The improved DnCNN for linear noise attenuation

Yue Zheng, Yijun Yuan, Xu Si*
School of Geophysics and Information Technology, China University of Geosciences, Beijing, China

Summary

Seismic data are often highly corrupted by different kinds of noise, including linear noise. Therefore, the attenuation of linear noise has been an essential step in seismic data processing. Traditional methods of linear noise suppression are mostly based on the difference of signals and noise in transform domains. However, the application of these traditional methods is limited to some particular assumptions. For this reason, we utilize an algorithm based on deep convolutional neural network (DnCNN) to attenuate linear noise. DnCNN is proposed to suppress Gaussian noise in images. In term of the characteristics of linear noise, we make some improvements to the original DnCNN, like patch size, convolutional kernel number. Tests on two types of synthetic data both indicate that the improved DnCNN algorithm is capable of linear noise attenuation in the seismic data.

Introduction

With the development of seismic exploration, the requirement for signal-to-noise ratio (SNR) of seismic data is getting higher and higher. However, in the process of the seismic data acquisition, geophones often receive a lot of noise. Linear noise is one of the common types of noise in seismic data. Thus, Linear noise attenuation plays an important role in seismic data process. In order to suppress linear noise, a variety of techniques have been developed over the past years. Traditional methods of linear noise suppression are mostly based on the difference of signals and noise in frequency, wave number or other transform domains. Such as *f-x* deconvolution (Abma and Claerboutet, 1995), Karhunen-Loève transform (Yedlin et al., 1987), *f-k* filtering (Yilmaz, 2001), Radon transform (Claerbout and Johnson, 1971), mathematical morphological filtering (MMF) (Huang et al., 2017) and so on. These methods can get favourable denoising performance under the ideal conditions. However, if linear noise overlaps with signals in the transform domains, they may generate unsatisfactory results.
The denoising convolutional neural network (DnCNN) is different from the methods above. It is not subject to certain conditions. In recent years, the convolutional neural network algorithm has got good achievements in different fields, including image denoising. Initially, LeNet is considered as the pioneering work in Convolutional Neural Networks (LeCun et al., 1990). Later, Jain and Seung (2009) made use of it to eliminate image noise. Zhang et al. (2010) applied the neural network to seismic denoising field. To improve the accuracy of computation, the deep network is proposed by Xie (Xie et al., 2012). Zhang et al. (2017) introduced DnCNN and obtained good results in suppressing Gaussian noise of images. In the paper, an introduction to the theory of DnCNN is followed by a detailed description of improved part of the method. The operations to attenuate linear noise in seismic data are making use of the network to obtain linear noise from the seismic records and then removing noise. For the neural network, before using it to process seismic data, an essential step is to input a large amount of data to train the network until parameters of the network become optimum. So, we first construct the pairs of noisy seismic data and corresponding linear noise which are called training dataset. When the process of training ends, denoising with the improved DnCNN can begin. The improved DnCNN is tested on synthetic seismic data and Marmousi model data.

Theroy

The model of seismic data with noise can be considered as follows:

$$x = x^c + v\ , \tag{1}$$

Where x^c is clean data, v is noise and x represents raw seismic data. DnCNN algorithm obtains noise directly. Thus, applying DnCNN to suppress noise needs to remove noise from the original data and then yields clean data. This process can be expressed by the following equation:

$$x^c = x - R(x;\theta)\,, \tag{2}$$

where θ denotes some parameters which can express the information of linear noise, and $R(x;\theta)$ is the predicted noise. Equation (2) reflects a mapping between x and x^c . Before this, the neural network is trained to learn features of linear noise. The training process can be divided into forward propagation and backpropagation. The forward propagation is expressed as follows:

$$\mathrm{z}^{[l]} = w^{[l]} \cdot a^{[l-1]} + b^{[l]}\,, \tag{3}$$

where $w^{[l]}$ is the weights of the *l*-th layer, $b^{[l]}$ is the bias of the *l*-th layer, $\mathrm{z}^{[l]}$ is the convolution output of the *l*-th layer, and $a^{[l-1]}$ is the output of the (*l*-1)-th layer. In particular, $a^{[0]}$ represents the input of the first layer, which is the array transformed from seismic data. The output of forward propagation will be compared according to the loss function. The loss function can be described by

$$l(w,b,\gamma,\beta)=\frac{1}{2N}\sum_{i=1}^{N}\left\|R(x_i;w,b,\gamma,\beta)-(x_i-x_i^{c})\right\|_F^2 , \tag{4}$$

where $R(x)$ is considered as a function to express the whole system, γ and β are two parameters that need to be updated in backpropagation with $w^{[l]}$ and $b^{[l]}$ mentioned in equation (3). During backpropagation, we use the gradient descent algorithm to update the parameters until the loss function reaches the minimum. Hence, the network obtains the best parameters to capture features of linear noise.

The neural network in DnCNN is set to 17 layers. Figure 1 shows the architecture and layers of neural network. The layers of the neural network are divided into three categories. (a) Conv+ReLU: for the first layer, 64 convolutional kernels (Conv) of size 3×3×1 are used to create 64 features maps and rectified linear units (ReLU) become the whole system non-linear. (b) Conv+BN+ReLU: for layers 2 ~ 16, 64 kernels of size 3×3×64 are utilized and Batch Normalization (BN) is added. (c) Conv: for the last layer, a kernel of size 3×3×64 are used to reconstruct the output.

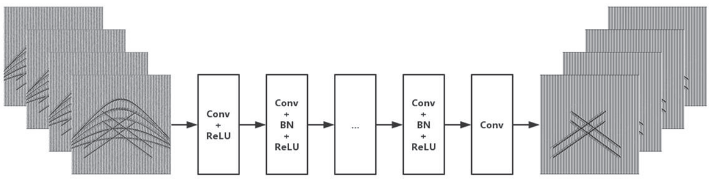

Figure 1 *The architecture and layers of the DnCNN network.*

Because DnCNN is originally designed to attenuate Gaussian noise of images, the performance of suppressing linear noise in seismic data with direct application of DnCNN is not very good. So, we change some parameters of the algorithm like patch size and convolutional kernel number to fit the properties of linear noise. Before the images are input to the network, they usually need to be cut into some small size, which can speed up the training process, the same as seismic data. Patch -based image denoising technique sets size of patch as about 40×40 (Burger et al., 2012). However, high noise level usually needs larger effective patch size to capture more information to restoration (Levin et al., 2011). Because the amplitude of linear noise is larger than that of the effective signals, we set the patch size as 80×80. Meanwhile, we cut more patches from the section of linear noise, which ensures that those patches including linear noise are as much as possible. The number of convolutional kernels can affect the extraction of features as well. More convolution kernels can extract more features but slow the process of convolution. To balance the denoising performance and the training efficiency, we adjust the number of convolutional kernels of the first layer to be 128. Correspondingly, the size of convolutional kernels of the remaining layers is set to be 128×3×3×128, and the last convolutional layer used a kernel of size 3 × 3 × 128.

Example

In the first example, we use synthetic data simulated by the Ricker wavelet to test the feasibility and effectiveness of the improved DnCNN. Figure 2a shows one of the synthetic signals. The number of seismic traces is 110, the record length is 2044 ms, and the sampling interval is 4 ms. As shown in figure 2b, linear noise with high amplitude is added to the signals to make the synthetic data closer to the field data, and random noise is set as background noise. It is apparent that linear noise seriously interferes with the signals. There are 100 pairs of synthetic seismic shot records like Figure 2b and corresponding linear noise in total in the training set. Figure 2c shows the denoising results of the improved DnCNN. Note that there is almost no obvious linear noise in Figure 2c. Figure 2d displays linear noise obtained by the improved DnCNN. We can barely find the signals in it. To demonstrate the feasibility of the improved DnCNN algorithm, we also use original DnCNN and *f-k* filtering to suppress linear noise in Figure 2b. Comparisons between the improved DnCNN algorithm and other methods on synthetic data are presented. Figure 2e and Figure 2f presents the denoised result and removed noise after using the original DnCNN, respectively. It can be observed that the original DnCNN suppresses almost all of linear noise energy and causes no damage to the signals. Figure 2g shows the denoised results processed with *f-k* filtering. It is clear that much residual noise still exists in Figure3e. Figure 2h is the difference between Figure 2b and Figure 2g. Note that some hyperbolic events are also filtered out conspicuously by *f-k* filtering as well. Hence, in this example, the improved DnCNN and the original DnCNN both obtain satisfying denoising results.

In the second example, we report a test on the Marmousi model data. Figure 3a shows one of the Marmousi model data with 96 traces, 725 time samples, 4 ms time interval. As shown in Figure 3b, linear noise with high amplitude is added to the Marmousi model data. Figure 3c, Figure 3d shows the denoising result and the removed linear noise by the improved DnCNN, respectively. It is obvious that the signals and linear noise are separated well with the improved DnCNN. The energy of linear noise is extracted almost completely from the noisy data. In this section, we use the original DnCNN and *f-k* filtering to process the same noisy data to make a comparison as well. Figure 3e shows the denoising performance of the original DnCNN. There is still residual linear noise in the red circle. Figure 3f shows that the removed noise by the original DnCNN. Figure 3g displays the denoised output with *f-k* filtering. Part of linear noise is suppressed while there is obviously residual noise existing in the denoised result. Figure 3h demonstrates the removed linear noise. We can find some energy of the signals, which indicates that *f-k* filtering causes damage to the signals. It is clear that the improved DnCNN still achieves an outstanding result in a complicated noise pattern while the original DnCNN and *f-k* filtering cannot provide good estimations of linear noise.

Conclusions

We applied an improved DnCNN algorithm to attenuate linear noise in noisy seismic data. Some improvements based on original DnCNN are proposed, like patch size, convolutional kernel number. The neural network learns features of linear noise through training and then obtains the ability of identify linear noise from seismic data. During the training process, we need adjust the hyperparameters in order to get appropriate parameters of the network, which can make denoising results more satisfying. Comparing with the original DnCNN and *f-k* filtering in removing linear noise of synthetic data, the improved DnCNN achieved better results. If higher quality and larger amount of dataset is used in training process, the improved DnCNN can obtain greater denoising outcome.

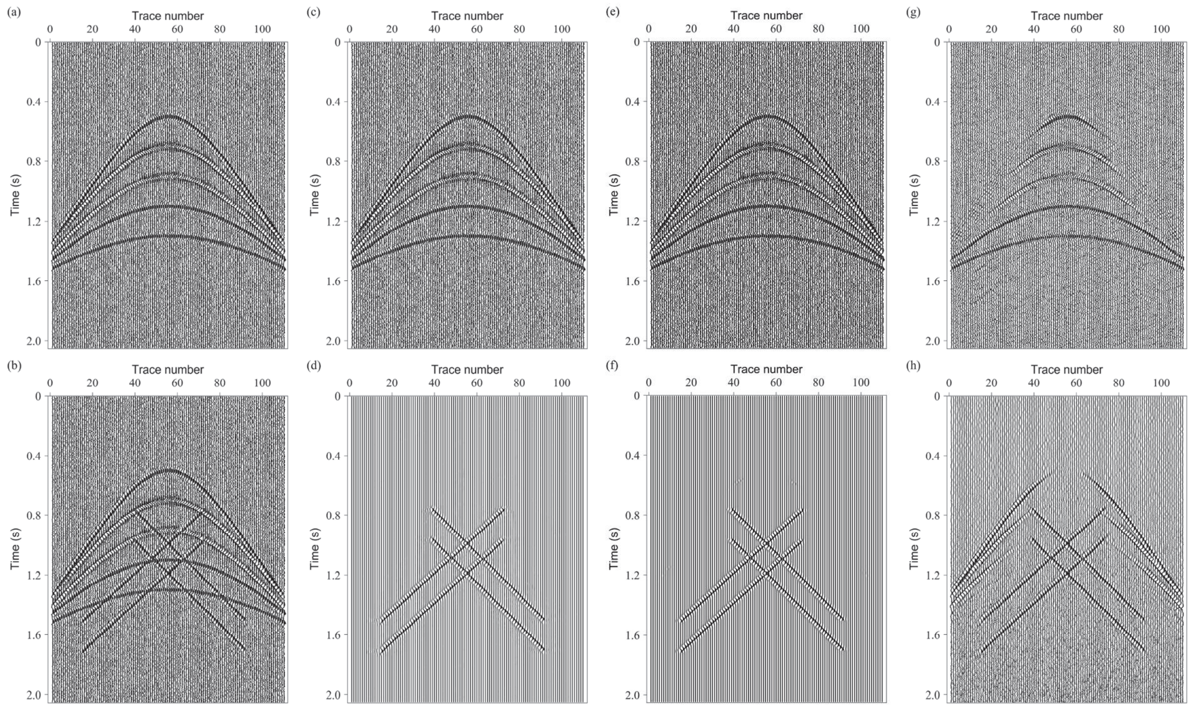

Figure 2 *Synthetic data. (a)Raw data (b)Synthetic data with linear noise (c)Denoising with Improved DnCNN (d)Residual image from Improved DnCNN (e)Denoising with Original DnCNN (d)Residual image from Original DnCNN (g)Denoising with f-k filtering (h)The difference obtained by (b)-(g).*

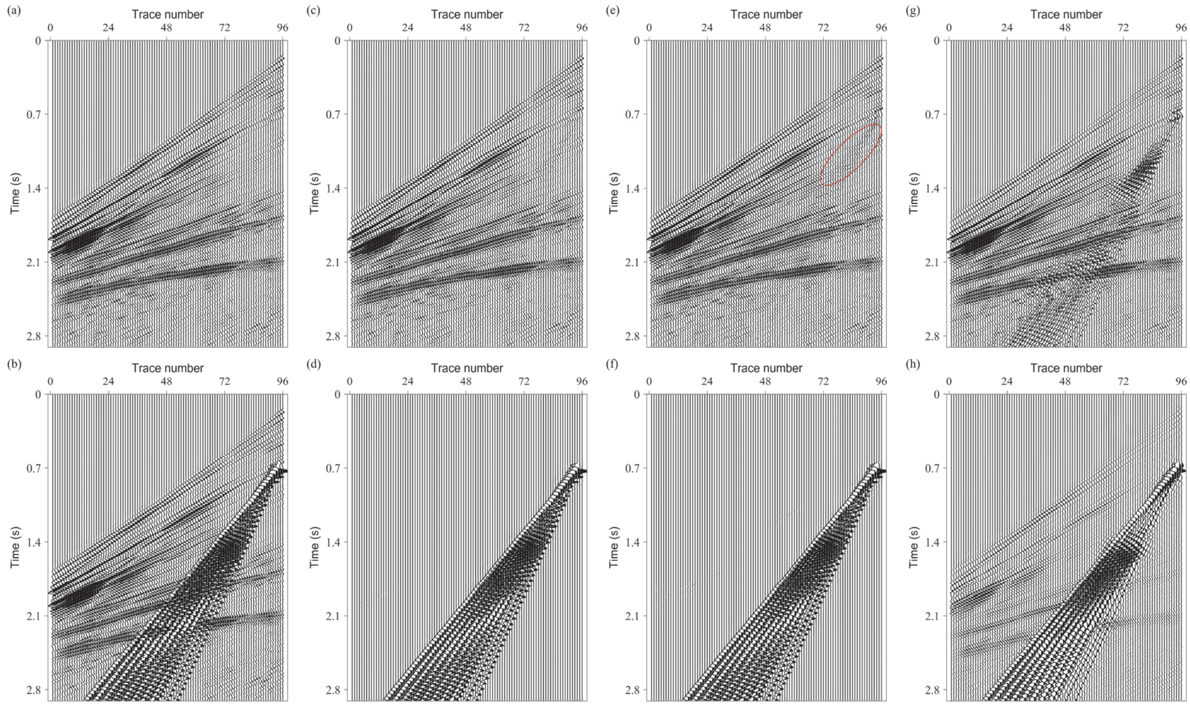

Figure 3 *Marmousi model. (a)Raw data (b) Marmousi data with linear noise (c)Denoising with Improved DnCNN (d)Residual image from Improved DnCNN (e)Denoising with Original DnCNN (d)Residual image from Original DnCNN (g)Denoising with f-k filtering (h)The difference obtained by (b)-(g).*

References

Abma, R., and J. Claerbout, 1995, Lateral prediction for noise attenuation by t-x and f-x techniques: Geophysics, 60, 1887-1896.

Burger, H. C., C. J. Schuler, S. Harmeling, 2012, Image denoising: Can plain neural networks compete with BM3D: The IEEE Conference on Computer Vision and Pattern Recognition, 2392-2399.

Claerbout, J. F., and G. Johnson, 1971, Extrapolation of time-dependent waveforms along their path of propagation: Geophysical Journal International, 26(1), 285-293.

Huang, W., R. Wang, D. Zhang, Y. Zhou, W. Yang, and Y. Chen, 2017, Mathematical morphological filtering for linear noise attenuation of seismic data: Geophysics, 82 (6), 369-384.

Jain, V., and S. Seung, 2009, Natural image denoising with convolutional networks: Advances in neural information processing systems, 769-776.

LeCun, Y., B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, 1990, Handwritten digit recognition with a back-propagation network: Advances in neural information processing systems, 2, 396-404.

Levin, A., and B. Nadler, 2011, Natural image denoising: Optimality and inherent hounds: The IEEE Conference on Computer Vision and Pattern Recognition, 2833-2840.

Xie, J., L. Xu, and E. Chen, 2012, Image denoising and inpainting with deep neural networks: Advances in neural information processing systems, 1, 347-349.

Yedlin, M., I. Jones, and B. Narod, 1987, Application of the Karhunen-Loève transform to diffraction separation: IEEE Transactions on Acoustics, Speech, and Signal Processing, 35, 2-8.

Yilmaz, O., 2001, Seismic data analysis: Processing, inversion, and interpretation of seismic data: Society of Exploration Geophysicists.

Zhang, Y., X. Tian, X. Deng, and Y. Cao, 2010, Seismic denoising based on modified BP neural network: 6th International Conference on Natural Computation, 4, 1825-1829.

Zhang, K., W. Zuo, Y. Chen, D. Meng, and L. Zhang, 2017, Beyond a gaussian denoiser: residual learning of deep CNN for image denoising: IEEE transactions on Image Processing, 26, 3142-3155.