies. A further challenge is handling the injection of source energy localized in both space and time using PINNs, which is commonly used for modeling explosive and marine air-gun sources. For these scenarios a NN network will often accept a null solution if there are an insufficient number of sampling points in the localized region.

To overcome the aforementioned challenges, we propose to use a time-domain PINN formulation to solve the AWE for pressure wavefields. In contrast with conventional PINN approaches, we train a modified NN in which the initial conditions are satisfied explicitly using hard constraints in the same NN used to approximate the solution, which is important in terms of solution convergence. We test our approach using both homogeneous and heterogeneous acoustic velocity models. For the latter we advocate the use of Fourier neural networks (FNNs) [17] instead of the more commonly used fully connected NNs (FCNNs) to alleviate the spectral bias toward low-frequency components during the training process.

This paper starts with a review of the theory of wave propagation in acoustic media and then details our 2-D PINN approach including the inputs and outputs of FCNNs and the loss terms involved in the training procedure. Next, we briefly discuss FNNs and the differences between these and FCNNs. We then discuss the procedure of enforcing initial conditions in a hard manner using a single neural network and changes in network output. Finally, we present numerical examples including using FNNs that are able to better handle increasing complex velocity model heterogeneity.

2 Theory

The PINN framework uses physical laws that include the PDE and initial and boundary conditions (ICs and BCs) as terms in a loss function used to train the NN parameters. In this section, we first discuss the theoretical aspects of AWE including the ICs and BCs. We then present a standard PINN formulation using FCNNs, briefly introduce FNNs,

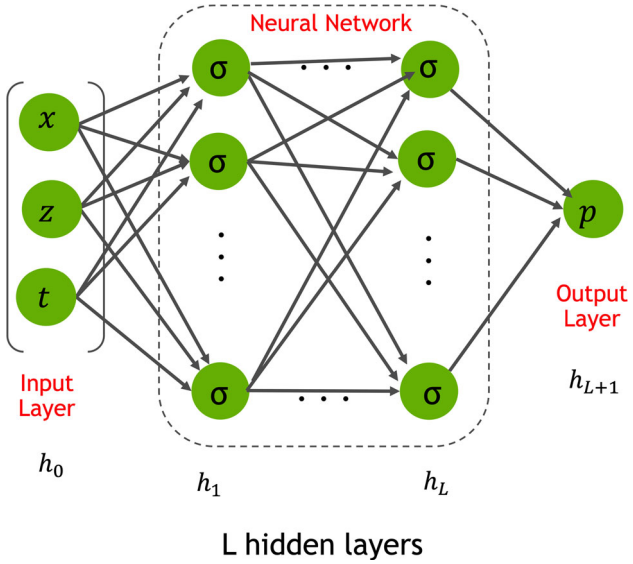

Fig. 1 Example of a basic deep neural network (DNN) consisting of an input layer of coordinates $\mathbf{x}(x, z, t)$, L hidden layers of parameters σ, and an output pressure layer representing the desired pressure field solution p

and describe the differences there between. Finally, we discuss the hard enforcement of ICs within the NN instead of using separate loss-function terms (i.e., soft enforcement) [1].

2.1 Acoustic wave equation

Two-dimensional (2-D) acoustic wave propagation in a fluid is described by a coupled system of first-order partial differential equations that represent the conservation of linear momentum,

$$\rho \partial_t v_i + \partial_i p = 0, \qquad i = 1, 2, \tag{1}$$

and the conservation of mass,

$$\partial_t p + \rho c^2 \partial_i v_i = f, \qquad i = 1, 2, \tag{2}$$

where ρ is the fluid density, v_i is the particle velocity, p is the pressure, index $i = 1, 2$ represents the two different coordinate axes, c is the medium velocity, and f is the body force per unit volume. Summation notation over repeated indicies is assumed. For a constant density medium, we can further substitute Eq. 1 into Eq. 2 to obtain a second-order PDE,

$$\partial_t \partial_t p - c^2 \partial_i \partial_i p = f, \quad i = 1, 2, \tag{3}$$

which is the form of the PDE solved in the present work.

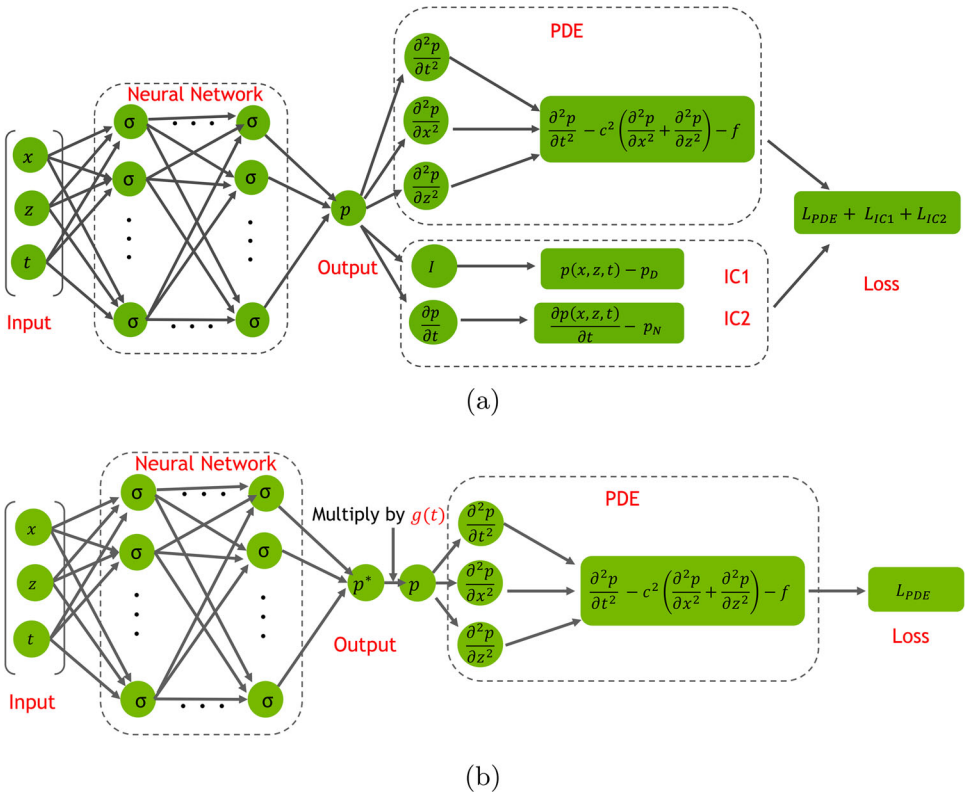

Fig. 2 (a) Schematic representation of PINNs framework. The neural network with l hidden layers on the left-hand side. (b) Modified PINN framework with hard-enforcement of initial conditions

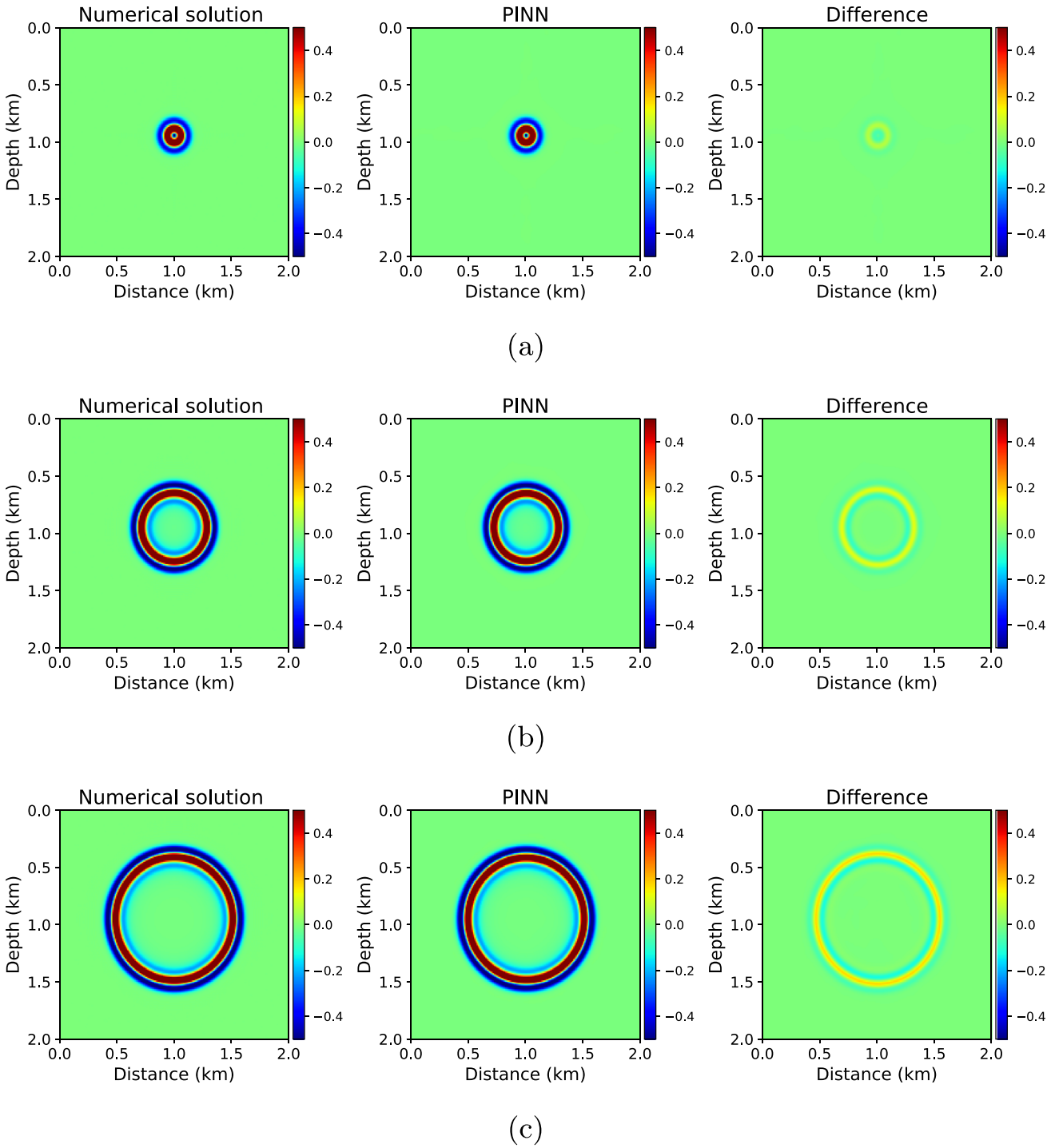

Fig. 3 Snapshots for homogeneous velocity model. The left and center columns present the pseudo-spectral and PINN solutions while the right column shows the direct differences. The top (a), middle (b) and lower (c) rows show the wavefield (differences) at times 0.15 s, 0.30 s, and 0.45 s, respectively

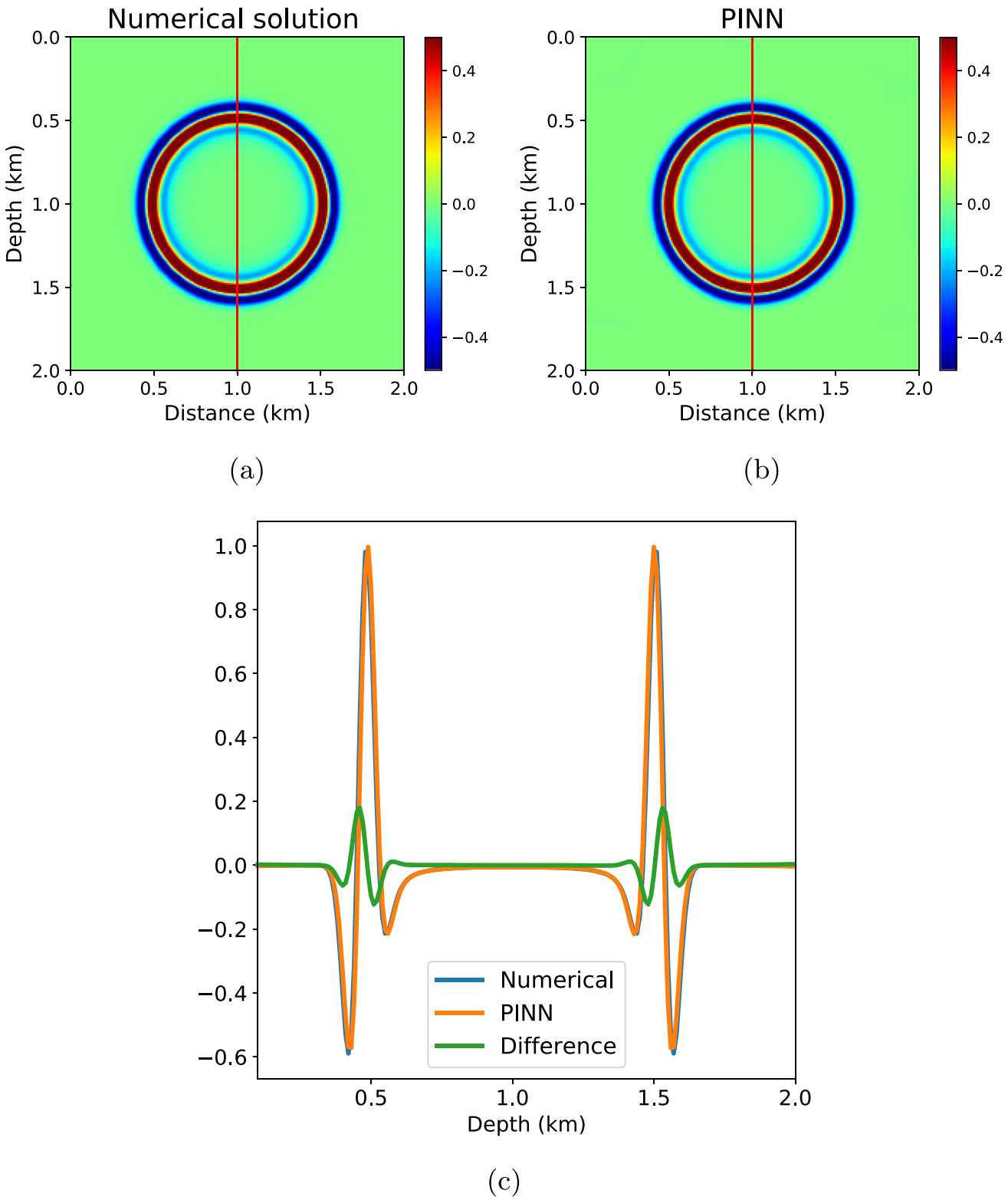

Fig. 4 Snapshots for homogeneous velocity model at t=0.45 s. (a) Pseudo-spectral wavefield solution. (b) PINN wavefield solution. (c) Wavefield difference between solutions in (a) and (b) computed along the vertical red line in (a) and (b)

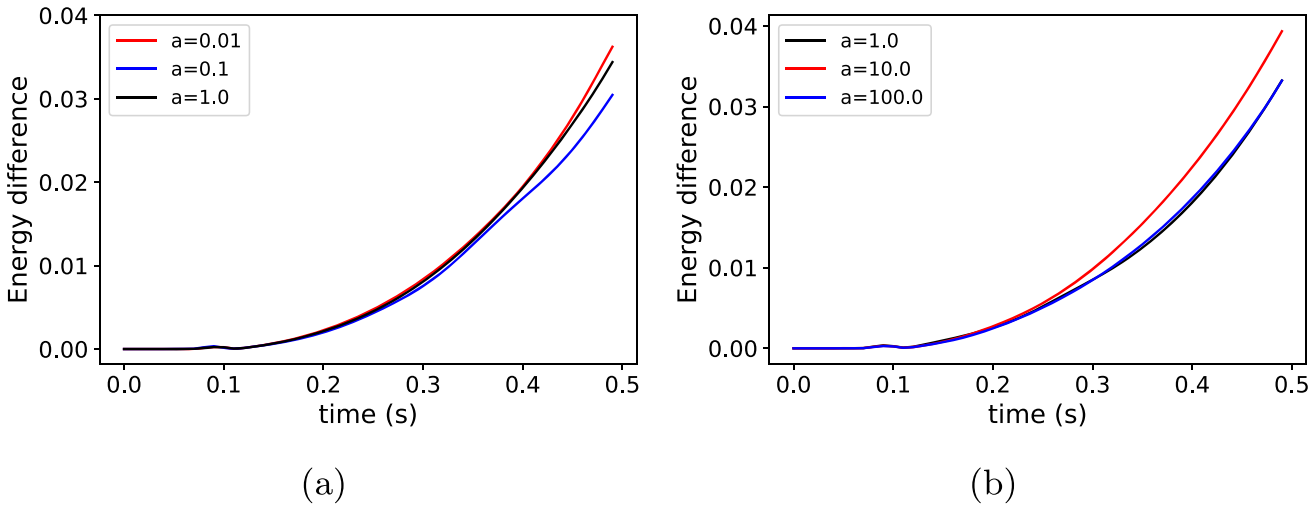

Fig. 5 Energy difference plots for functions with different scaling factors for homogeneous velocity model: (a) $g_1 = a_1 t$, and (b) $g_2 = 1 - \text{sech}\, a_2 t$

We assume a quiescent medium at $t = 0$ s, which defines the two ICs:

$$p(x, z, t = 0) = 0, \tag{4}$$

$$\partial_t p(x, z, t = 0) = 0. \tag{5}$$

We use absorbing boundary conditions (ABCs) [18] to handle reflections from the computational domain boundaries. In contrast to other numerical methods where one explicitly applies the ABC operators, the PINN approach does not require additional treatment (e.g., via an additional PDE loss term near the boundaries).

2.2 Physics-informed neural networks (PINNs)

PINN approaches commonly use a fully connected DNN to approximate the solution variable, which herein is the pressure wavefield p given in Eq. 3. A DNN is a stack of neurons organized in different layers (i.e., input, multiple hidden, and output; see Fig. 1).

The input layer includes the spatial (x, z) and temporal t coordinates as represented by the tensor $\mathbf{x}(x, z, t) \in R^N$, where N is the number of neurons. As an example, for a system with l hidden layers with i^{th} hidden layer having N_i neurons, the connection between $(i-1)^{th}$ to i^{th} hidden layer

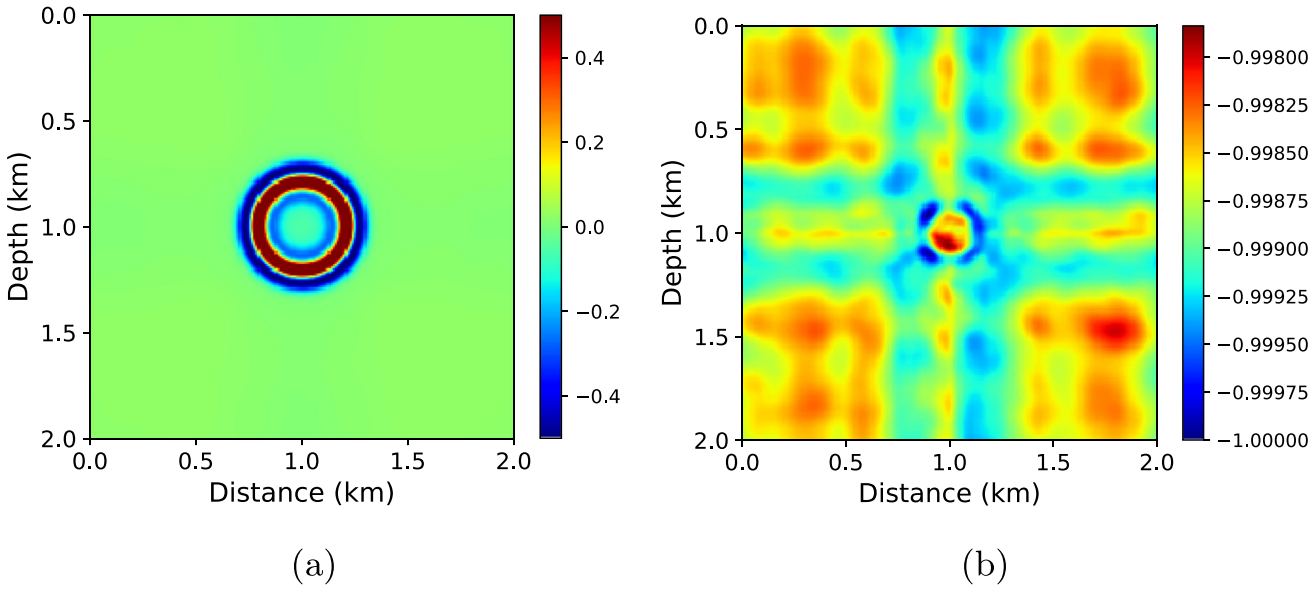

Fig. 6 Snapshots for homogeneous velocity model at t=0.2 s using (a) hard and (b) soft constraints

may be represented as

$$\mathbf{h}_i = \sigma\left(\mathbf{b}_i + \mathbf{W}_i \mathbf{h}_{i-1}\right) \quad 1 \leq i \leq l, \tag{6}$$

where $\mathbf{h}_i$ is the output tensor, $\mathbf{h}_{i-1}$ is the input tensor, $\mathbf{W}_i \in \mathbb{R}^{N_i \times N_{i-1}}$ is the trainable weight matrix, $\mathbf{b}_i \in \mathbb{R}^{N_i}$ is the trainable bias vector, and σ is the non-linear activation function. In this case, we also define $\mathbf{h}_0 = \mathbf{x}(x, z, t)$ while the output layer $\mathbf{h}_{l+1}$ represents the AWE pressure field solution. The goal is to find the set of all tunable parameters, $\mathbf{W}_i$ and $\mathbf{b}_i$, for all L hidden layers that minimizes a loss function defined by the PDE and the IC terms:

$$L_{PDE} : \partial_t \partial_t p - c^2 \partial_j \partial_j p - f, \quad i = 1, 2 \tag{7}$$

$$L_{IC1} : \partial_t p(x, z, t = 0), \tag{8}$$

$$L_{IC2} : p(x, z, t = 0). \tag{9}$$

Thus, the loss function we aim to minimize is given by

$$L = \|L_{PDE}\|_2 + \beta_1 \|L_{IC1}\|_2 + \beta_2 \|L_{IC2}\|_2, \tag{10}$$

where $\|\cdot\|_2$ indicates the L^2 norm, and β_1 and β_2 are scalar weights to respectively control the relative contribution of the IC terms in the loss function.

Figure 2a presents a schematic representation of the PINN workflow using a DNN to approximate acoustic pressure wavefield. The NN outputs the pressure wavefield and computes the output spatial and temporal derivatives using automatic differentiation to evaluate the three loss function terms. The standard optimization goal is to update the NN parameters such that they globally minimize all three loss-term components. For this work we use NVIDIA SimNet package [19], based on the tensorflow framework, to solve the PINN optimization problem.

2.2.1 Fourier neural network

DNNs are generally biased toward low-frequency solutions [20] that create challenges for learning high-frequency wavefield components of AWE solutions. This is especially true for heterogeneous velocity models where NNs must learn sharp-gradient wavefield features. In this work, we use a Fourier neural network (FNN) for our PINN framework [17]. In FNNs, the input coordinates $\mathbf{x}(x, z, t)$ are mapped before being input to the NN as

$$\mathbf{h_0} = [\sin(2\pi \mathbf{F}\mathbf{x}); \cos(2\pi \mathbf{F}\mathbf{x})]^{\mathbf{T}}, \tag{11}$$

where $\mathbf{F} \in \mathbb{R}^{n_f \times N_0}$ is the trainable frequency matrix, n_f is the number of frequency sets, N_0 are number of neurons in first layer, and $\mathbf{T}$ is matrix transpose. The remainder of the layers remain same as in a FCNN in Eq. 6. This transformation allows the network to better learn the higher frequencies by mapping the inputs to a higher-dimensional feature space using high-frequency functions controlled by parameter $\mathbf{F}$. We refer the reader to existing work for further details [17, 19].

2.3 Enforcing initial conditions in PINNs

In the conventional PINN framework, the ICs and BCs are enforced as loss terms in the objective function used for network training. However, this will not necessarily satisfy the ICs because there is no guarantee that the loss terms will be identically zero during the optimization process. Therefore, enforcing the ICs in a soft manner as a loss term will not yield a unique solution to the specific PDE system in question.

To avoid this issue, we enforce the ICs using hard constraints by pre-multiplying a function $g(t)$ to the NN output such that $g(t = 0) = g'(t = 0) = 0$, where the prime indicates a temporal derivative and $t = 0$ s is the assumed time of source excitation. The NN input is still $\mathbf{x} = \mathbf{x}(x, z, t)$, but the NN output is now a transformed variable p^* that is related to p via

$$p = g\, p^*. \tag{12}$$

Using this expression, Eq. 7 can be rewritten as

$$\partial_t p = g' p^* + g\, \partial_t p^*. \tag{13}$$

When $t = 0$ s, Eqs. 12 and 13 are exactly zero and thus will satisfy the two ICs defined in Eqs. 8 and 9. Accordingly, there is no need to include terms L_{IC1} and L_{IC2} in our loss function, and thus we explicitly set $\beta_1 = \beta_2 = 0$ in Eq. 10. Figure 2b shows the modified PINN workflow dia-

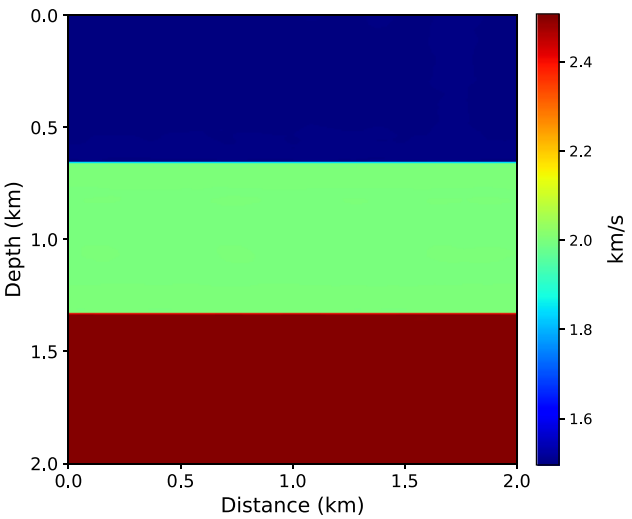

Fig. 7 Layered velocity model

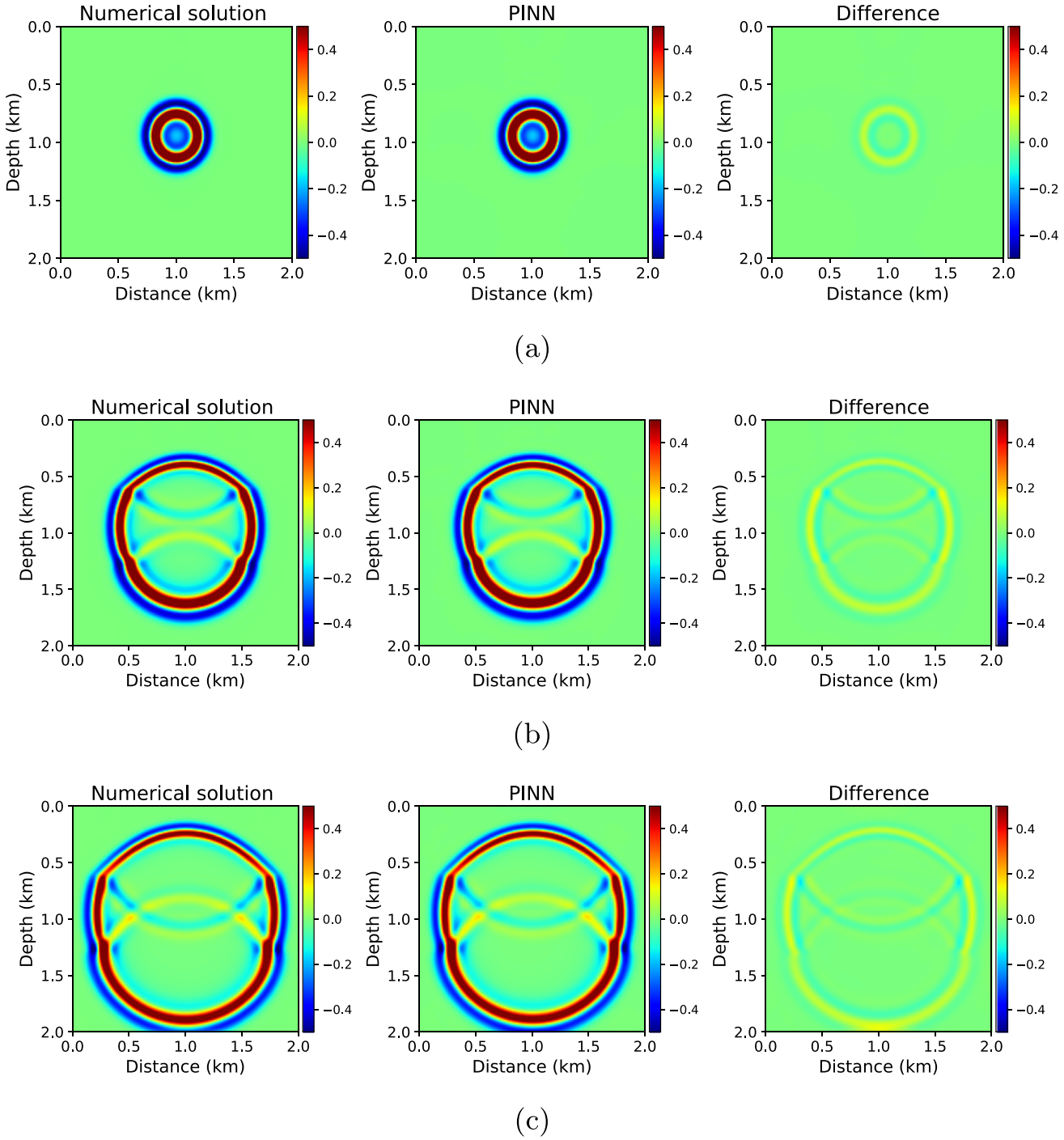

Fig. 8 Snapshots for layered velocity model. The left and center columns present the pseudo-spectral and PINN solutions while the right column shows the direct differences. The top (a) , middle (b) and lower (c) rows show the wavefield (differences) at times 0.15 s, 0.30 s, and 0.45 s, respectively

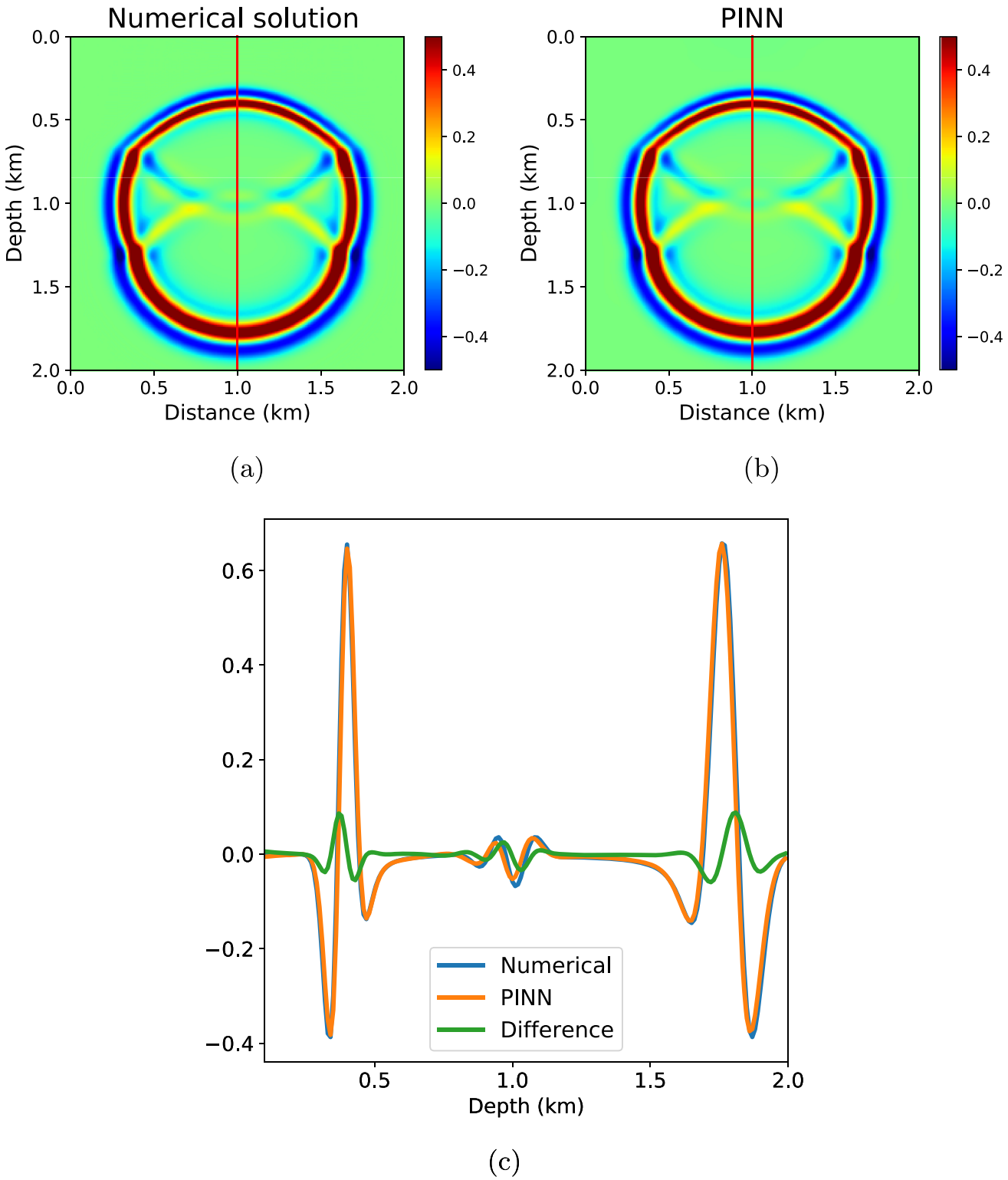

Fig. 9 Snapshots for layered velocity model at t=0.45 s. (a) Pseudo-spectral and (b) PINN wavefield solutions. (c) Wavefield differences computed at the vertical red line indicated in (a) and (b)

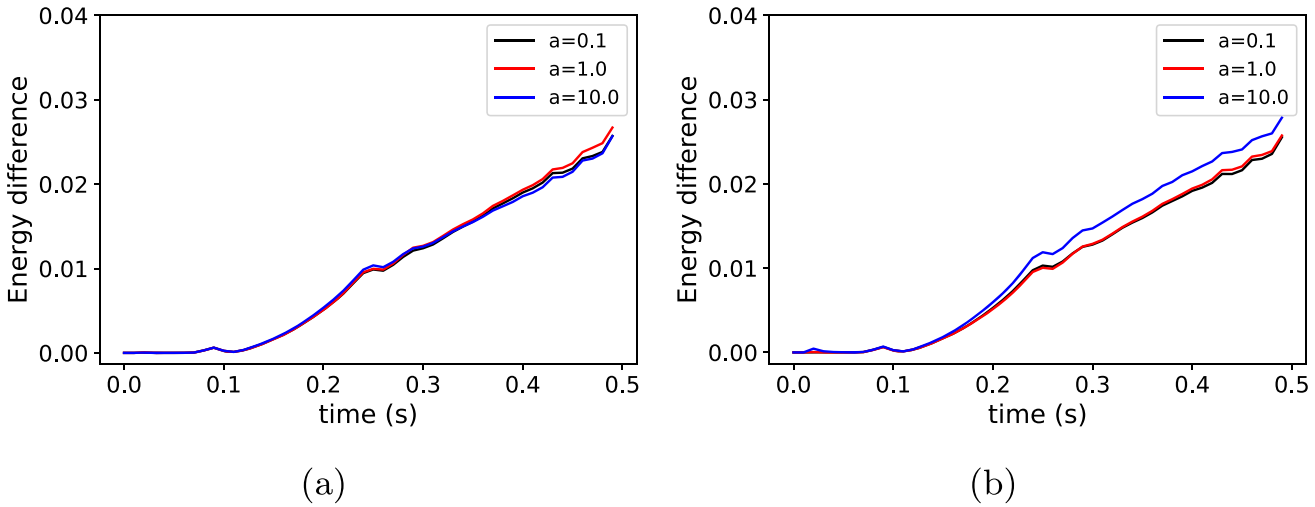

Fig. 10 Energy difference plots for functions with different scaling factors for layered velocity model: (a) $g_1 = a_1 t$ and (b) $g_2 = 1 - \text{sech}\, a_2 t$

gram with hard enforcement of the initial conditions, which significantly simplifies the NN.

For our numerical experiments we compared two functions that satisfy the two ICs in Eqs. 8 and 9:

$$g_1 = a_1 t^2, \tag{14}$$

$$g_2 = 1 - \text{sech}\,(a_2\, t)\,, \tag{15}$$

where a_1 and a_2 are scaling factors. We did not observe significant differences in the results as long as the chosen f_i function satisfies the ICs. We briefly discuss the numerical results from both functions in the Discussion section below.

3 Numerical examples

We test our PINN framework on both homogeneous and heterogeneous velocity models and employ FNNs for all numerical tests. We first consider a $N_z \times N_x = 200 \times 200$ homogeneous velocity model ($c = 1.5$ km/s) with $dx = dz = 10$ m spacing. We use a source term in Eq. 3 given by

$$f(x, z, t) = s(x, z)\, r(t), \tag{16}$$

which is the product of a 20 Hz Ricker wavelet $r(t)$ injected as a spatially distributed Gaussian source $s(x, z)$ centered at

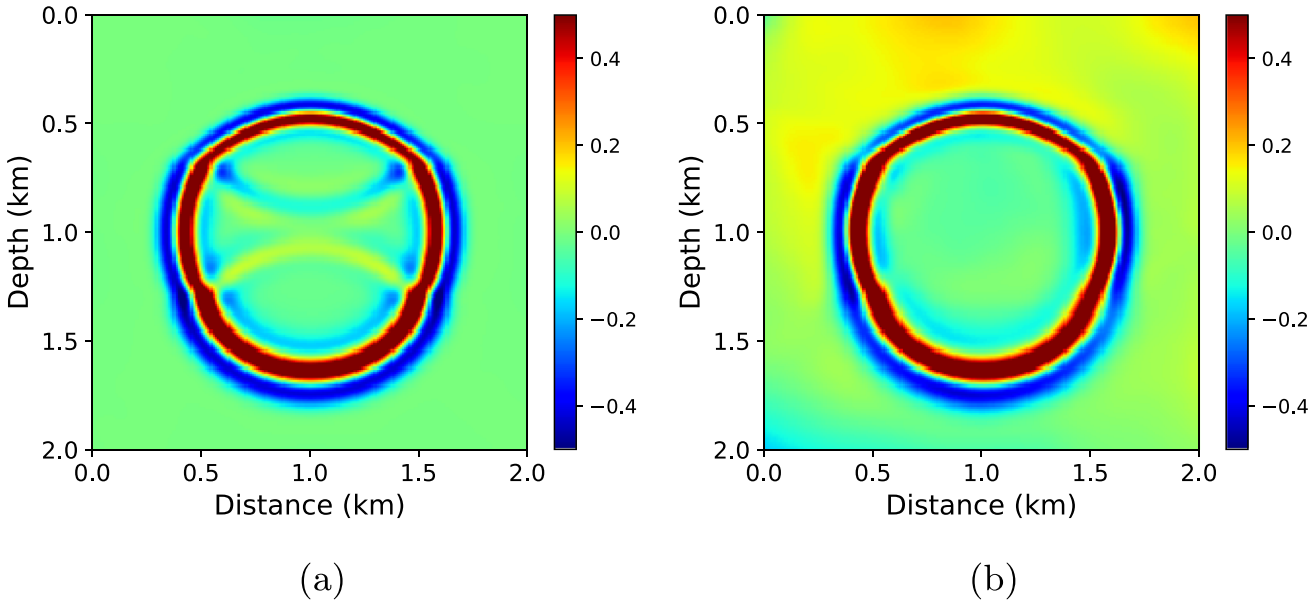

Fig. 11 Snapshots for layered velocity model at t=0.3 s using the (a) FNN and (b) FCNN approaches

$[s_x, s_z] = [1.0, 1.0]$ km with weights given by

$$s(x, z) = \mathrm{e}^{-10000\left((x-s_x)^2+(y-s_y)^2\right)}. \tag{17}$$

We run training for 60,000 iterations and compare the computed PINN pressure field to a solution simulated using a highly accurate pseudo-spectral method [21].

3.1 Homogeneous model

For the first numerical test we use a homogeneous model with a velocity of 1.5 km/s. Figure 3 shows the wavefield snapshots for the pseudo-spectral method (left column), the PINN approach (center column), as well as their respective direct wavefield differences (right column).

The training points [i.e., the $\mathbf{x}(x, z, t)$ coordinates] were randomly sampled across the whole computational domain up to 0.5 s to compute the PDE loss function. We defined a circular region of radius 0.2 km around the source location and used 2^{16} points in total to ensure sufficient sampling in the neighborhood of the source location. A total of 2^{11} points were used in the remaining 4.0 km^2 area to randomly sample the rest of the rectangular propagation domain. The top, center, and bottom rows show the evolution of the wavefield at 0.15 s, 0.30 s and 0.45 s, respectively. The PINN algorithm has accurately learned the phases of the direct arrivals; however, there are minor observable amplitude mismatches.

Figure 4 shows the wavefield snapshots at 0.45 s and the amplitude variation with depth extracted at the red line. The normalized difference between FNN and pseudo-spectral solution is approximately 2.5%. Figure 5 shows the energy difference between the FNN and pseudo-spectral solutions for the homogeneous model using g_1 and g_2 functions with different scaling parameters computed as

$$E(t) = \frac{\sum_{N_x}\sum_{N_z}\left(p_{NN}(x, z, t) - p_{SP}(x, z, t)\right)^2}{\sum_{N_x}\sum_{N_z} p_{SP}^2(x, z, t)}, \tag{18}$$

where $p_{NN}(x, z, t)$ and $p_{SP}(x, z, t)$ are the NN and pseudo-spectral solutions, respectively. We observe the energy differences for $g_1 = a_1 t^2$ and $g_2 = 1 - \mathrm{sech}(a_2 t)$ are negligible and the solutions from other scaling parameters for both functions are comparable.

We also compare our solution with PINN with soft constraints using Eq. 12 with $\beta_1 = \beta_2 = 1$. Figure 6 shows wavefield snapshots after 60,000 iterations for both experiments. The solution using soft constraints fails to converge as it does not enforce the condition that $p(x, z, t = 0) = 0$, which is required to make a unique solution to the PDE system. We also tried increasing the number of iterations; however, this approach did not improve the results.

3.2 Layered model

We next consider the 200 × 200 heterogeneous velocity model with three layers shown in Fig. 7. We again inject a 20 Hz Ricker wavelet as a spatially distributed source centered at $[s_x, s_z] = [1.0, 1.0]$ km and use the same FNN for training, but now run a total of 100,000 training iterations.

The left and center columns of Fig. 8 respectively show wavefield snapshots for the pseudo-spectral and PINN solutions at different time steps, while the right column presents the direct wavefield differences. The training points [i.e., $\mathbf{x}(x, z, t)$ coordinates] were randomly sampled across the whole computational domain up to 0.5 s to compute the PDE loss function. The top to bottom rows respectively show the wavefields at 0.15 s, 0.30 s and 0.45 s. We note that the PINN solver has learned the internal reflections generated by the layers.

Figure 9 shows the snapshot at t=0.45 s and the amplitudes extracted at the red vertical line. The normalized difference between NN and the pseudo-spectral solution is approximately 2.0%. Figure 10 shows the energy difference between the FNN and pseudo-spectral solutions for the layered mode using g_1 and g_2 functions with different scaling parameters. The energy difference computed using both functions are comparable and are less than 3.0% throughout the entire computational domain.

Using FNN improved the convergence rate of the training and the overall results compared to FCNN. Figure 11b shows the wavefield snapshots using FCNN after 200,000 iterations that spanned roughly five hours of training time. The PINN solution cannot learn the internal reflectivity from discontinuous model structure. However, the FNN solution presented in in Fig. 11a converges in 100,000 iterations or about 2.5 hours of training time.

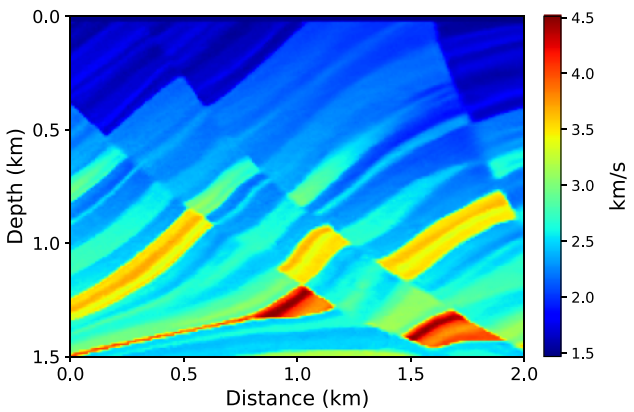

Fig. 12 Modified Marmousi velocity model

3.3 Marmousi model

Finally, we test our algorithm on modified Marmousi model (see Fig. 12). A 20 Hz Ricker wavelet is injected as a spatially distributed source centered at $[s_x, s_z]$=[1.0,0.75] km. Figure 13 shows the snapshots at different time steps and the PINN and the pseudo-spectral numerical solution difference. The training points [i.e., the $\mathbf{x}(x, z, t)$ coordinates] were randomly sampled across the whole computational domain up to 0.35 s to compute the PDE loss function. We observe good phase agreement between the arrivals originating from sharp velocity model gradients; however, there are again fairly minor pressure wavefield amplitude differences (see Fig. 14).

Figure 15 shows the energy difference between the FNN and pseudo-spectral solutions for the modified Marmousi model using g_1 and g_2 functions with different scaling parameters. The energy difference computed using both functions are comparable and are less than 2.0% throughout the full computational domain.

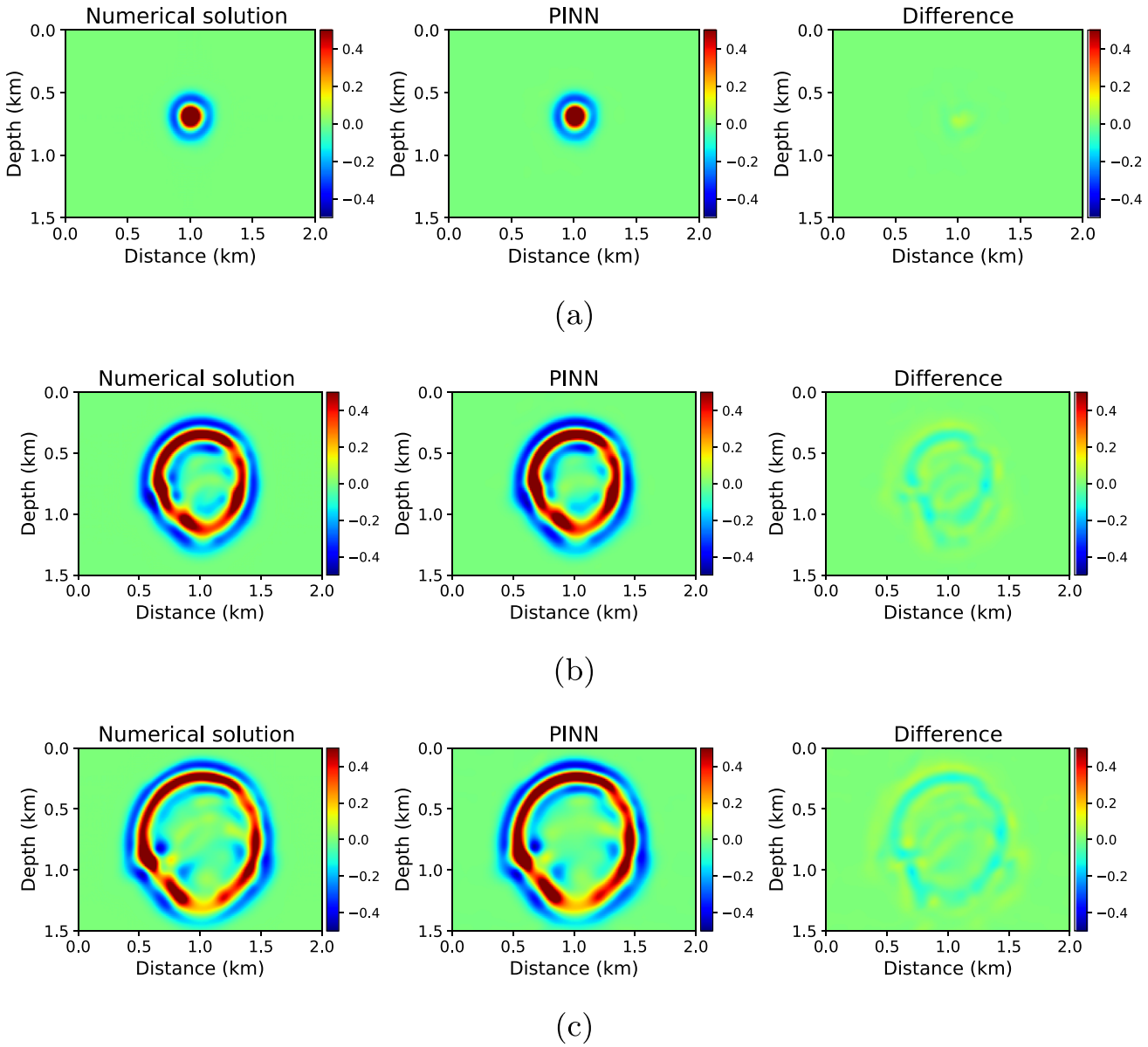

Fig. 13 Snapshots for modified Marmousi velocity model. The left and center columns present the pseudo-spectral and PINN solutions while the right column shows the direct differences. The top, middle and lower rows show the wavefield (differences) at times 0.15 s, 0.30 s, and 0.45 s, respectively

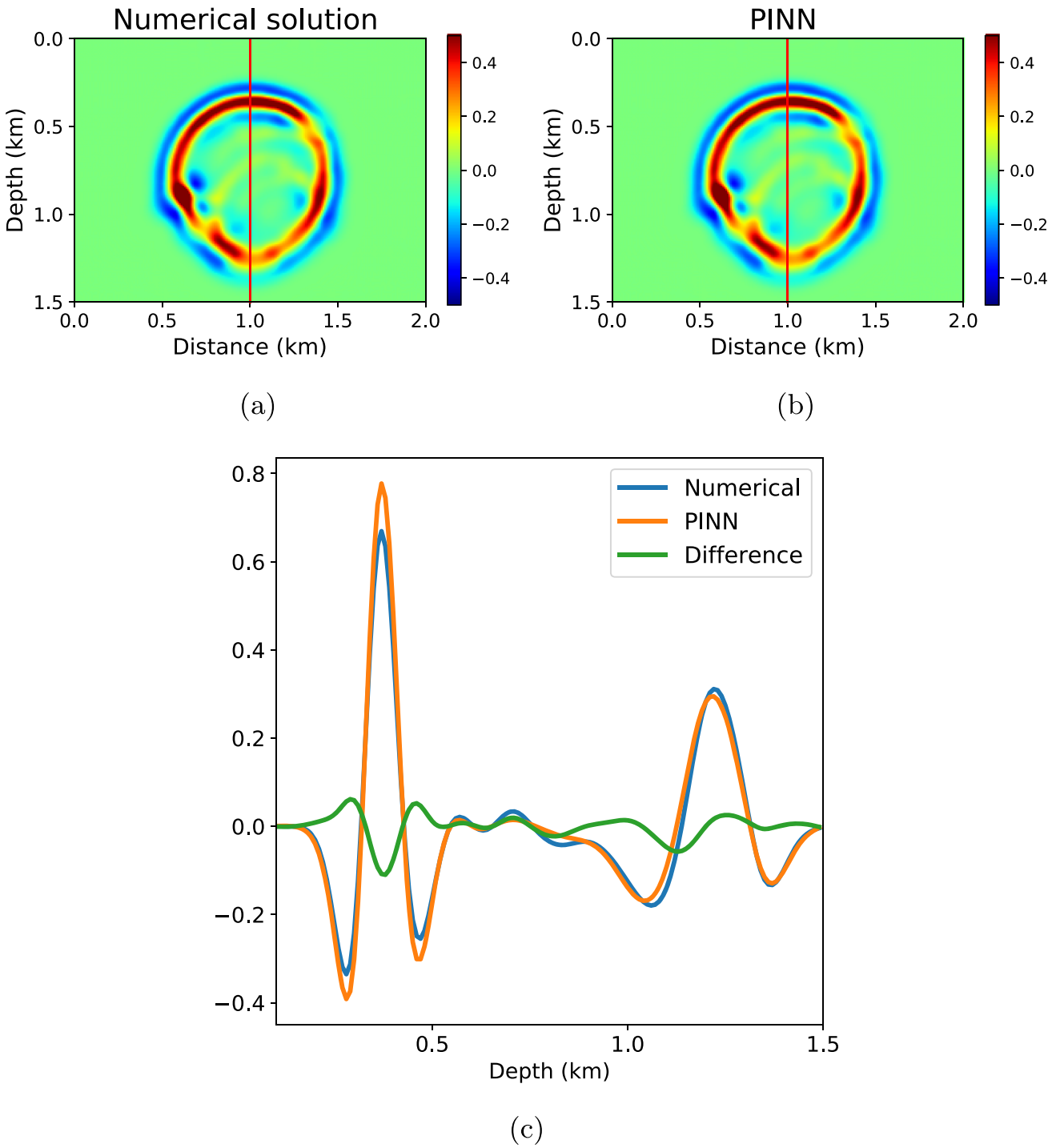

Fig. 14 Snapshots for modified Marmousi model at t=0.55 s. (a) Pseudo-spectral and (b) PINN wavefield solutions. (c) Wavefield differences computed at the vertical red line indicated in (a) and (b)

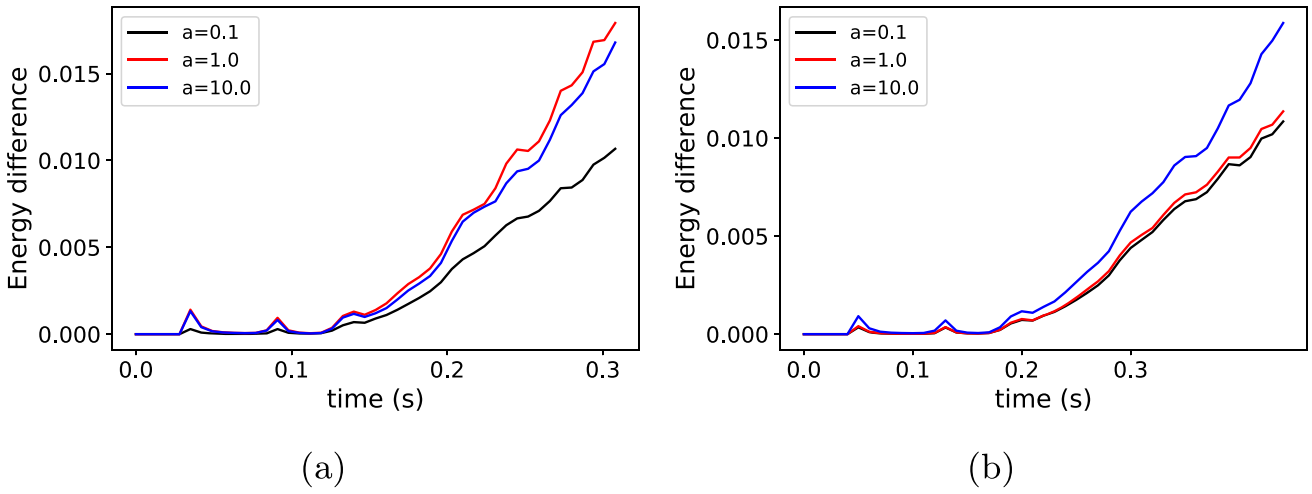

Fig. 15 Energy plots for functions with different scaling factors for modified Marmousi model: (a) $g_1 = a_1 t$ and (b) $g_2 = 1 - \operatorname{sech} a_2 t$

4 Discussion

The time-domain PINN algorithm can learn AWE wavefield solutions for both homogeneous and more complex heterogeneous models. However, there are still challenges in learning amplitudes in the absence of training data. FNN training requires choosing a set of frequencies. For homogeneous and layered velocity models, we opted for 20 frequencies uniformly distributed between 0-4 Hz in Eq. 11. For the Marmousi model, though, we increased this to 50 frequencies between 0-10 Hz. We also experimented with including higher frequencies; however, this approach increased the overall training time while yielding minimal improvements to the observed amplitude accuracy. Thus, we held the number of layers at eight, the number of neurons in each layer at 256, and employed the swish activation function [22].

We also observe an accuracy improvement with increasing the number of layers from eight to ten and increasing the number of neurons in each layer from 256 to 1024; however, this drastically increased the total training time. Figures 5, 10 and 15 show the normalized energy difference (using Eq. 18) computed between the PINN and pseudo-spectral solution for homogeneous, layered, and modified Marmousi velocity models, respectively. We observe the error to be in similar range for different scaling parameters using both test functions. We choose $f(t) = 0.1t^2$ as our test function for the numerical experiments to be consistent. From the numerical experiments, we observe that the PINN approach accurately matches the phases of different wavefield arrivals; however, we assert that further work is required to learn more accurate wavefield amplitudes.

Another challenge is extending the PINN solution beyond the training domain. While our approach does not require the wavefield solution for training the NN, it still requires the spatial and temporal coordinates as inputs for wavefield prediction when computing the loss function. In homogeneous models, we selected the temporal solution interval to be [0-0.5] s; however, the FNN prediction after 0.6 s was unstable. For homogeneous and modified Marmousi model, the FNN was capable of accurately estimating wavefield solutions at longer simulation times (up to 0.8 s) than in the homogeneous model. PINN approaches, though, still face challenges in learning solutions for longer run times in the absence of training data.

5 Conclusions

We developed a time-domain PINN approach to learn the AWE wavefield solutions without using training data. We used hard enforcement of the initial conditions of the PDE system by multiplying a temporal function to the network output, which proved to be crucial for training convergence. The comparison of PINN-based results with those from the pseudo-spectral method demonstrates that latter produces accurate phases of different arrivals; however, wavefield amplitudes exhibited minor inaccuracies even for heterogeneous velocity models. The normalized energy difference between both solutions were less than 4% throughout the computational domain for all of our numerical tests. Our quantitative results suggests that the FNN performs slightly better for models with sharp gradients. However, challenges remain in training FNN-based PINN solvers for longer run times; the observed solution divergence as the propagation time increases requires investigation. Thus, further refinement of this approach is deemed necessary for applications where a high-degree of amplitude fidelity and/or longer simulations are required.

Acknowledgements The authors would like to acknowledge Oliver Hennigh, Sanjay Choudry, and NVIDIA Simnet team for technical help as well as the sponsors of the Center for Wave Phenomena research consortium at the Colorado School of Mines. We thank the two anonymous reviewers for helpful comments that improved quality of the manuscript. We also acknowledge the Mines HPC facility for compute time allocations on the *Wendian* cluster.

Funding This research was partially sponsored by Center for Wave Phenomena research consortium at the Colorado School of Mines.

Code Availability The code can be accessed by contacting the authors.

Declarations

Competing interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. 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)
2. Baydin, A.G., Pearlmutter, B.A., Radul, A.A., Siskind, J.M.: Automatic differentiation in machine learning: a survey. J. Mach. Learn. Res. **18**(153), 1–43 (2018)
3. Sun, L., Gao, H., Pan, S., Wang, J.-X.: Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data. Comput. Methods. Appl. Mech. Eng. **361**, 112732 (2020)
4. Jin, X., Cai, S., Li, H., Karniadakis, G.E.: NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. J. Comput. Phys. **426**, 109951 (2021)
5. Kissas, G., Yang, Y., Hwuang, E., Witschey, W.R., Detre, J.A., Perdikaris, P.: Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4D flow MRI data using physicsinformed neural networks. Comput. Methods Appl. Mech. Eng. **358**, 112623 (2020)
6. Tanios, R.: Physics Informed Neural Networks In Computational Finance: High Dimensional Forward & Inverse Option Pricing. Master's thesis (2021)
7. Jiang, X., Wang, D., Fan, Q., Zhang, M., Lu, C., Lau, A.P.T.: Solving the Nonlinear Schrödinger Equation in Optical Fibers Using Physics-informed Neural Network. In: 2021 Optical Fiber Communications Conference and Exhibition (OFC), pp. 1–3 (2021). IEEE
8. Zhang, R., Liu, Y., Sun, H.: Physics-guided convolutional neural network (PhyCNN) for data-driven seismic response modeling. Eng. Struct. **215**, 110704 (2020)
9. Sahli Costabal, F., Yang, Y., Perdikaris, P., Hurtado, D.E., Kuhl, E.: Physics-informed neural networks for cardiac activation mapping. Frontiers in Physics **8**, 42 (2020)
10. Waheed, U., Haghighat, E., Alkhalifah, T., Song, C., Hao, Q.: PINNeik: Eikonal solution using physics-informed neural networks. Computers & Geosciences **155**, 104833 (2021)
11. Moseley, B., Markham, A., Nissen-Meyer, T.: Solving the wave equation with physics-informed deep learning (2020)
12. Karimpouli, S., Tahmasebi, P.: Physics informed machine learning: Seismic wave equation. Geosci. Front. **11**(6), 1993–2001 (2020)
13. Rasht-Behesht, M., Huber, C., Shukla, K., Karniadakis, G.E.: Physicsinformed neural networks (pinns) for wave propagation and full waveform inversions. J. Geophys. Res. Solid Earth **127**(5), 2021–023120 (2022)
14. Alkhadhr, S., Liu, X., Almekkawy, M.: Modeling of the forward wave propagation using physics-informed neural networks. In: 2021 IEEE International Ultrasonics Symposium (IUS), pp. 1–4 (2021). https://doi.org/10.1109/IUS52206.2021.9593574
15. Song, C., Alkhalifah, T., Waheed, U.B.: Solving the frequency-domain acoustic VTI wave equation using physics-informed neural networks. Geophys. J. Int. **225**(2), 846–859 (2021)
16. Song, C., Alkhalifah, T., Waheed, U.B.: A versatile framework to solve the helmholtz equation using physics-informed neural networks. Geophys. J. Int. **228**(3), 1750–1762 (2022)
17. Tancik, M., Srinivasan, P.P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J.T., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. arXiv preprint arXiv:2006.10739 (2020)
18. Engquist, B., Majda, A.: Absorbing boundary conditions for numerical simulation of waves. Proc. Natl. Acad. Sci. **74**(5), 1765–1766 (1977)
19. Hennigh, O., Narasimhan, S., Nabian, M.A., Subramaniam, A., Tangsali, K., Fang, Z., Rietmann, M., Byeon, W., Choudhry, S.: NVIDIA SimNet^{TM}: An AI-accelerated multi-physics simulation framework. In: International Conference on Computational Science, pp. 447–461 (2021). Springer
20. Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., Courville, A.: On the spectral bias of neural networks. In: International Conference on Machine Learning, pp. 5301–5310 (2019). PMLR
21. Carcione, J.M.: A generalization of the Fourier pseudospectral method. Geophysics **75**(6), 6–5356 (2010)
22. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv preprint arXiv:1710.05941 (2017)