Monte Carlo data-driven tight frame for seismic data recovery

Shiwei Yu[1], *Jianwei Ma*[2] *and Stanley Osher*[3]

[1]Department of Mathematics, Harbin Institute of Technology, Harbin, China
[2]Department of Mathematics, Harbin Institute of Technology, Harbin, China
[3]Department of Mathematics, University of California, Los Angeles, CA 90095, United States

ABSTRACT

Data-driven tight frame (DDTF) has recently been introduced for seismic data denoising and interpolation, and its adaptability to seismic data allows the method to achieve good recovery quality. Tight frame constraint makes DDTF much more efficient for 2D seismic data recovery than traditional dictionary learning methods, but still results in high computational cost for 3D or 5D seismic data. The motivation behind this work is to accelerate the filter bank training process of DDTF while maintaining recovery quality. The most common method for this process is to only use a randomly selected subset of the training set. However, this random selection method uses no prior information of the data. Instead, we have designed a new patch selection method for DDTF seismic data recovery that assumes patches with higher variance contain more information related to complex structures, and should be selected into the training set with a higher probability. First, we calculate the variance of all available patches. Then for each patch, a Gaussian distributed random number is generated and the patch is preserved if its variance is greater than the random number. Finally, all selected patches are used for the filter bank training. We refer to the above process as Monte Carlo DDTF (MC-DDTF), and tested the trained filter bank derived from this process by conducting seismic data denoising and interpolation. The numerical results of the MC-DDTF method surpasses random or regular patch selection DDTF when the sizes of the training sets are the same. For completeness, interpolation methods based on curvelet transform and BM4D (Block Matching 4D) are carried out as comparison to the proposed method for field data.

INTRODUCTION

In seismic exploration, noise and missing traces are unavoidable. The noise comes from various sources (Liu et al., 2015). Missing traces can emerge due to dead traces, or the restriction of environment or economic. Noisy and incomplete data cause lower resolution for

seismic migration, inversion, and AVA (amplitude versus angle analysis). Thus, denoising and interpolation are essential preprocessing steps in the seismic data processing chain.

Methods to attenuate random noise can generally be classified into prediction error filter methods, rank reduction methods, and transform domain methods. Prediction error filter methods can be carried out in the f-x (Canales, 1984) or t-x (Abma and Claerbout, 1995) domain. Rank reduction methods, such as singular spectrum method (Sacchi, 2009) or Cadzow filtering (Trickett, 2008), first arrange each frequency slice of seismic data into a Hankel matrix, then force the Hankel matrix to a lower rank matrix to attenuate the noise. Transform domain methods, such as discrete cosine transform (Lu and Liu, 2007), curvelet transform (Neelamani et al., 2008), and seislet transform (Fomel and Liu, 2010) have also been proposed for seismic data denoising.

Interpolation algorithms mainly fall into wave equation methods and signal processing methods. The former requires an assumption of underground velocity, and are computationally prohibitive, while signal processing methods are more popular due to their simplicity and efficiency. For example, prediction error filter methods (Spitz, 1991; Naghizadeh and Sacchi, 2007) and rank reduction methods (Trickett et al., 2010; Oropeza and Sacchi, 2011; Kreimer and Sacchi, 2012; Kreimer et al., 2014) assume seismic data is composed of linear events. After applying a Fourier transform along the time axis of the data, each obtained frequency slice gives rise to a regression relationship. This relationship can be accurately solved for low frequencies using a linear regressive method or rank reduction method, while the relationship at higher frequencies can be predicted from the low frequency relationship in conjunction with the assumption of linear events.

Sparse inversion methods have recently attracted the attention of researchers due to the development of the L_1 constraint problem. Among sparse transforms, the Fourier transform is the most popular (Spitz, 1991; Zwartjes and Sacchi, 2007; Liu and Sacchi, 2004; Duijndam et al., 1999; Trad, 2009; Xu et al., 2005). Multi-scale and multi-direction transforms, such as curvelet (Naghizadeh and Sacchi, 2010) and shearlet (Hauser and Ma, 2012), are also good

solutions due to their ability to sparsely represent seismic data, while the seislet transform (Fomel and Liu, 2010) and the physical wavelet (Zhang and Ulrych, 2003) are specially designed wavelets for seismic data. However, the sparse inversion methods mentioned above can only provide expansion of seismic data on a predefined basis, which is not adaptive to seismic data and may lead to low denoising or interpolation quality for data with complex structures. Instead, adaptive dictionary learning methods have been proposed for image processing (Aharon et al., 2006; Mairal et al., 2009). These methods first decompose the 2D data into overlapped patches (or cubes for 3D data[4]), which are used as samples to train the dictionary. The dictionary is trained in an optimized sparse representation manner, then used for image processing. Dictionary learning methods achieve outstanding results over fixed basis methods, but require a significant amount of computation time and memory, which limits their application for large scale data processing, i.e., seismic data processing.

Recently, Cai et al. (2014) proposed using a data-driven tight frame (DDTF) method for image denoising. DDTF is different from most existing dictionary learning methods in that it constructs a tight frame of elements rather than an over-complete dictionary. As a result, DDTF is very computationally efficient, while still achieving comparable performance to traditional dictionary learning methods. This improvement makes it promising to apply this adaptive dictionary learning method to seismic data processing. Liang et al. (2014) first introduced DDTF for 2D seismic data interpolation, while Yu et al. (2015) extended the DDTF method to 3D and 5D seismic data for simultaneous interpolation and denoising.

Despite its improvements, DDTF remains computationally prohibitive and impractical for large-scale high-dimensional seismic data processing. The filter training is still computationally and memory intensive because of the huge number of training samples. To improve efficiency, a randomly selected subset of all available training patches can instead be used for training the filter bank (Cai et al., 2014; Liang et al., 2014). A number of research efforts have looked at solving large-scale problems using this 'random' strategy. For example, precisely computing the gradient of the objective function for an objective

[4]we will use the term 'patch' for both 2D and 3D data for consistency in this paper.

function with a large number of components is difficult. Instead, stochastic gradient methods (Bottou, 2010; Xu and Yin, 2015) can estimate this gradient more easily on a single randomly picked component and still provide moderate accuracy. NLM (nonlocal means) is another time-consuming image processing method. To avoid searching the whole image, one naturally performs computations on a preselected subset of the image (Coupe et al., 2006). Offering an alternative, Chan et al. (2014) proposed a Monte Carlo NLM method to eliminate noise in each pexel of an image by using that used a randomly selected set of reference pixels picked according to some sampling pattern.

Inspired by the Monte Carlo NLM, we have designed a new strategy for the selection of the training subset for DDTF. This proposed strategy significantly reduces the amount of required training samples (less than 0.5% of all available patches), and reduces the filtering training time from about 500 seconds to 10 seconds for 3D seismic data containing $128 \times 128 \times 128$ data points. At the same time, we achieve better denoising and interpolation results (1-2 dB higher peak signal-to-noise ratio, PSNR) than can be achieved using a randomly or regularly selected training subset.

The rest of this paper is arranged as follows. In the second section, we briefly introduce DDTF theory, then describe our new strategy for the training subset selection, followed by introducing the denoising and interpolation methods for seismic data. A summary of our algorithm is given at the end of this section. In the third section, we show the results from numerical experiments for denoising and simultaneously denoising and interpolation on both simulated and field seismic data. Finally, we conclude the paper and give directions for future work.

THEORY

Data-driven tight frame

Dictionary learning is equivalent to matrix decomposition with constraints, i.e., decomposes the data as a product of a dictionary matrix and a coefficient matrix, with constraints on both matrices. Usually, approximate decomposition is used in the presence of noise. For example, K-SVD imposes a sparse constraint on the coefficient matrix, an energy constraint on the dictionary matrix, and a quadratic fidelity term is used for the approximate decomposition (Aharon et al., 2006). The objective function of K-SVD is

$$\underset{V,W}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{V}-\boldsymbol{W}\boldsymbol{P}\boldsymbol{G}\|_F^2 \text{ s.t. } \forall i, \|v_i\|_0 \leq \boldsymbol{T}_0,\ \forall j, \|w_j\|_2 \leq 1. \tag{1}$$

Here, $\mathbf{G}$, $\mathbf{V}$, and $\mathbf{W}$ are data, coefficient, and dictionary matrices, respectively; $\mathbf{P}$ is the patch transform (its definition can be found in, e.g., Ma (2013), with the adjoint patch transform denoted as $\mathbf{P}^T$); $\|\cdot\|_2$, $\|\cdot\|_F$, and $\|\cdot\|_0$ are the vector norm, Frobenius norm, and sparsity (the number of non-zeros in a vector), respectively; v_i is the ith column of $\mathbf{V}$; w_j is the jth row of $\mathbf{W}$; and $\mathbf{T}_0$ controls the sparsity of v_i. Equation 1 can be solved by alternatively solving for $\mathbf{V}$ and $\mathbf{W}$, i.e., the sparsity coding step and the dictionary updating step. In the dictionary updating step, one SVD decomposition is used to update each w_j, leading to inefficiency.

Similarly to K-SVD, DDTF also imposes a sparse constraint on the coefficient matrix, but instead imposes a 'tight frame' constraint on the dictionary matrix. Thus, the objective function of DDTF is

$$\underset{V,W}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{V}-\boldsymbol{W}\boldsymbol{P}\boldsymbol{G}\|_F^2 + \alpha\|\boldsymbol{V}\|_0 \text{ s.t. } \boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}, \tag{2}$$

where $\mathbf{I}$ is the identity matrix. The constraint $\mathbf{W}^T\mathbf{W}$=$\mathbf{I}$ means that $\mathbf{W}$ is a tight frame.[5] The coefficient α balances the fidelity and regularization terms. Note that equation 2 can

[5]This expression is not strict. Actually $\mathbf{W}$ and $\mathbf{W}^{\mathrm{T}}$ are the analytic and synthetic operators of the tight frame, where $\mathbf{W}$ is obtained by all shifts of an orthogonal filter bank. We call the filter bank $\mathbf{W}$ (we refer to the rows of $\mathbf{W}$ as filters) directly, and find that this nonstrict expression is more simple and straightforward. Readers can refer to Cai et al. (2014) for the strict expression.

also be solved by alternatively solving for $\mathbf{V}$ and $\mathbf{W}$ (the reader can refer to Cai et al. (2014) for details). With the tight frame constraint, $\mathbf{W}$ is updated by one SVD decomposition in the dictionary update step, allowing DDTF to achieve a much higher efficiency than K-SVD.

Monte Carlo patch selection

A patch transform breaks the original data into small patches that are then used for training the filter bank. For 3D data of size n^3, using a patch size of r^3 means the patch set contains $(n-r+1)^3 r^3$ data points. However, an inordinately large number of patches will create a significant memory and computation burden during the filter training progress. For this reason, the K-SVD and DDTF algorithms available online typically only use a randomly selected subset of original patches as the training set. This raises the question, if the number of training patches is bounded, can we obtain a "better" filter bank by selecting the subset of the training patches in some manner other than randomly? Alternatively, can we select a subset with a smaller size than what is needed with random selection for a given recovery quality? The term "better" suggests a filter bank trained from the selected patches produces a higher quality denoising or interpolation result.

In this work, we propose a new selection strategy based on Monte Carlo theory. The basic assumption is that patches containing complex structures should have a higher probability of being used in the training subset in order to keep most of the data detail. For each patch, we naturally use its variance to represent the complexity of its structure. Each pixel in the image corresponds to the variance of one patch, so variances of all patches can form an "image of variance." For the 2D seismic data composed of 512 traces and 512 time samples (where the time sample interval is set to 1) shown in Figure 1(a), the variance image is given in Figure 1(c). In Figure 1(b), the data is contaminated by Gaussian noise with a PSNR equal to 22.11. The PSNR is defined as the ratio between the maximum amplitude of the clean data and the standard deviation of the noise. The variance image

of Figure 1(b) is given in Figure 1(d). The patch size is set to 8 (time)×8 (offset). As we can see, the variance distribution of the noise-free version and the noisy version are almost the same, so no preparation is needed before calculating the variance. The distribution of variances is given in Figure 1(e). From the distribution, we can see that 80% of the variances are of relatively low value. If we select the patch uniformly or randomly, most of the selected patches will contain little information about complex structures. Thus, those patches with a higher variance should be given priority consideration. However, patches with a lower variance should not be neglected either. The proposed selection algorithm is similar to the Monte Carlo method, so we reefer to it as a Monte Carlo patch selection, which is summarized in algorithm 1.

Algorithm 1 Monte Carlo patch selection algorithm

Input: Patches after patch transform on data.

1: Compute the maximum of the variance v_{max}. Here we use non-overlapping patches as an approximation.

2: For patch i, first compute its variance v_i, then generate a random number $r_i \subseteq (0, v_{max})$ with a Gaussian distribution. If $r_i < kv_i$, then keep patch i. Coefficient k controls the percentage of the chosen patches from all patches. Figure 2(a) shows the relationship between k and the percentage of selected patches.

Output: Selected training subset.

Seismic data denoising and interpolation

The obtained filter bank $\mathbf{W}$ is used for denoising the original data $\mathbf{G}$ with a thresholding method. We denote the denoising function as $\mathbf{D}_\lambda(\mathbf{G})$, defined as

$$\boldsymbol{D}_\lambda(\boldsymbol{G}) = \boldsymbol{P}^T\boldsymbol{W}^T\boldsymbol{T}_\lambda(\boldsymbol{W}\boldsymbol{P}\boldsymbol{G}), \tag{3}$$

where λ is a denoising parameter associated with the noise level, and $\mathbf{T}_\lambda$ is the soft shrinkage operator.

MC-DDTF	DDTF based on Monte Carlo patch selection
RD-DDTF	DDTF based on Random patch selection
RG-DDTF	DDTF based on Regular patch selection
FP-DDTF	DDTF based on all available patches (Full Patch)

Table 1: Abbreviations

For seismic interpolation we introduce a POCS-based (projection onto convex sets) method (Abma and Kabir, 2006). The method is described in algorithm 2.

Algorithm 2 POCS Algorithm

Input: $\mathbf{G}^0$: sub-sampled data, $\mathbf{M}$: sampling matrix, λ: denoising parameter, $\mathbf{G}$: initial data (e.g., nearest neighborhood interpolation can be used here).

1: Denoise the data with the denoising equation 3

$$\boldsymbol{G}^* = \boldsymbol{D}_\lambda(\boldsymbol{G}) \tag{4}$$

2: Reinsert original traces

$$\boldsymbol{G}^* = (\boldsymbol{I} - \boldsymbol{M})\boldsymbol{G}^* + \boldsymbol{M}\boldsymbol{G}^0 \tag{5}$$

3: Set $\mathbf{G}$=$\mathbf{G}^*$ and repeat steps 1 and 2 until a given convergence condition is satisfied.

Output: Interpolated data $\mathbf{G}^*$.

Monte Carlo DDTF for seismic data recovery

The abbreviations for the different DDTF algorithms with different patch selection methods are summarized in Table 1. The MC-DDTF based seismic data denoising and interpolation algorithms are described in algorithms 3 and 4, respectively.

In Figure 3, we apply different patch selection methods on a toy 2D model. The model

Algorithm 3 MC-DDTF based seismic data denoising algorithm

Input: Noisy data, denoising parameter.

1: Transform the raw data into patch space, and use the Monte Carlo patch selection method (algorithm 1) to select a subset of the patch space as a training set.

2: Train the filter bank with problem 2.

3: Denoise the data with the trained filter bank using the denoising formula 3.

Output: Denoised data.

Algorithm 4 MC-DDTF based seismic data interpolation algorithm

Input: Sub-sampled data, Sampling matrix.

1: Transform the raw data into patch space, and use the Monte Carlo patch selection method (algorithm 1) to select a subset of the patch space as a training set.

2: Train the filter bank with problem 2.

3: Interpolate the data with the trained filter bank using the POCS method (algorithm 2).

4: Use the interpolated data as raw data, and repeat steps 1–3 until the convergence condition is met.

Output: Interpolated data.

in Figure 3(a) is composed of four linear events, with 128 traces and 128 time samples (the time/space sample interval is set to 1). The noisy version (PSNR=28.05) of the model is illustrated in Figure 3(b). The patch size is 8 (time)×8 (space), with 0.94% of all patches used for filter training in each method (except for FP-DDTF). The patches selected in RD-DDTF, RG-DDTF, MC-DDTF, and FP-DDTF are shown in Figures 3(c)–3(f). We can see that the patches are mostly located on the events with MC-DDTF in Figure 3(e). Note that Figures 3(g)–3(j) are the trained filters from Figures 3(c)–3(f), respectively. Also note that the MC-DDTF method achieves filters that are the most similar to those obtained by FP-DDTF. Figures 3(k)–3(n) are the denoising results of Figures 3(c)–3(f) with PSNR = 35.33, 34.58, 36.14, and 37.48, respectively. From the figures, we can see that FP-DDTF achieves the highest PSNR, followed by MC-DDTF. However, the filter training time of MC-DDTF is only 0.034 s, more than 15 times faster than FP-DDTF (0.53 s).

In Figure 4(a), we test the filter training time of FP-DDTF and MC-DDTF with respect to different sizes of 3D seismic data. Approximately 0.78% of all patches are used in the MC-DDTF method. We can see that MC-DDTF achieves much higher efficiency than FP-DDTF. During our tests, FP-DDTF runs out of memory (4 GB RAM) when the data size is over $80 \times 80 \times 80$. In Figure 4(b), we test the patch selection time of RD-DDTF and MC-DDTF. We can see that MC-DDTF takes twice the amount of time for patch selection as RD-DDTF, i.e., the additional variance calculation only takes the same amount of time as the random patch selection. In total, the calculation time of MC-DDTF is about 1.5 times that of RD-DDTF for the given tested data sizes.

NUMERICAL RESULTS

In this section, we test the suitability of the MC-DDTF method for providing denoising and interpolation on 3D/5D simulated data and field data. We show that MC-DDTF significantly accelerates the filter training progress, with little damage to reconstruction quality. Compared with RD-DDTF and RG-DDTF, we achieve a better reconstruction

quality in PSNR manner. For the field data, we also compare the MC-DDTF method with other state of the art methods: curvelet transform-based interpolation and a BM4D-based (Block Matching 4D) (Maggioni et al., 2013) interpolation method. All tests were carried out on a personal laptop, using $Matlab^{TM}$ on the $Windows\ 7^{TM}$ operating system, with 4 GB RAM and an Intel $CORE^{TM}\ i7$ CPU.

Denoising on shot gather data set 1

A shot gather (Shahidi et al., 2013) is displayed in Figure 5(a), with its noise corrupted version (PSNR=22.11) shown in Figure 5(b). Note that, if not specially mentioned when displaying 3D data, the center slice of each axis is shown. The data is composed of 128 (time)×128 (inline)×128 (crossline) samples. The time sample interval is 4 ms, while the space sample intervals in both the inline and crossline directions are 0.02 km. The patch size is 8 (time)×8 (inline)×8 (crossline) (if not specified, the sample interval and patch size are the same for subsequent tests). Figures 5(c)–5(f) show the denoising results. FP-DDTF achieves the highest PSNR at 38.23, with a filter training time of 427 s. For MC-DDTF, RD-DDTF, and RG-DDTF, we selected 0.32% of all patches as a training subset, which required a filter training time of 9.74 s. MC-DDTF achieving a higher PSNR (37.93) than RD-DDTF (37.56) and RG-DDTF (37.54). The filtering time 96.67 s was the same for all methods. Figures 6(a)–6(d) show the difference between the recovery data and the noisy data corresponding to Figures 5(c)–5(f), respectively. We mention here that, when using FP-DDTF in this test, the program ran out of memory and virtual disk memory had to be used to complete the experiment.

Denoising on migration data set 2

Migration data from the North Sea region, courtesy of Elf Aquitaine (Sergey Fomel, 2003), and its noise corrupted version (PSNR=28.13) are shown in Figure 7(a) and Figure 7(b), respectively. The data size is 512 (time)×512 (inline)×16 (crossline). Note that we did

not test the FP-DDTF method on this data, as the computer used for the experiment has far from sufficient memory. For MC-DDTF, RD-DDTF, and RG-DDTF, 0.072% of all patches were used for training the filter bank, which required only 6.35 s. Filtering time was 182.00 s. The denoising results are shown in Figures 7(c)–7(e), with MC-DDTF, RD-DDTF, and RG-DDTF achieving PSNRs of 9.03, 38.45, and 38.32, respectively. In Figure 8, we give the recovery difference and the magnified version (time: 0-0.512 s, crossline: 3.86-6.40 km, inline: 0.16 km) corresponding to Figures 7(c)–7(e). From Figures 8(d)–8(f), we can see that MC-DDTF maintains the energy of the original data better than RD-DDTF and RG-DDTF.

Figure 9 shows how the size of the training subset affects the filtering result. Figures 9(a) and 9(b) correspond to tests in Figure 5 and Figure 7, respectively. In the figures, the x-axis is the percentage of the training subset of all patches and the y-axis is the PSNR of the filtering result. In Figures 9(a), when the percentage is less than 0.5%, MC-DDTF achieves a higher PSNR than RD-DDTF and RG-DDTF, and the advantage is much greater when the percentage is less than 0.1%. This is because the patches on the seismic event have high variance, and will be selected with higher probability in the Monte Carlo patch selection. In contrast, random and regular patch selection treat the patches the same whether or not they occur on the event, so information in the events may become submerged by too much information selected from other areas. When the percentage is over 0.6%, MC-DDTF, RD-DDTF, and RG-DDTF all achieve almost the same PSNR. Because with the simulated model in Figure 5(a), important information is determined by all three methods when the percentage is over 0.6%. For the data in Figure 7(a), which contains more complex structures, MC-DDTF always achieves a higher PSNR than RD-DDTF and RG-DDTF, with the difference becoming more obvious at lower patch percentages. This is because Monte Carlo patch selection guarantees the selection of more patches containing complex structures than random or regular patch selection.

Interpolation for migration data set 2

In Figure 10, we decimate the noisy data in Figure 7(b) with 50% random traces in the inline-crossline plane. For comparison, the original data is shown in Figure 10(a) and the decimated data is shown in Figure 10(b). Nearest neighborhood interpolation is used to generate the initial training data, with four loops used in algorithm 3 (the same setting is used in the following examples). For the filter training, 0.21% of the patches are used (an average of the 4 loops),. One iteration of the filter training taking 5.08 s and one pass of interpolation taking 1210 s. We achieve a PSNR of 38.96 using MC-DDTF, with the result shown in Figure 10(c). For completeness, we compared the result to the results from using BM4D (PSNR=38.53) interpolation and curvelet transform interpolation (PSNR=37.89), as shown in Figure 10(d) and Figure 10(e). Note that BM4D gives a watercolor effect in the results, while the curvelet transform makes the events over connected. In Figure 11, we give the magnified version (time: 0-0.512 s, crossline: 5.14-7.68 km, inline: 0.16 km) of the results in Figure 10.

Interpolation for migration data set 3

A 3D migration data set (from Bingbing Sun) is displayed in Figure 12(a). The data size is 240 (time)$\times$221 (inline)$\times$271 (crossline), with the $120th$ slice of each direction shown. The 50% sub-sampled version is shown in Figure 12(b). A total of 0.027% of all patches were used in the filter training progress, with filter training taking just 6.30 s for one pass and interpolation taking 6500 s for one pass. The interpolation results are shown in Figures 12(c)–12(e). We achieved a PSNR of 27.74 with MC-DDTF, higher than was achieved using either the curvelet transform (PSNR=24.68) or BM4D (PSNR=27.60).

Interpolation for 5D simulated data

Finally we tested simultaneous denoising and interpolation on a 5D simulated model. The parameters of the 5D model can be found in Yu et al. (2015). Four 2D sections of the 5D

data are shown in Figure 13(a), with its noisy and sub-sampled version in Figure 13(b). For this test, 0.043% of the patches were used for filter bank training, with a 4^5 patch size. The training subset formed a 1024×725 matrix, which required 2.84 MB of memory. This compares to the memory requirement for all patches, where the size of the matrix was 1024× 1685099 and required 6.56 GB of memory. Recovery results using MC-DDTF and RG-DDTF are shown in Figure 13(c) and Figure 13(d), respectively. MC-DDTF achieves a higher PSNR (30.89) than RG-DDTF (30.48). The recovery error (times 3) sections are shown in Figures 13(e) and 13(f), respectively.

CONCLUSION

We introduced a Monte Carlo-based DDTF for high dimensional seismic data denoising and interpolation. This MC-DDTF method accelerates the filtering training progress with less damage to the recovery result than RD-DDTF or RG-DDTF. Future work will focus on paralleling the filtering process and making the adaptive dictionary learning method handle high dimensional seismic data within a reasonable amount of time.

ACKNOWLEDGEMENT

REFERENCES

Abma, R., and J. Claerbout, 1995, Lateral prediction for noise attenuation by t-x and f-x techniques: Geophysics, **60**, 1887–1896, doi:10.1190/1.1443920.

Abma, R., and N. Kabir, 2006, 3D interpolation of irregular data with a POCS algorithm: Geophysics, **71**, no. 6, E91-E97, doi:10.1190/1.2356088.

Aharon, M., M. Elad, and A. Bruckstein, 2006, K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation: IEEE Transcations on Signal Processing, **54**, no. 11, 4311–4322, doi:10.1109/TSP.2006.881199.

Bottou, L., 2010, Large-Scale Machine Learning with Stochastic Gradient Descent: Proc. of COMPSTAT 2010.

Cai, J., H. Ji, Z. Shen, and G. Ye, 2014, Data-driven tight frame construction and image denoising: Applied and Computational Harmonic Analysis, **37**, no. 1, 89–105, doi:10.1016/j.acha.2013.10.001.

Chan, S. H., T. Zickler, and Y. M. Lu, 2014, Monte Carlo non local means: Random sampling for large-scale image filtering: IEEE Trans. Image Process, **23**, no. 8, 3711–3725, doi:3711–3725.10.1109/TIP.2014.2327813.

Canales, L., 1984, Random noise reduction: 54th Annual International Meeting, SEG, Expanded Abstracts, 525–527.

Coupe, P., P. Yger, and C. Barillot, Fast non local means denoising for 3D MR images: Proc. MICCAI, 33–40.

Duijndam, A., M. Schonewille, and C. Hindriks, 1999, Reconstruction of band-limited signals, irregularly sampled along one spatial direction: Geophysics, **64**, no. 2, 524–538, doi:10.1190/1.1444559.

Fomel, S., 2003, Time-migration velocity analysis by velocity continuation: Geophysics, **68**, no. 5, 1662-1672, doi:10.1190/1.1620640.

Fomel, S., and Y. Liu, 2010, Seislet transform and seislet frame: Geophysics, **75**, no. 3, V25-V38, doi:10.1190/1.3380591.

Hauser, S., and J. Ma, 2012, Seismic data reconstruction via shearlet-regularized directional

inpainting, http://www.mathematik.uni-kl.de/uploads/tx_sibibtex/seismic.pdf, accessed 15 May 2012.

Kreimer, N., and M. Sacchi, 2012, A tensor higher-order singular value decomposition for prestack seismic data noise reduction and interpolation: Gephysics, **77**, no. 3, V113–V122, doi:10.1190/geo2011-0399.1.

Kreimer, N., A. Stanton and M. D. Sacchi, 2014, Tensor completion based on nuclear norm minimization for 5D seismic data reconstruction: Geophysics, **78**, no. 6, V273–V284, doi:10.1190/geo2013-0022.1.

Liang, J., J. Ma, and X. Zhang, 2014, Seismic data restoration via data-driven tight frame: Geophysics, **79**, no. 3, 65–74, doi:10.1190/geo2013-0252.1.

Liu, B., and M. D. Sacchi, 2004, Minimum weighted norm interpolation of seismic records: Geophysics, **69**, no. 6, 1560–1568, doi:10.1190/1.1836829.

Liu, Y., N. Liu, and C. Liu, 2015, Adaptive prediction filtering in t-x-y domain for random noise attenuation using regularized nonstationary autoregression: Geophysics, **80**, no. 1, V13–V21, doi:10.1190/geo2014-0011.1.

Lu, W., and J. Liu, 2007, Random noise suppression based on discrete cosine transform: 77th Annual International Meeting, SEG, Expanded Abstracts, 2668–2672.

Ma, J., 2013, Three-dimensional irregular seismic data reconstruction via low-rank matrix completion: Geophysics, **78**, no. 5, 181–192, doi:10.1190/geo2012-0465.1.

Maggioni, M., V. Katkovnik, K. Egiazarian, and A. Foi, 2013, Nonlocal transform-domain filter for volumetric data denoising and reconstruction: IEEE Transactions on Image Processing, **22**, no. 1, 119–133, doi:10.1109/TIP.2012.2210725.

Mairal, J., F. Bach, J. Ponce, and G. Sapiro, 2009, Online dictionary learning for sparse coding: in Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 689–696, doi:10.1145/1553374.1553463.

Neelamani, R., A. I. Baumstein, D. G. Gillard, M. T. Hadidi, and W. I. Soroka, 2008, Coherent and random noise attenuation using the curvelet transform: The Leading Edge, **27**, 240–248, doi:10.1190/1.2840373.

Naghizadeh, M., and K. Innanen, 2011, Seismic data interpolation using a fast generalized Fourier transform: Geophysics, **76**, no. 1, V1–V10, doi:10.1190/1.3511525.

Naghizadeh, M. and M. D. Sacchi, 2007, Multistep autoregressive reconstruction of seismic records: Geophysics **72**, no. 6, V111-V118, doi:10.1190/1.2771685.

Naghizadeh, M., and M. D. Sacchi, 2010, Beyond alias hierarchical scale curvelet transform interpolation of regularly and irregularly sampled seismic data: Geophysics, **75**, no. 6, 189–202, doi:10.1190/1.3509468.

Oropeza, V., and M. Sacchi, 2011, Simultaneous seismic data denoising and reconstruction via multichannel singular spectrum analysis: Geophysics, **76**, no. 3, V25–V32, doi:10.1190/1.3552706.

Sacchi, M. D., 2009, FX singular spectrum analysis: Presented at CSPG CSEG CWLS Convention.

Shahidi R., G. Tang, J. Ma, F. Herrmann, 2013, Application of randomized sampling schemes to curvelet transform-based sparsity-promoting seismic data recovery, Geophysical Prospecting, **61**, no. 5, 973-997, doi:10.1111/1365-2478.12050.

Spitz, S., 1991, Seismic trace interpolation in the fx domain: Geophysics, **56**, no. 6, 785–794, doi:10.1190/1.1443096.

Trad, D., 2009, Five-dimensioinal interpolation: Recovering from acquisition constraints: Geophysics, **74**, no. 6, V123–V132, doi:10.1190/1.3245216.

Trickett, S., 2008, F-xy Cadzow noise suppression: 78th Annual International Meeting, SEG, Expanded Abstracts, 2586–2590.

Trickett, S., L. Burroughs, A. Milton, L. Walton, and R. Dack, 2010, Rank reduction-based trace interpolation: 80th Annual International Meeting, SEG, Expanded Abstracts, 3829–3833.

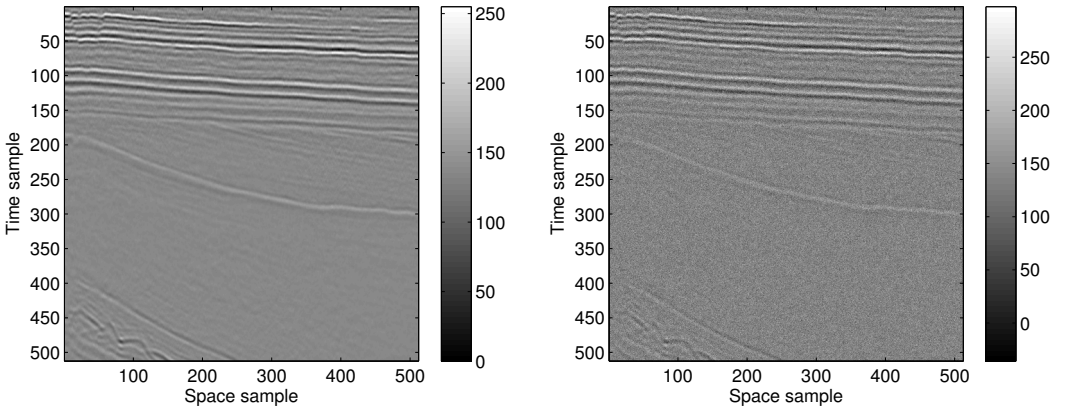
Xu, S., Y. Zhang, D. Pham, and G. Lambaré, 2005, Antileakage Fourier transform for seismic data regularization: Geophysics, **70**, no. 4, V87–V95, doi:10.1190/1.1993713.

Xu Y., W. Yin, Block stochastic gradient iteration for convex and nonconvex optimization: Optimization and Control, arXiv:1408.2597

Yu, S., J. Ma, X. Zhang, M. Sacchi, Denoising and interpolation of high-dimensional seismic data by learning tight frame: Geophysics, 2015, revised.

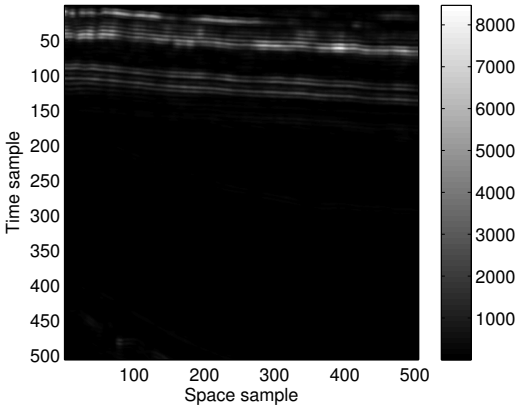
Zhang, R., and T. Ulrych, 2003, Physical wavelet frame denoisng: Geophysics, **68**, no. 1, 225-231, doi:10.1190/1.1543209.

Zwartjes, P., and M. Sacchi, 2007, Fourier reconstruction of nonuniformly sampled, aliased seismic data: Geophysics, **72**, no. 1, V21–V32, doi:10.1190/1.2399442.

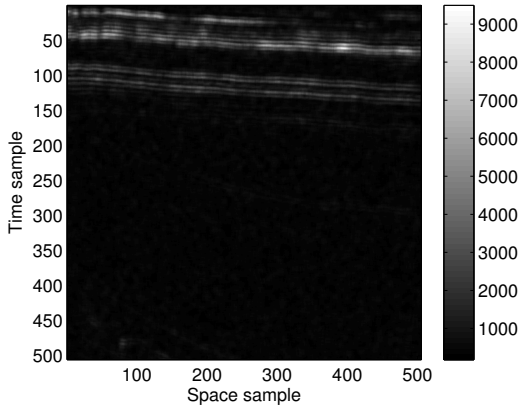


(a)

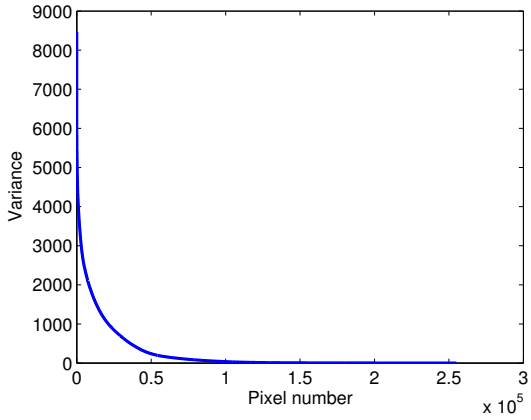
(b)



(c)

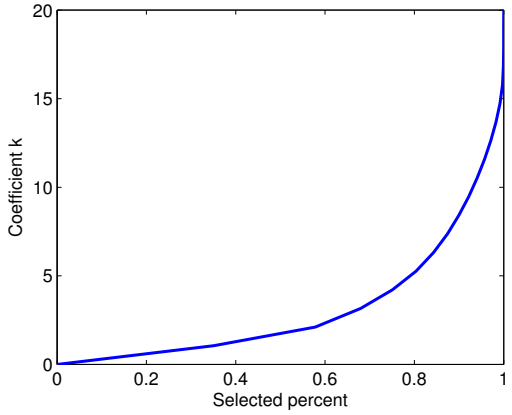


(d)



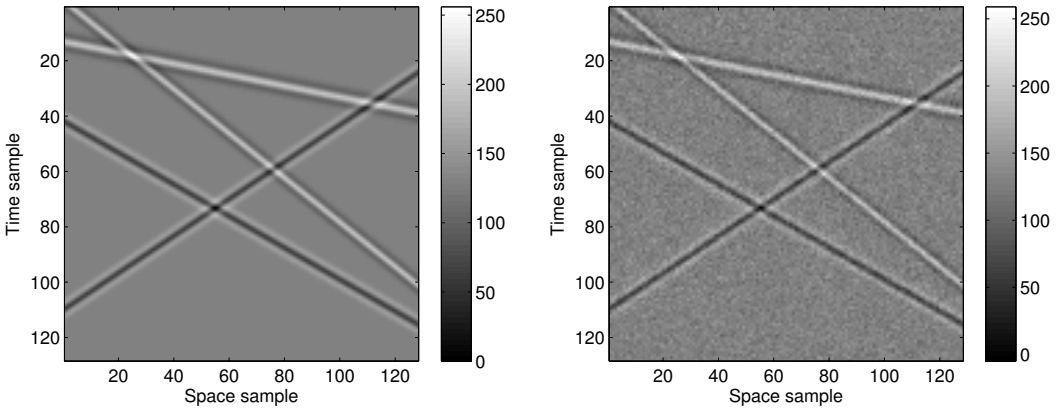
(e)

Figure 1: A demonstration of the variance distribution of seismic data. (a) 2D seismic data. (b) Noisy version of (a). (c)(d) Variance distribution with patch size=8 for (a)(b). (e) Sorted variance distribution.

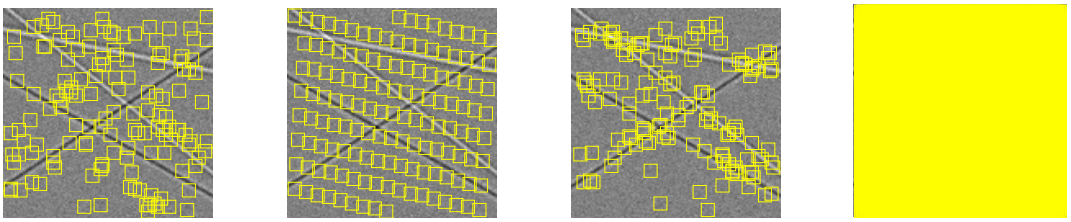


(a)

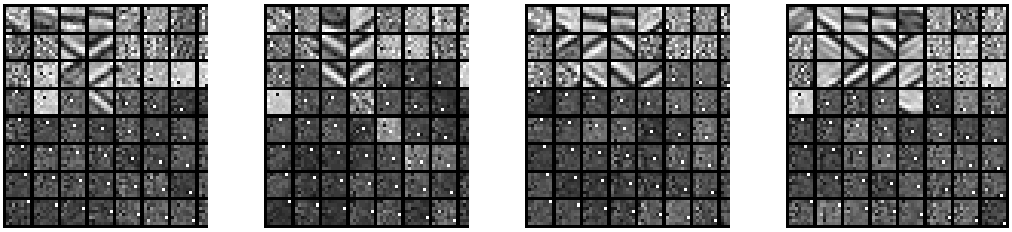
Figure 2: The percentage of selected patches from all patches versus coefficient k.



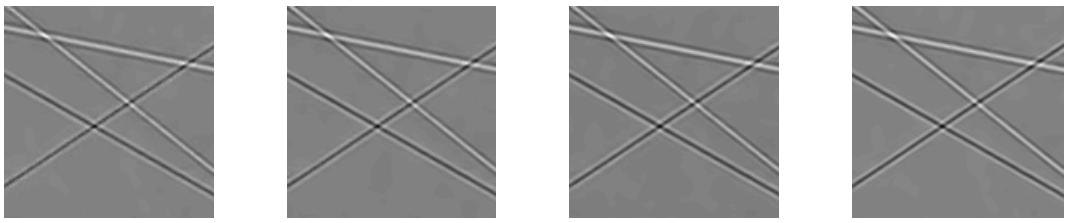
(a) (b)



(c) (d) (e) (f)

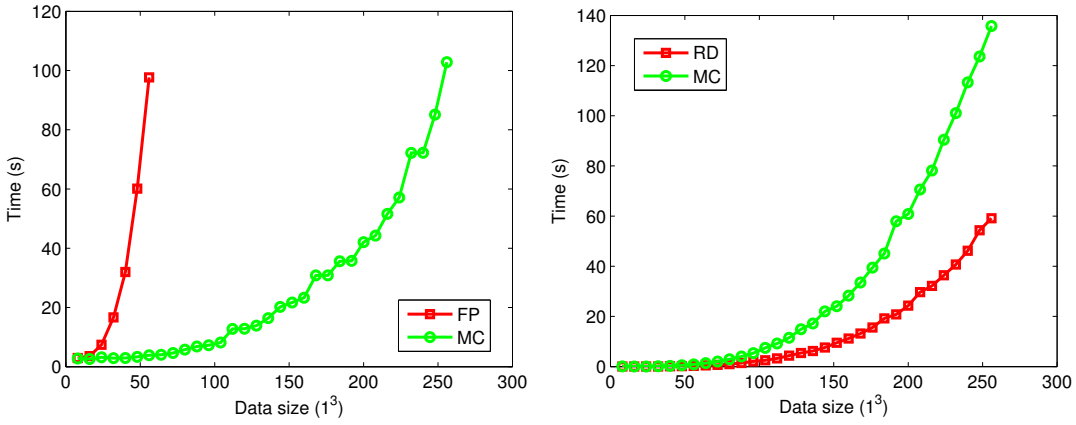


(g) (h) (i) (j)



(k) (l) (m) (n)

Figure 3: A demonstration of different patch selection methods. (a) 2D seismic data. (b) Noisy version of (a). (c)-(f) Patches selected using RD, RG, MC, and FP, respectively. (g)-(j) Trained filters from (c)-(f). (k)-(n) Denoising results of (c)-(f) with PSNR = 35.33, 34.58, 36.14, and 37.48, respectively.



(a) (b)

Figure 4: Time analysis on 3D seismic data. (a) Filter training time for FP and MC patch selection. (b) RD and MC Patch selection time.

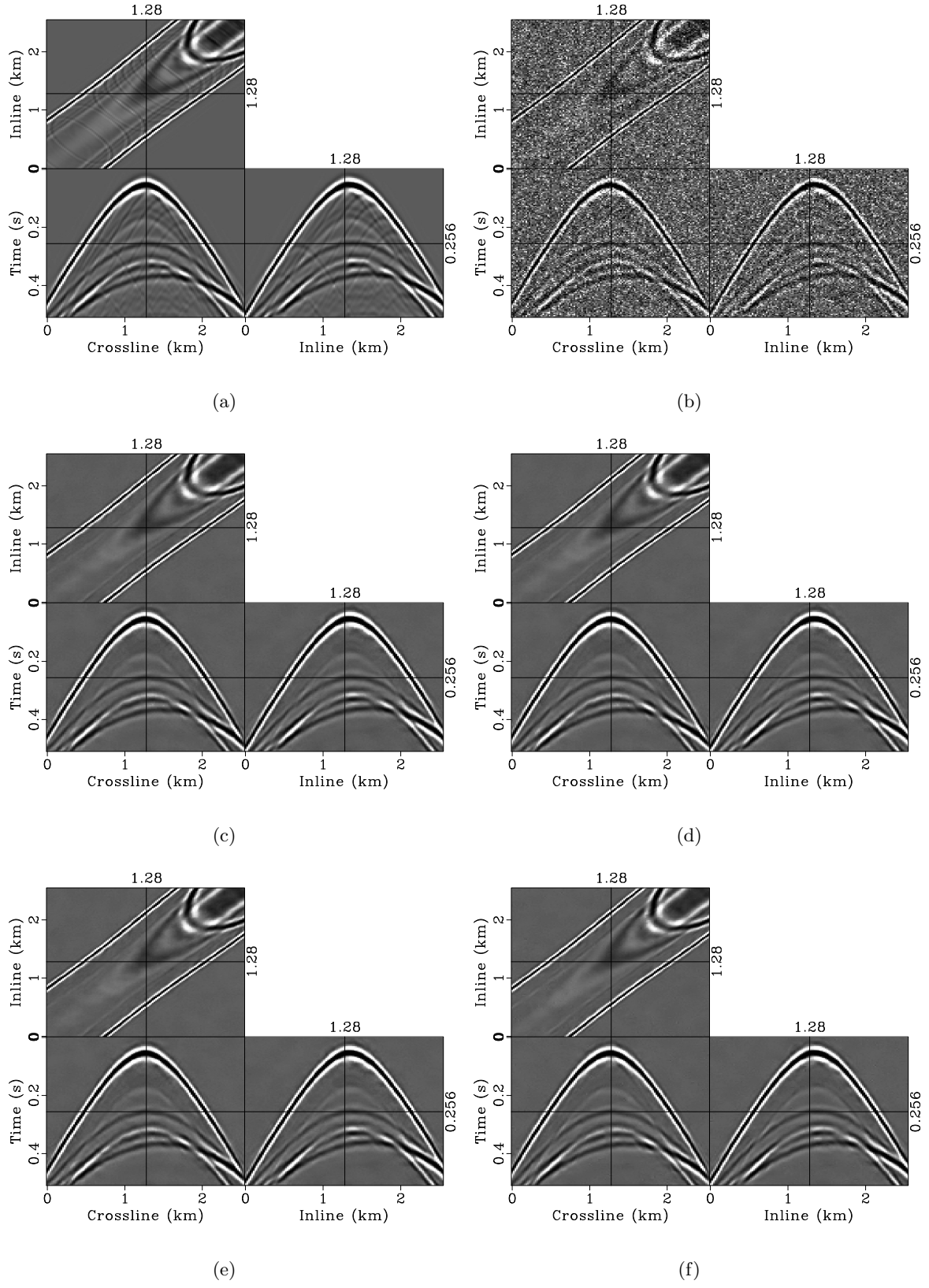
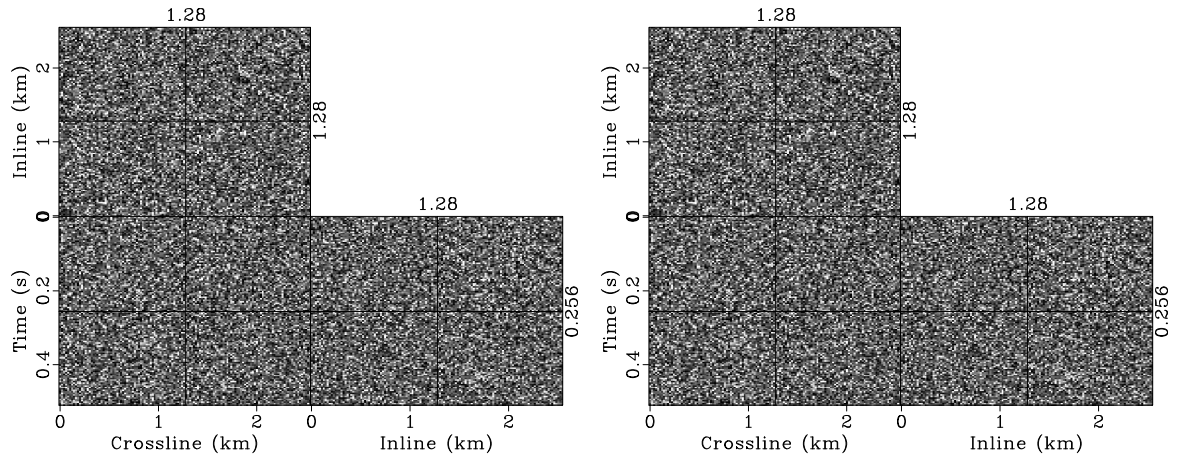
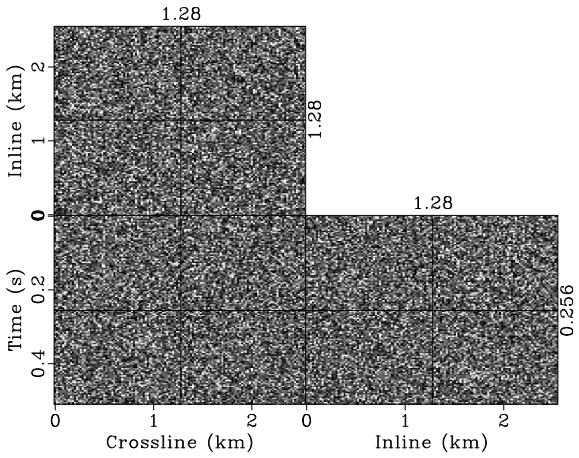


Figure 5: Denoising result for simulated data. (a) Clean data. (b) Noisy data (PSNR=22.11). (c) Recovery with RD-DDTF (PSNR=37.56). (d) Recovery with RG-DDTF (PSNR=37.54). (e) Recovery with MC-DDTF (PSNR=37.93). (f) Recovery with FP-DDTF (PSNR=38.23).

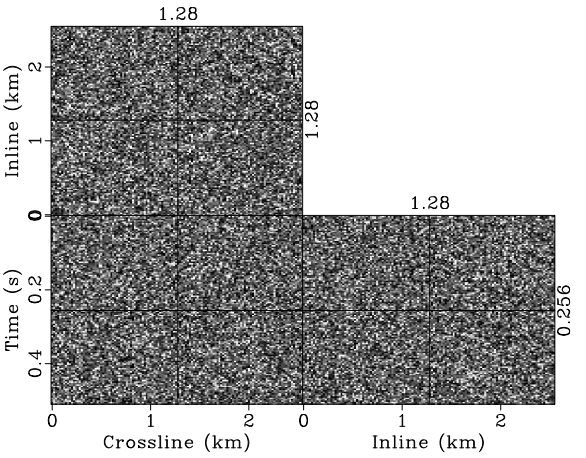


(a)

(b)



(c)



(d)

Figure 6: Recovery residual for simulated data. (a)-(d) correspond to Figures 5(c)–5(f), respectively.

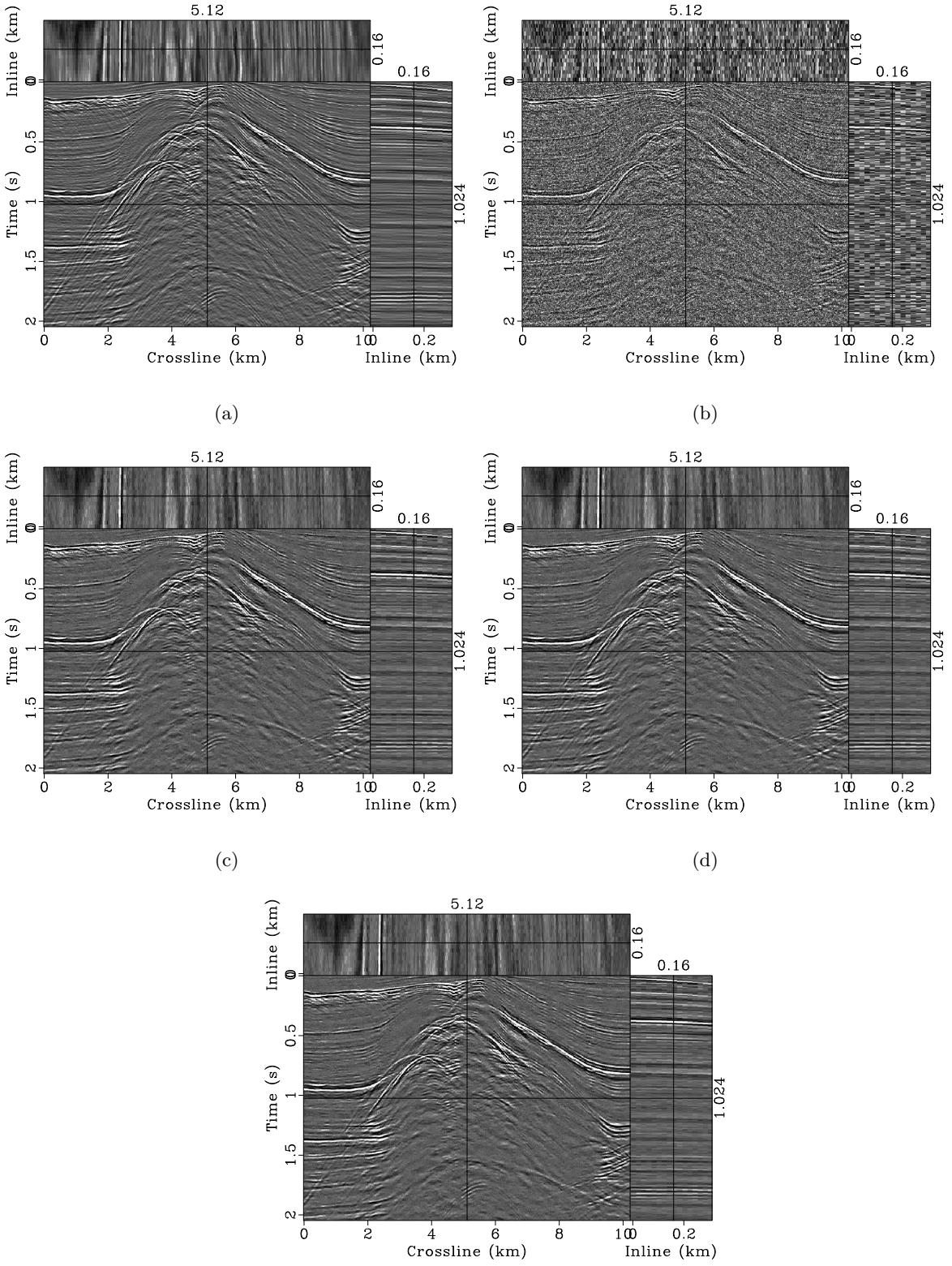


Figure 7: Denoising result for field data. (a) Clean data. (b) Noisy data (PSNR=28.13). (c) Recovery with RD-DDTF (PSNR=38.45). (d) Recovery with RG-DDTF (PSNR=38.22). (e) Recovery with MC-DDTF (PSNR=39.03).

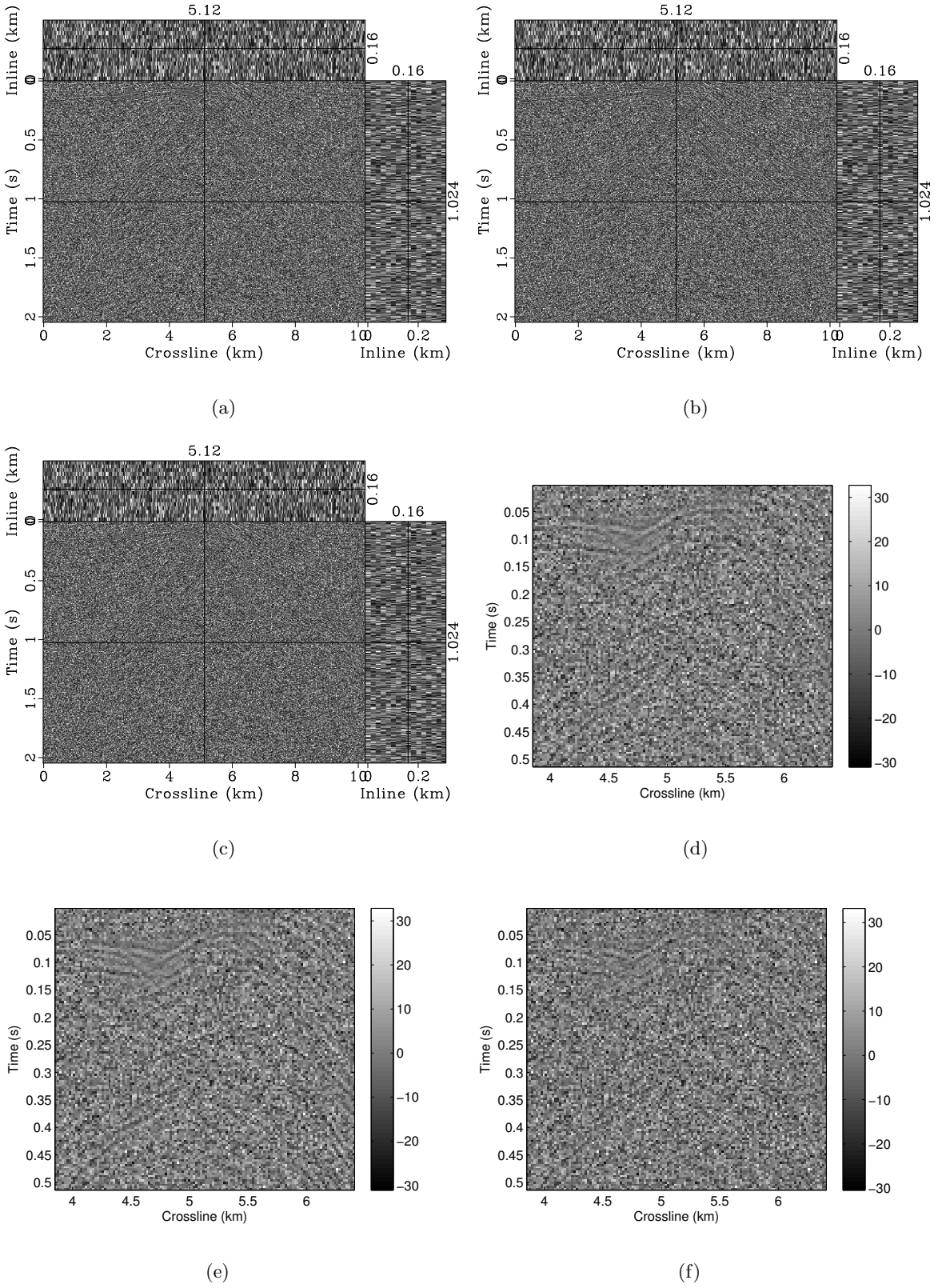
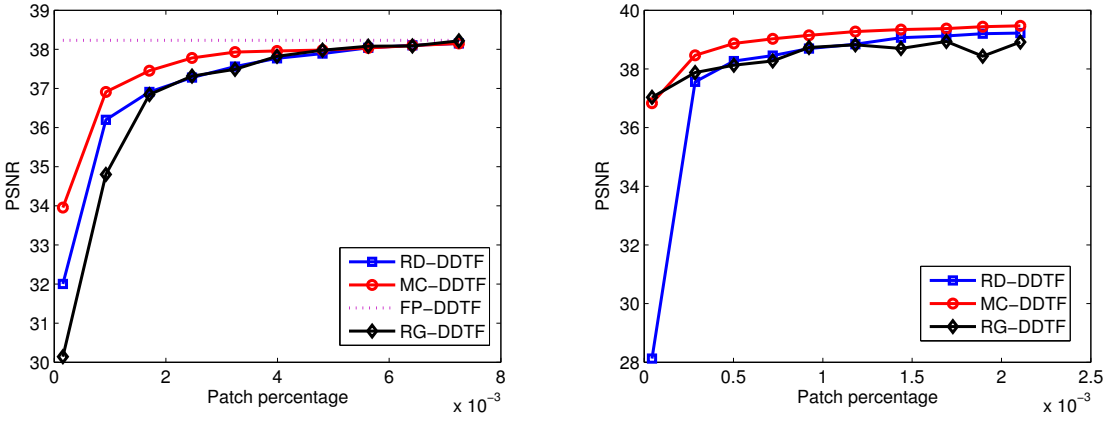
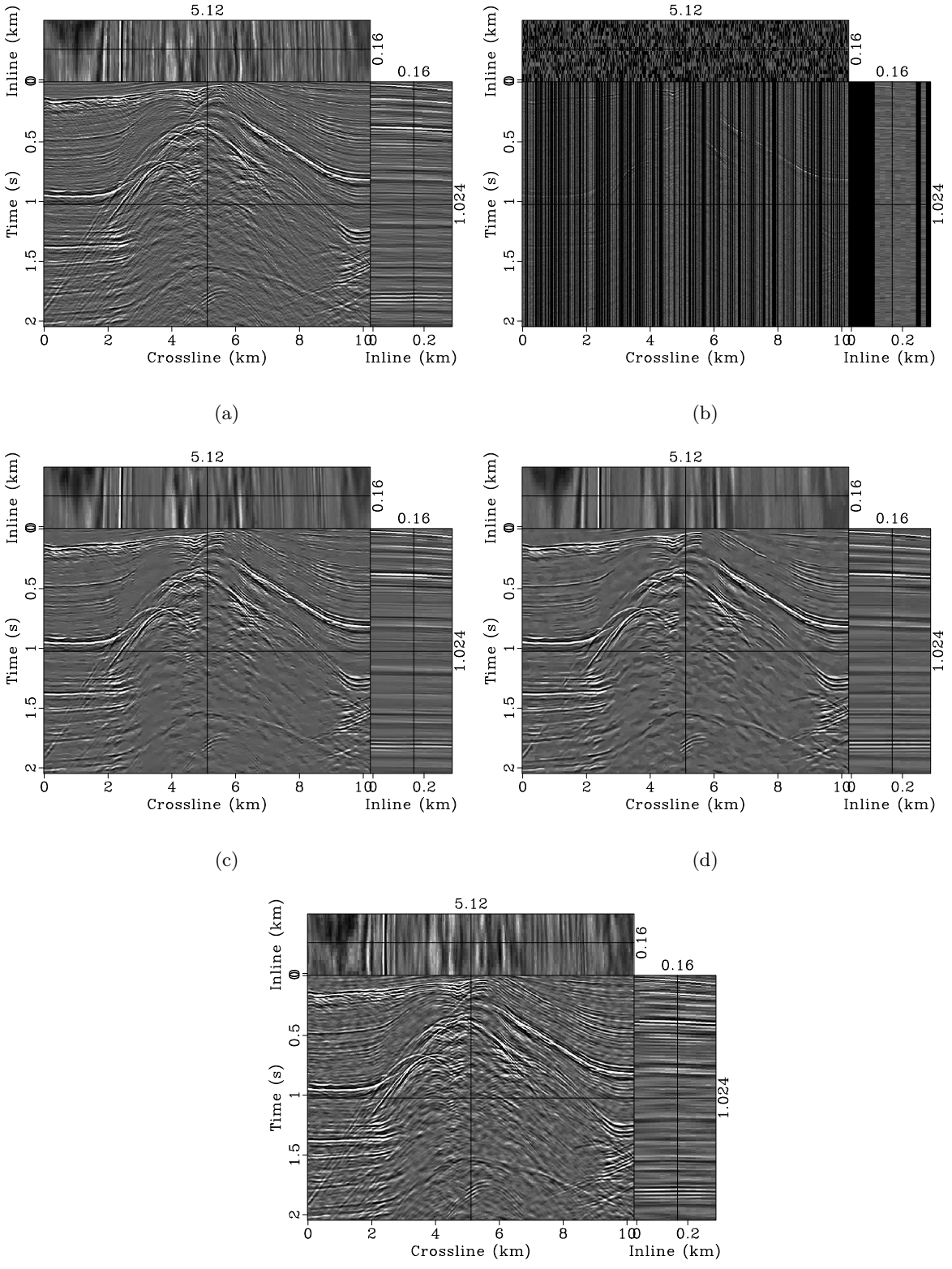


Figure 8: Denoising result for 3D data. (a)-(c) The difference between denoising data and noisy data corresponding to Figures 7(c)–7(e), respectively. (d)-(f) are the magnified version of (a)-(c), respectively.



(a) (b)

Figure 9: PSNR versus patch percentage. (a)(b) correspond to tests in Figure 5 and Figure 7, respectively.



(a)

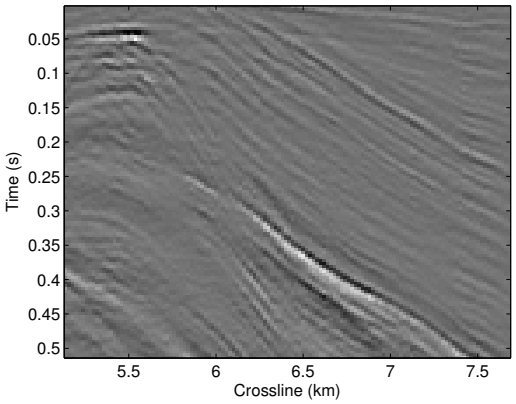
(b)

(c)

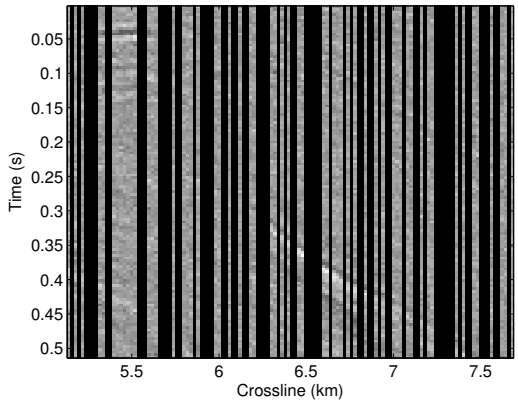
(d)

(e)

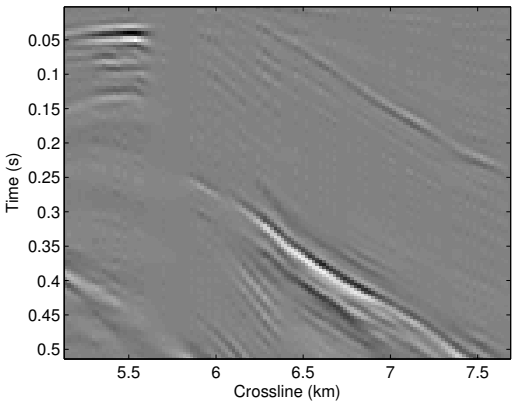
Figure 10: Simultaneous interpolation and denoising for field data in Figure 7(a). (a) Original data. (b) 50% sub-sampled and noisy data. (c) Recovery with MC-DDTF (PSNR=38.96). (d) Recovery with BM4D (PSNR=38.53). (e) Recovery with curvelet transform (PSNR=37.89).



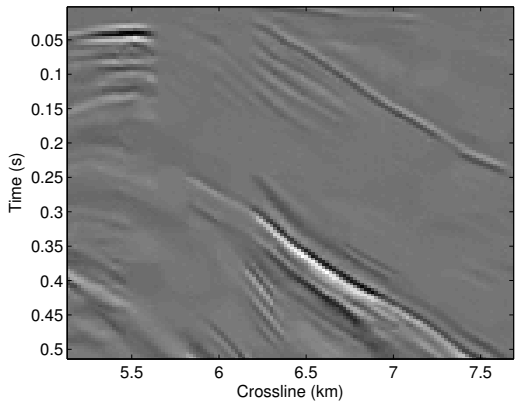
(a)



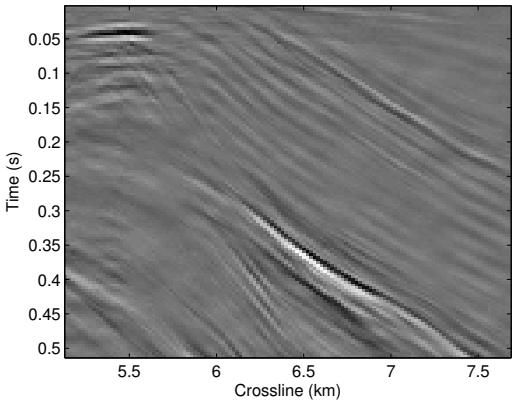
(b)



(c)



(d)



(e)

Figure 11: Magnified version of the recovery results corresponding to results in Figure 10.

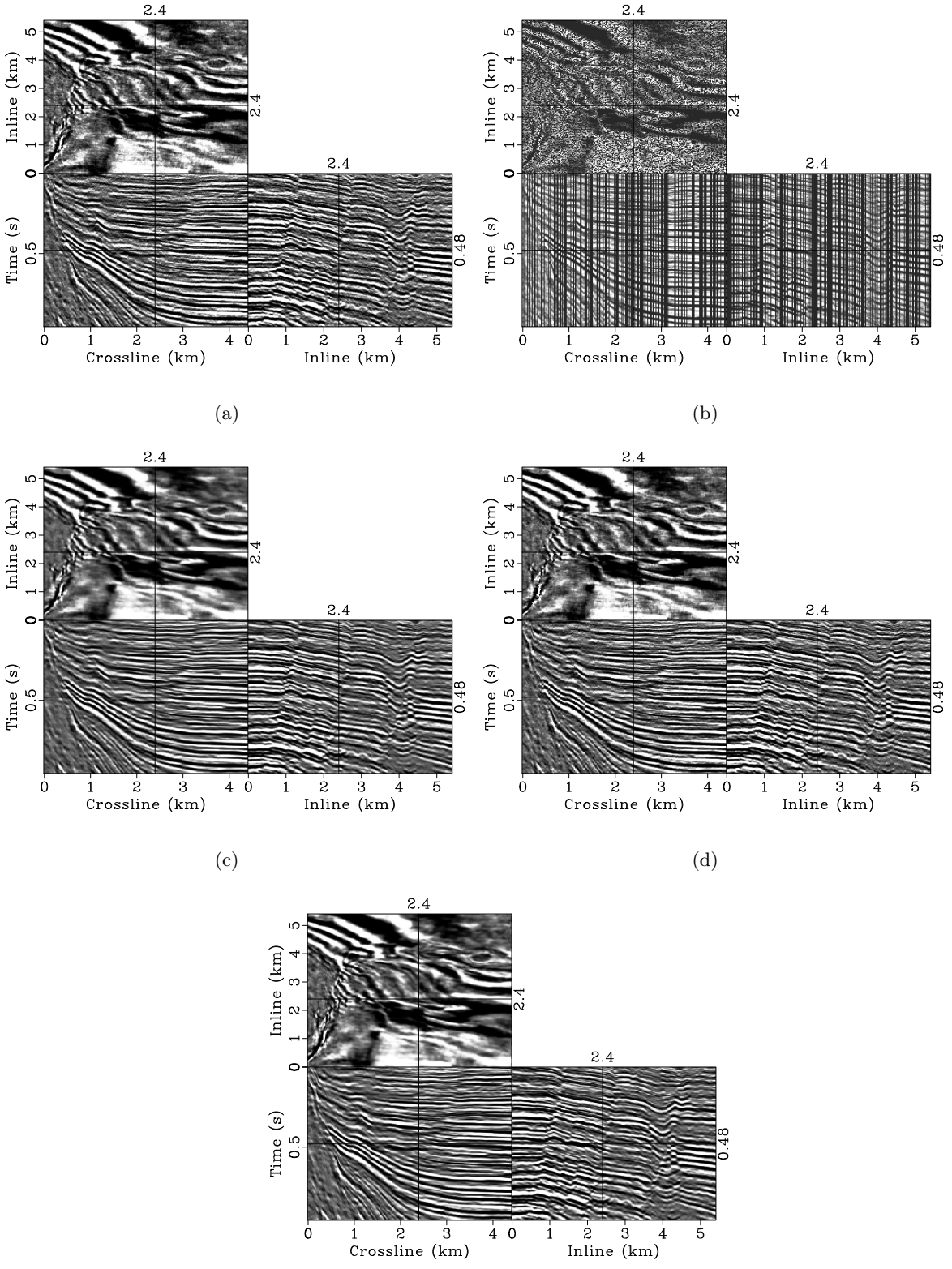


Figure 12: Simultaneous interpolation and denoising for field data. (a) Original data. (b) 50% sub-sampled data. (c) Recovery with MC-DDTF (PSNR=27.74). (d) Recovery with BM4D (PSNR=27.60). (e) Recovery with curvelet transform (PSNR=24.68).

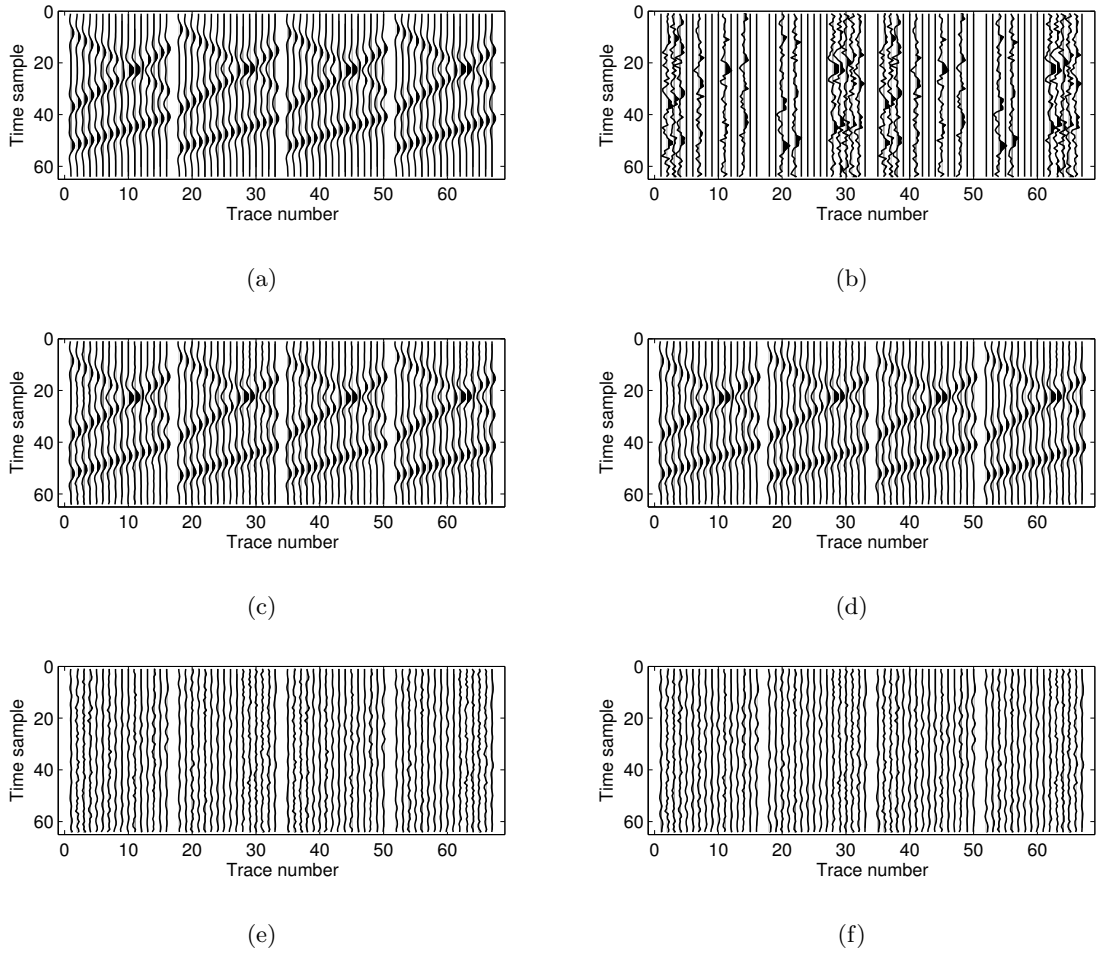


Figure 13: Simultaneous interpolation and denoising for 5D simulated data. (a) Simulated data. (b) 1/3 sub-sampled data. (c) Recovery with MC-DDTF (PSNR=30.89). (d) Recovery with RG-DDTF (PSNR=30.48). (e)(f) Recovery error $\times 3$ corresponding to (c) and (d), respectively.