Numerical Dispersion Mitigation Neural Network with the Model-Based Training Dataset Optimization

Elena Gondyul[1(✉)], Vadim Lisitsa[1], Kirill Gadylshin[2], and Dmitry Vishnevsky[1]

[1] Institute of Petroleum Geology and Geophysics SB RAS, 3 Koptug ave, Novosibirsk 630090, Russia
e.gondyul@g.nsu.ru, lisitsavv@ipgg.sbras.ru
[2] Sobolev Institute of Mathematics SB RAS, 4 Koptug ave., Novosibirsk 630090, Russia

Abstract. A neural network is used to approximate the transition operator from seismic data modeled on a large computational grid to data obtained on a small one. Thus, we obtain an effective way of suppressing numerical dispersion in numerically modeled seismic fields. This article discusses a method for constructing an optimal training dataset based on the properties of a velocity model. We build a distance matrix for the parts of the model that correspond to the positions of the sources and build a dataset in such a way that the distance between the training set and all sources is limited.

Keywords: Deep learning · seismic modelling · numerical dispersion

1 Introduction

The primary method for solving a system of elastodynamic equations is the finite difference method [16]. However, its use often leads to numerical dispersion in the solution, which may be inappropriately high if a coarse grid is used. Numerical dispersion in seismic modeling is a specific type of numerical error that appears in the odd-order derivatives present in the approximation error of the differential operator. To suppress the numerical dispersion, several pre-processing and post-processing algorithms were suggested [11]. However, they are computationally intense. Moreover, they can treat the dispersion caused by the discretization of the temporal derivatives, whereas mitigation of spatial dispersion is still an unsolved problem.

Indeed, advanced numerical methods may be applied to improve accuracy. These include the high-order finite-difference schemes, optimization schemes that

V.L. developed the algorithm of optimal dataset construction, D.V. performed seismic modeling, E.G. performed numerical experiments on NDM-net training under the support of RSF grant no. 22-11-00004, K.G. optimized the NDM-net hyperparameters under the support of RSF grant no. 22-11-00104

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14106, pp. 19–30, 2023.
https://doi.org/10.1007/978-3-031-37111-0_2

suppress the numerical error [9], the discontinuous Galerkin method and spectral element methods[5,8,14]. The above methods do not necessarily lead to a reduction in computational complexity. As the order of approximation increases, the number of operations and the memory used increases.

A different approach which was recently suggested is based on the use of machine learning methods to suppress numerical dispersion [2,3]. The idea of the approach is to approximate a nonlinear transition operator from synthetic data calculated on a coarse grid to data modeled on a fine grid using the machine learning technique. This approach makes it possible to suppress variances and artifacts to a large extent with less memory and time consumption than the other post-processing techniques.

However, building a neural network is a challenging task that requires careful consideration of various factors. One such factor is the selection of appropriate hyperparameters. This can be a time-consuming process as it involves a lot of trial and error. Another critical aspect of constructing neural networks is the creation of an optimal training dataset. A well-designed dataset can provide a high degree of generalization, enabling the network to perform well on unseen test data. The importance of creating a good dataset cannot be overstated, as the neural network's ability to learn and make accurate predictions heavily depends on the data it is trained on. This work aims to create a representative sample for training a neural network.

2 Seismic Modeling

The seismic data we use in this research was obtained by solving the elastic wave equation defined in a 2D unbounded domain. We are using the finite difference method to numerically solve a set of these equations with different right-hand sides (sources). The entire dataset is generated using a coarse enough mesh with fewer than 5 points per dominant wavelength. The coarse grid solution can be generated relatively fast. The training dataset includes the seismograms, which are simulated using fine enough mesh, i.e., with 10 or more grid points per wavelength. Generating this dataset takes most of the computational time, thus we need to minimize the number of seismograms in this dataset. In both cases, the formal problem can be stated as:

$$L[\boldsymbol{w}(t,\boldsymbol{x})] = \boldsymbol{f}(t,\boldsymbol{x}^{\boldsymbol{s}}),$$

where $\boldsymbol{w}-$ the solution of elastic wave equation, $\boldsymbol{f(t,x^s)}$ is the time-dependent right-had size or external forces, $\boldsymbol{x^s}$ is the source position. Also, from this problem we get the velocity components on the receiver's positions

$$\boldsymbol{w_s}(t,\boldsymbol{x}^{\boldsymbol{r}}),$$

where $\boldsymbol{x^r}$ is the receiver positions.

And, for a finite-difference approximation of the differential operator with the fourth-order of approximation in space and second-order in time, we can rewrite our problem as

$$L_h[\boldsymbol{w_h}] = \boldsymbol{f_h}(t,\boldsymbol{x}^{\boldsymbol{s}}),$$

where $\boldsymbol{f_h}$ is the approximation of the right-hand side, $\boldsymbol{u_h}$ is the finite-difference solution corresponding to the grid with the step h.

According to the general theory of the finite difference schemes, if two grids with steps h and H so that $h < H$ are considered, then the errors

$$\|\boldsymbol{w}_h - [\boldsymbol{w}]_h\| \leq \varepsilon_h, \|\boldsymbol{w}_H - [\boldsymbol{w}]_H\| \leq \varepsilon_H,$$

satisfy the inequality $\varepsilon_h < \varepsilon_H$. So, we manage to construct a map $\boldsymbol{w}_{ML} = G[\boldsymbol{w}_H]$ that transforms the coarse grid data to the fine grid data with appropriate accuracy; i.e., $\|\boldsymbol{w}_{ML} - \boldsymbol{w}_h\| = \varepsilon_{ML} << \varepsilon_H$, we may significantly improve the coarse mesh solution without simulation of the entire dataset on a fine mesh.

3 Network Architecture

We apply a neural network based on U-net architecture [15] to translate data modeled on a coarse grid into data modeled on a fine grid. The U-net architecture is composed of an encoder and a decoder. The encoder captures the context of the image by reducing the spatial resolution of the input image through a series of convolutional and pooling layers. The decoder then recovers the original spatial resolution by using a series of deconvolutional and upsampling layers. This network, called here as NDM-net (Numerical Dispersion Mitigation Network), contains 16 convolutional layers: 8 upsampling layers and 8 concatenation layers. The depth of the network was set according to the dimensions of the input images. The layers consist of the convolution operator, the batch normalization operator, and the Rectified Linear Unit (ReLU) as the activation function. At the convolutional stage, layers have LeakyRelu instead of ReLu with a negative slope factor of 0.2. Figure 1 shows the U-net architecture. Dropout is used to reduce overfitting and generalization errors as well as early stopping.

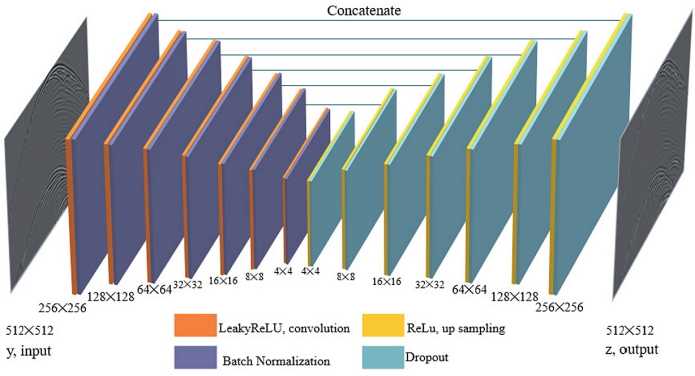

Fig. 1. U-net architecture

Let $(Y, Z) = \{y_i, z_i\}_{i=1}^N$ be training pairs consisting of the N-th number of seismograms modeled on a coarse grid and a fine grid, respectively.

The NDM-net learns all the parameters of the model by minimizing the loss function, defined as following

$$L = \sum_i ||G(y_i) - z_i||_1$$

where $G : y_i \rightarrow z_i$ is the generated set of seismograms.

The update of the parameters occurs according to the chain rule and is derived as

$$\theta_l = \theta_l - \gamma(\delta l / \delta \theta_l),$$

where γ is the learning rate, which changes its value with the system's evolution, and $\delta l / \delta \theta_l$ is acquired from the last layer to layer l by the chain rule.

It should be noted that the results of training a neural network are highly dependent on the input data. Therefore, it is difficult to talk about the applicability of this architecture for various noisiness.

Also, the choice of a neural network is not limited to this network, we believe that cGan [6,10], pix2pix [4] and others can be promising architects in the seismic data processing. There are many applications of neural network in seismic data processing and interpretation, some use deep convolutional neural network (CNN) for denoising and ground-roll noise attenuation [7,17], another use cycle-Gan, cGan for interpolation[6,12], etc.

4 Input Data

We simulate the seismic data set using the BP subsurface model shown in Fig. 2. This model was built on a $12.5m \times 6.25m$ grid and had a size of 67 km long and 12 km deep. The velocity V_p varies from 1430 m/s to 4790 m/s.

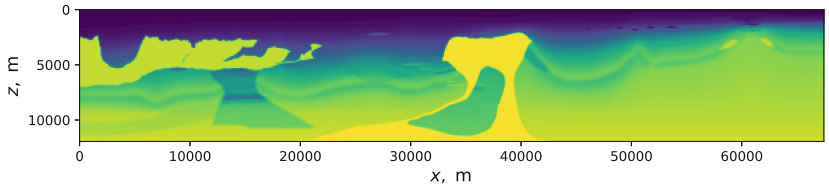

Fig. 2. BP model

The right part of the model consists of a simple background and slowly changes. In contrast, the central and left part is more complex and have a salt body and sharp velocity.

The acquisition included 2696 sources with a distance of 25 m. We recorded the wavefield by 2401 receivers for each shot with maximal source-receiver offsets of 15 km. The receivers were 12.5 m apart from one another. For this model, we

generated seismograms using a grid with a step h_x of 6 m (coarse grid) and 3 m (fine grid). The source function was the Ricker wavelet with a center frequency is 30 Hz. The time sampling step is 3 ms.

We cropped seismograms to 512×512 points as shown in Fig. 3.

5 Training Dataset Construction

An important stage of neural network training is the creation of the training dataset. This section presents three methods for constructing training datasets and evaluating their effectiveness. In order to assess whether the generalization to the test data is sufficiently well carried out, we will use the Normalized Mean Squared Error (nMSE). The nMSE is defined by the following expression

$$nMSE(y_i, z_i) = \frac{1}{2} \frac{\|y_i - z_i\|_2}{\|y_i\|_2 + \|z_i\|_2}.$$

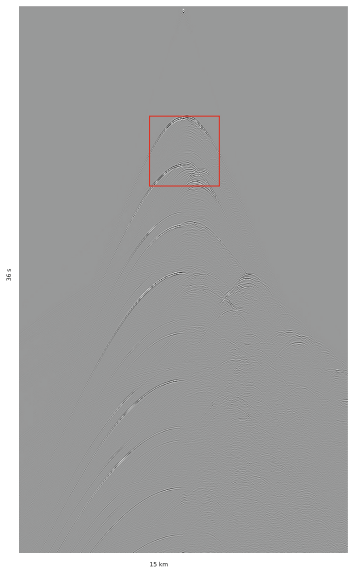

Fig. 3. A example of a seismogram. The red rectangle corresponds to the cropped zone. (Color figure online)

We also need to introduce the average error between two datasets containing the same number of samples:

$$R(Y, Z) = \frac{1}{N} \sum_i nMSE(y_i, z_i).$$

This formula is used throughout the paper to estimate the distance between the coarse and fine-grid data, as well as between the generated dataset $U^{ML} = \{G(y_i)\}_{i=1}^N$ and dataset Z modeled on a fine grid.

5.1 Equidistantly Distributed Sources

The simplest sampling option is to include seismograms with correspond to equidistantly distributed sources to cover various combinations of input variable values. The set of indices for such a training dataset is designed as follows:

$$J_k^{eq} = 1, k, 2k, ..., q(k)k,$$

where $k \in \mathbf{N}$, $k > 1$, and $q(k) = [N/k]$ that is the integer part of the ratio.

We constructed three datasets for the BP model: $D_{10}^{eq}, D_5^{eq}, D_{2.6}^{eq}$, which corresponds to datasets with $10\%, 5\%, 2.6\%$ of the total number of sources, respectively.

The number of seismograms in the training data, the average error between the generated set and set Y, and the distance between the initial sets for each dataset are given in Tables 1 and 2.

Table 1. Description of the datasets with equidistantly distributed for the BP model

Dataset	Number of sources	$R(U^{ML}, Z)$
$D_{2.6}^{eq}$	68	0.206
D_5^{eq}	135	0.168
D_{10}^{eq}	270	0.138

Table 2. Description of all dataset for the BP model

Dataset	Length of all dataset	R(Y,Z)
Initial	2696	0.469

5.2 Model-Based Dataset

In our previous research [1] we illustrated that the seismograms corresponding to closely spaced sources are similar. Moreover, the similarity of the seismograms correlates with the similarity of the model where the wavefield propagates. In this paper, we consider the models for which the seismogram is generated and construct the training dataset that corresponds to the most typical models. Note that the original model is elongated in a horizontal direction, exceeding the active acquisition system for a fixed source position. Thus, we may restrict the model for a particular source position $\boldsymbol{x}_j^s$ by the size of the active acquisition. As a result, we may consider a set of models $\mathrm{M} = \{M_1, ..., M_{N_s}\}$ with the number of models equal to the number of sources.

For example, the distance between M_{j+1} and M_j is calculated by the Normalized Mean Squared Error. Thus, a distance matrix was obtained (Fig. 4), which shows how the model changes depending on the distance between the sources.

Then, we suggest building the training sample in such a way as to limit the maximum distance between the entire data set and the training set.

So, we have a max-min problem:

$$\max_{k\in 1,..,N} \min_{j\in J_t} nMSE(M_k, M_j) \leq Q,$$

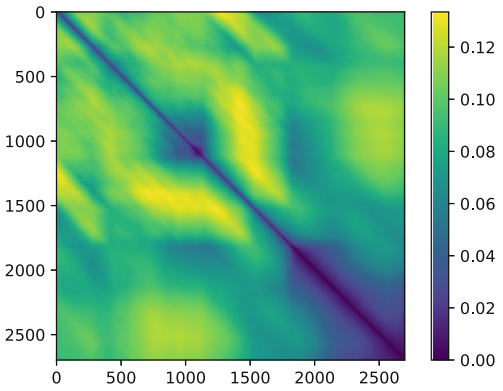

Fig. 4. Distance matrix

where J_t is a set of training dataset sources indices and Q is the desired error level.

Thus, the obtained data sets (Fig. 5) are dense in the areas where the model is highly variable in the horizontal direction, but they are sparse where the model is close to the horizontally layered. Description of the samples used and errors are presented in the Table 3

Table 3. Description of the datasets with L_2-preserving for the BP model

Dataset	Number of sources	$R(U^{ML}, Z)$
$D^M_{2.6}$	70	0.223
$D^M_{4.9}$	133	0.159
$D^M_{10.2}$	275	0.136

5.3 Comparison of the Sampling Techniques

We will use nRMS, as well as the Structural Similarity Index (SSIM) [13], to check the similarity of the generated seismograms with the original ones[1]. The last metric deals with the structural elements of images. SSIM is calculated for

[1] We can add other metrics.

each overlapping image block using a pixel-by-pixel sliding window, and therefore it can provide a distortion/similarity map in the pixel domain. The following equation determines this metric

$$SSIM(x_i, y_i) = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

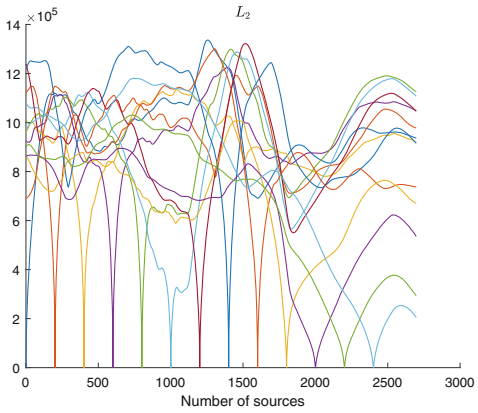

Fig. 5. The L_2 distances between the parts of the model as a function of the distance between the sources for BP model

where μ_x and σ_x are the mean intensity and standard deviation of x_i,μ_y and σ_y are the mean intensity and standard deviation of y_i, σ_{xy} is the covariance of x_i and y_i, C_1 and C_2 are the constants in order to maintain stability. The closer this indicator is to one, the more similar the seismograms are. The above metric is used to compare a set of samples, the average values for all training sets are shown in the Table 4

Table 4. SSIM

Dataset	$SSIM(U^{ML}, Y)$
$D^M_{2.6}$	0.993
$D^M_{4.9}$	0.996
$D^{eq}_{2.5}$	0.994
D^{eq}_{5}	0.995

Another metric for evaluating the effectiveness of a neural network on different samples is nMSE (Fig. 6)

Compared to results obtained using equidistant sources, the optimized datasets provide a smaller error for a given number of training data. Note that

the greater the number of seismograms in the set for all types of samples, the more accurate the results. However, the training results on an optimal sample provide less benefit when the distance matrix has a value close to zero, even with a large amount of training data.

6 Numerical Results

In this section, we present the result of our experiments, obtained from applying the neural network to the data described above. In order to show what the generated traces look like, we take the simplest way to construct the training set - equidistantly distributed sources. The networks are trained on a workstation with one NVIDIA GeForce RTX 3090 with 8 GB memory and Intel Core CPU with 256 GB RAM.

In order to show a qualitative result, we take the most optimal training set D_{10}^{eq}. The neural network takes 3 seconds to train one epoch on GPU. An early stop to prevent overfitting was at epoch 500. Thus, it will take at least 30 minutes to get acceptable results. While the application of a neural network to test data requires less than a second.

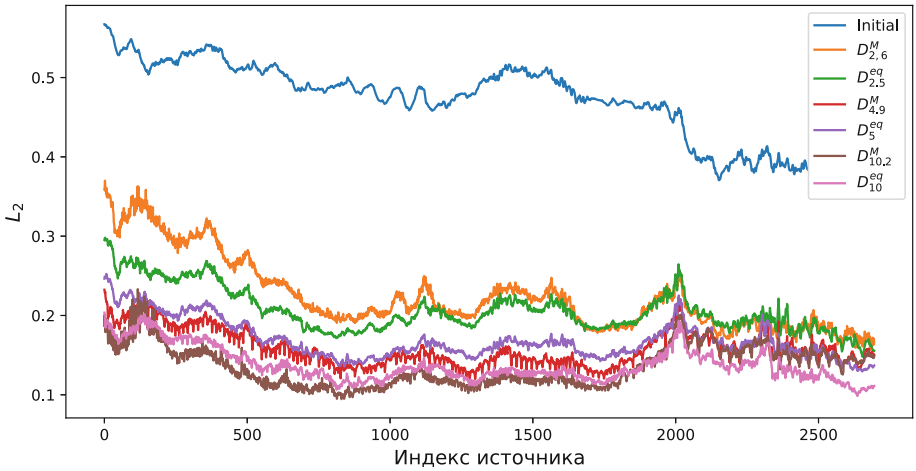

Fig. 6. The pair-wise L_2 distances between the fine-grid data and the NDM-net predicted data for different datasets.

Figure 7 shows traces, one of which is generated by a neural network $G(y)$ and the other two are reference ones. Below, in Fig. 8, the difference between generated trace and z is presented, as well as difference between y and z.

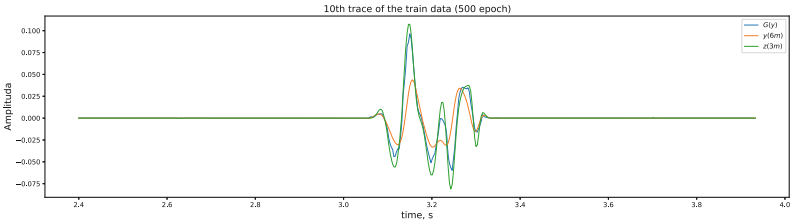

Fig. 7. Seismic traces $y(6m)$ and $z(3m)$ and generated trace $G(y)$ by NDM-net

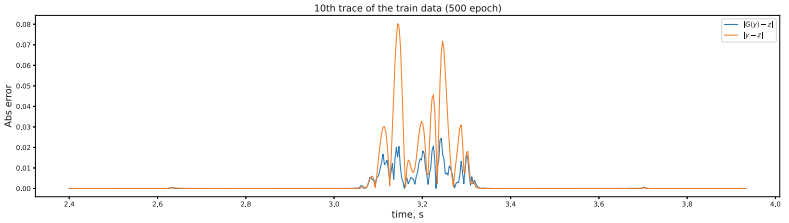

Fig. 8. Difference between generated trace $G(y)$ and trace $z(3m)$ and difference between reference data

7 Conclusion

We used a neural network to suppress numerical dispersion in seismic data. Specifically, we approximated the transition operator from data modeled on a coarse grid to data modeled on a fine grid. Our training results demonstrate that this method effectively detects correlation noise, numerical dispersion and suppresses a significant portion of numerical errors.

We created an optimal training set to improve the neural network's performance and speed up learning. Suppose we choose the training set so that the number of seismograms in it is minimal and at the same time representative, then we will get more efficient results. In this study, we consider two approaches to constructing a training dataset: a method based on equidistantly distributed sources and a method that constructs an optimized sample based on the properties of the velocity model.

The second method to construct the training dataset is limiting the distance calculating for the parts of the model that correspond to the positions of the velocity model. We are assumed to choose a limiting value and thus construct a set of sources and their corresponding seismograms.

References

1. Gadylshin, K., Lisitsa, V., Gadylshina, K., Vishnevsky, D.: Optimization of the training dataset for numerical dispersion mitigation neural network. In: Gervasi, O., Murgante, B., Misra, S., Rocha, A.M.A.C., Garau, C. (eds.) Computational Science and Its Applications - ICCSA 2022 Workshops. ICCSA 2022. Lecture Notes in Computer Science, vol. 13378, pp. 295–309. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-10562-3_22
2. Gadylshin, K., Lisitsa, V., Gadylshina, K., Vishnevsky, D., Novikov, M.: Machine learning-based numerical dispersion mitigation in seismic modelling. In: Gervasi, O., et al. (eds.) ICCSA 2021. LNCS, vol. 12949, pp. 34–47. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86653-2_3
3. Gadylshin, K., Vishnevsky, D., Gadylshina, K., Lisitsa, V.: Numerical dispersion mitigation neural network for seismic modeling. Geophysics **87**(3), T237–T249 (2022)
4. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks (2016). arXiv: https://doi.org/10.48550/arXiv.1611.07004
5. Kaser, M., Dumbser, M., Puente, J.d.L., Igel, H.: An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes III. viscoelastic attenuation. Geophys. J. Int. **168**(1), 224–242 (2007). https://doi.org/10.1111/j.1365-246X.2006.03193.x
6. Kaur, H., Fomel, S., Pham, N.: Overcoming numerical dispersion of finite-difference wave extrapolation using deep learning. In: SEG Technical Program Expanded Abstracts, pp. 2318–2322 (2019). https://doi.org/10.1190/segam2019-3207486.1
7. Li, H., Yang, W., Yong, X.: Deep learning for ground-roll noise attenuation. In: SEG Technical Program Expanded Abstracts, pp. 1981–1985 (2018). https://doi.org/10.1190/segam2018-2981295.1
8. Lisitsa, V.: Dispersion analysis of discontinuous Galerkin method on triangular mesh for elastic wave equation. Appl. Math. Model. **40**, 5077–5095 (2016). https://doi.org/10.1016/j.apm.2015.12.039
9. Liu, Y.: Optimal staggered-grid finite-difference schemes based on least-squares for wave equation modelling. Geophys. J. Int. **197**(2), 1033–1047 (2014)
10. Mirza, M., Osindero, S.: Conditional generative adversarial nets. CoRR (2014). https://arxiv.org/abs/1411.1784
11. Mittet, R.: Second-order time integration of the wave equation with dispersion correction procedures. Geophysics **84**(4), T221–T235 (2019)
12. Oliveira, D.A.B., Ferreira, R.S., Silva, R., Vital Brazil, E.: Interpolating seismic data with conditional generative adversarial networks. IEEE Geosci. Remote Sens. Lett. **15**(12), 1952–1956 (2018). https://doi.org/10.1109/LGRS.2018.2866199
13. Palubinskas, G.: Image similarity/distance measures: what is really behind MSE and SSIM? Int. J. Image Data Fusion **8**, 32–53 (2017)
14. Pleshkevich, A., Vishnevskiy, D., Lisitsa, V.: Sixth-order accurate pseudo-spectral method for solving one-way wave equation. Appl. Math. Comput. **359**, 34–51 (2019)
15. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

16. Virieux, J., Calandra, H., Plessix, R.E.: A review of the spectral, pseudo-spectral, finite-difference and finite-element modelling techniques for geophysical imaging. Geophys. Prospect. **59**(5), 794–813 (2011). https://doi.org/10.1111/j.1365-2478.2011.00967.x
17. Yu, S., Ma, J., Wang, W.: Deep learning for denoising (2019). arXiv: https://arxiv.org/abs/1810.11614