Monte Carlo data-driven tight frame for seismic data recovery

Siwei Yu[1,2], *Jianwei Ma*[1], *Stanley Osher*[2]

[1]*Department of Mathematics, Harbin Institute of Technology, Harbin,* 150001*, China*

[2]*Department of Mathematics, University of California, Los Angeles, CA,* 90095*, USA*

ABSTRACT

Seismic data denoising and interpolation are essential preprocessing steps in any seismic data processing chain. Sparse transforms with fixed basis are often used in these two steps. Recently an adaptive learning method, the data driven tight frame (DDTF) method, was introduced for seismic data denoising and interpolation. With its adaptability to seismic data, the DDTF method achieves high quality recovery. For 2D seismic data, the DDTF method is much more efficient than traditional dictionary learning methods. But for 3D or 5D seismic data, the DDTF method results in high computational expense. The motivation behind this work is to accelerate the filter bank training process in DDTF, while do less damage to the recovery quality. The most frequently used method involves only a randomly selective subset of the training set. However, this random selection method uses no prior information of the data. We have designed a new patch selection method for DDTF seismic data recovery. We suppose that patches with higher variance contain more information related to complex structures, and should be selected into the training set with higher probability. First we calculate the variance of all available patches. Then for each patch, an uniformly 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 filter bank training. We call this procedure the Monte Carlo DDTF method. We test the trained filter bank on seismic data denoising and interpolation. Numerical results using this Monte Carlo DDTF method surpass random or regular patch selection DDTF when the sizes of the training sets are the same. We also use state-of-the-art methods based on curvelet transform, block matching 4D, and multichannel singular spectrum analysis (MSSA) as comparisons when dealing with field data.

INTRODUCTION

In seismic exploration, noise and missing traces are unavoidable. Noise can come from various sources (Liu et al., 2015). Missing traces emerge due to dead traces, or environmental and economic restrictions. Noisy and incomplete data cause low resolution in seismic migration, inversion , and amplitude-versus-angle analysis, etc. Denoising and interpolation are important preprocessing steps in the seismic data processing chain.

Random noise attenuation methods can mainly be classified into prediction error filters methods, rank reduction methods , and transformed domain methods. Prediction error filters 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 methods (Sacchi, 2009) or Cadzow filtering (Trickett, 2008), may also be used. These methods first arrange each frequency slice of seismic data into a Hankel matrix, then force the Hankel matrix to be low rank to attenuate the noise. Transformed domain methods have also been proposed for seismic data denoising, such as the discrete cosine transform (Lu and Liu., 2007) , the curvelet transform (Neelamani et al., 2008) and the seislet transform (Fomel and Liu, 2010).

Interpolation algorithms mainly fall into wave equation methods or signal processing methods. The former requires an assumption of a known velocity, and is computationally expensive. Signal processing methods are more popular due to their simplicity and efficiency. 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 by linear events. After applying a Fourier transform along the time axis of the data, each frequency slice obtained is analyzed in a regression relationship. The relationship can be solved accurately in low frequencies using a linear regression method or a rank reduction method. The relationship in high frequencies can be predicted from low frequencies with the assumption that events are linear. These sparse inversion methods have attracted researchers' attention recently due to recent developments on solving L_1 constrained problem. Among the sparse transforms, the Fourier transform seems to be 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-directional transforms, such as the curvelet transform (Naghizadeh and Sacchi, 2010), and the shearlet transform (Hauser and Ma, 2012) are considered due to their ability to represent seismic data sparsely. The seislet transform (Fomel and Liu, 2010) and the physical wavelet (Zhang and Ulrych, 2003) are specially designed wavelets for seismic data.

The sparse inversion methods mentioned above used in denoising and interpolation expand seismic data with a predefined basis, which is not adaptive to seismic data and may lead to low denoising or interpolation quality for data with complex structures. 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, cubes for 3D data [1], 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 improved results over fixed basis methods, although they require significant computational time and memory, which limits their application on large scale data processing, i.e., seismic data processing.

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

DDTF is still computationally prohibitive and not practical for large-scale high-dimensional

[1] we will always use 'patch' for consistency in this paper

seismic data processing. The filter training progress is computationally intensive and requires a significant amount of memory because of the large size of training samples. To improve efficiency, a randomly selected subset of all available training patches was used for training the filter bank (Cai et al., 2014; Liang et al., 2014). Many efforts have been made on large-scale problems using this 'random' strategy. For objective functions with large numbers of components, computing the gradient of the objective function exactly is difficult. Stochastic gradient methods (Bottou, 2010; Xu and Yin, 2015) estimate this gradient on a single randomly picked component, with moderate accuracy. NLM (nonlocal means) is another time-consuming image processing method. To avoid searching the whole image, one performs computation on a pre-selected subset of the image (Coupe et al., 2006). In Chan et al. (2014), the authors propose a Monte Carlo NLM method. For each pixel i of an image, a randomly selected set of reference pixels according to some sampling pattern is used to eliminate noise in the pixel i.

Inspired by the Monte Carlo NLM method, we design a new strategy of training subset selection for DDTF. This strategy significantly reduces the amount of training samples (less than 0.5% of all available patches), so it can reduce filtering training time from about 500 seconds to about 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 than that using a randomly or regularly selected training subset. For the field data, we also compare MC-DDTF with three other state-of-the-art methods: the curvelet transform (Candes and Donoho, 2003; Ma and Plonka, 2010; Neelamani et al., 2008), the block matching 4D (BM4D) (Maggioni et al., 2013), and the multichannel singular spectrum analysis (MSSA) (Oropeza and Sacchi, 2011). The curvelet transform is one of the geometrical wavelet transforms that provide multiscale and multidirection analysis of seismic data. The curvelet interpolation method used for comparisons is based on sparsity promotion and POCS strategy (Abma and Kabir, 2006). BM4D finds similar cubes among nonlocal area and a 4D transform is applied on the similar group simultaneously to suppress the noise. BM4D interpolation used for comparisons is also based on the POCS (project onto convex set) strategy. MSSA organizes each frequency slice of the $f-x$ spectrum into a block Hankel matrix with rank equal to the number of plane waves in the data. Missing samples will increase the rank of the block Hankel matrix. Consequently, rank reduction is proposed as a method to recover missing traces.

The rest of this paper is arranged as follows. In the second section, we first briefly introduce DDTF theory, then we present our new strategy of training subset selection, followed by an introduction to denoising and interpolation methods for seismic data. A summary of our algorithm is given at the end of this part. In the third section, we show numerical experiments for both denoising and simultaneously denoising and interpolation for both synthetic and field data. Finally, we propose future work.

THEORY

Data-Driven Tight Frame

Dictionary learning is equivalent to matrix decomposition with constraints. Dictionary learning decomposes the data as the product of a dictionary matrix and a coefficient matrix, with constraints on these two matrices. Approximate decomposition is often used in the presence of noise. For example, K-SVD (K-mean Singular Value Decomposition) imposes a sparsity constraint on the coefficient matrix, an energy constraint on the dictionary matrix, and a quadratic fidelity term for the approximate decomposition (Aharon et al., 2006). The objective function of K-SVD is:

$$\underset{V,W}{\operatorname{argmin}} \frac{1}{2}\|W^TV - PG\|_F^2 \text{ s.t. } \forall i, \|v_i\|_0 \leq T_0,\ \forall j, \|w_j\|_2 \leq 1. \tag{1}$$

where G is the original data, P stands for a possible patch transform, which breaks the data into small patches and then arranges/ reshapes the small patches of the data into columns to form a new matrix for training, as defined in Ma (2013) and many others. We denote the adjoint patch transform as P^T. W is the matrix dictionary. V is the coefficient matrix after expanding PG on W. $\|\cdot\|_2$, $\|\cdot\|_F$ and $\|\cdot\|_0$ indicate the vector norm, the Frobenius norm and the L_0 norm (the number of non-zeros in a vector). v_i is the ith column of V, w_j is the jth row of W. T_0 controls the sparsity of v_i. The minimization problem (1) can be solved by alternatingly solving for V and W, i.e., there are a sparsity coding stage and a dictionary updating stage. In the dictionary updating stage, one SVD decomposition is used to update each w_j, causing its inefficiency.

In the manner as the K-SVD, DDTF imposes a sparse constraint on the coefficient matrix. But unlike the K-SVD, DDTF imposes a 'tight frame' constrain on the dictionary matrix. The objective function for the filters training in DDTF is as follows:

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

where I is the identity matrix. The constraint $W^T W = I$ means W is a tight frame [2]. The coefficient α balances the fidelity term and regularization term. The minimization in (2) can also be solved by alternatingly solving for V and W (see Cai et al. (2014) for details). With the tight frame constraint, W is updated by one SVD decomposition in the dictionary updating stage. In this way the DDTF achieves much higher efficiency than K-SVD.

Monte Carlo Patch Selection

The patch transform breaks the original data into small patches, which are used for training the filter bank. For 3D data with size n^3 and patch size r^3, the set of patches contains $(n - r + 1)^3 r^3$ data points. Large numbers of patches may be burdensome in terms of memory and computation during the filter training progress. From the K-SVD and DDTF codes available online, we see only a randomly selected subset of the original patches are used as the training set. We raise the question: If the number of the training patches is bounded, can we obtain a 'better' filter bank by selecting the subset of the training patches

[2]This expression is not strict. Actually W and W^T are the analytic and synthetic operators of the tight frame. W is obtained by all shifts of an orthogonal filter bank. We call W the filter bank (the rows of W filters) directly. We find this non strict expression is more simple and straightforward. The reader can refer to Cai et al. (2014) for the strict expression.

in some manner other than randomly? On the other hand, for a given recovery quality, can we select a subset with smaller size than random selection? By 'better', we mean a higher quality denoising or interpolation result can be obtained with a filter bank trained from the selected patches.

The answer to these questions is yes, because random selection uses no priori information of the patches. We propose a new patch selection strategy based on Monte Carlo theory. The basic assumption is that patches containing complex structures should be selected in the training subset with high probability in order to keep the details and improve the resolution of the result. More complex patches in the training set mean that the trained filter can sparsely represent complex patches, which will improve the recovery of complex structures. If the training set contains more simple patches than complex patches (which is the case in random selection or regular selection as shown and explained in Figure 1(e)), the trained filters tend to sparsely represent the simple ones rather than the complex ones. That is to say, information on simple patches will suppress the information on complex patches. Complex patches stand for the details of the data, which are more important for improving the resolution of the result and should be preferentially considered in the training set preferentially.

The next question is how to measure the complexity of the patches ? Suppose the data is clean and the amplitude of the events change only a little, we propose using the patch's variance to represent the complexity of its structure. For example, the variance equals zero means the values on the patch are the same everywhere, which also means the patch locates no events. Patches located on events result in higher variance. Each pixel in the image corresponds to variance of one patch (without considering the boundary), so variances of all patches can form an 'image of variance', which we will show later.

The amplitude of the events will affect the calculation of variance. The patches should be normalized before calculating their variances. Usually seismic energy is more related to vertical direction than horizontal direction. Energy will become weaker for deeper events. So we extract one trace to represent the seismic energy in depth. The normalization function is combined by using linear lines going through the points of highest amplitude of every event (Figure 1(f)). The points are selected manually empirically. The normalization function should form an envelope of the selected trace. We will automate this step in future work.

For data corrupted with noise, we also apply a simple median filter before calculating the variance. Our method does not deal with weak events and strong noise at the same time, where a large variance might not indicate useful features. There could be other methods of selecting the patches by utilizing the prior information, such as entropy, which is a concept used for measuring the amount of information in data. In the following numerical experiments, we show how the normalization and median filtering work as preprocessing steps for calculating the variance.

In Figure 1(a), the 2D seismic data is composed with 128 traces and 128 time samples. The data contains four linear events: the shallow ones have amplitude of 1.0 and the deep ones have amplitude of 0.4. The data is normalized to $(0, 1)$ before displaying. The patch size is set as 8 (time)$\times$8 (space). The variance image of this data is given in Figure 1(c). The variances in the deep area are much smaller than in the shallow area. Therefore the data must be normalized before calculating the variance, as shown in Figure 1(b). The corresponding variance image in Figure 1(d) contains equivalent variances at shallow and deep area. In Figure 1(f), the blue solid line and red dashed line show the selected trace and normalization function used in this example.

In Figure 2(a), the data is normalized but contaminated with Gaussian noise. The noise is amplified by the normalization function at the deeper area, resulting in higher variance, as shown in Figure 2(c). In Figure 2(b), a median filter is applied to the noisy data, and the corresponding variance image is shown in Figure 2(d), which is a better approximation of Figure 1(d).

The histogram of Figure 1(d) is shown in Figure 1(e). The horizontal axis is variance, the vertical axis is the number of pixels per variance, i.e., the density of pixels at that variance.

Both axes are normalized before displaying. From the distribution, we can see that most of the variances are relatively of low value. If we select the patch regularly or randomly, most of them will contain little information about complex structures. We emphasize that higher variance should be considered with priority, but smaller variance should not be neglected either. The selection algorithm is similar to a Monte Carlo method, so we named it the Monte Carlo patch selection, which is summarized in algorithm (1):

Algorithm 1 Monte Carlo patch selection algorithm

Input: Patches after patch transform on normalized and median filtered data.

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

2: For patch i, first compute its variance v_i, then generate a random number $r_i \subseteq (0, v_{max})$ with uniform distribution, if $r_i < kv_i$, then keep the original patch i before normalization and median filtering.

Output: Selected training subset.

In algorithm (1), the coefficient k controls both the efficiency of filter training and the quality of the filtering result. Figure 3 shows the relationship between the percentage of selected patches and k. The relationship is easily obtained based on numerical experiments with different k on data in Figure 1(b). For different data, we first obtain a similar curve by numerical experiments, then select k corresponding to certain selected percentage. Usually the selected percentage under 1% works well. In our tests, k is set as $0.01 \sim 0.1$.

Seismic Data Denoising and Interpolation

The obtained filter bank W is used for denoising the original data G with a thresholding method. We donate the denoising function as $D_\lambda(G)$, defined as follow:

$$D_\lambda(G) = P^T W^T T_\lambda(WPG) \tag{3}$$

where λ is a denoising parameter associated with noise level, and T_λ is the soft shrinkage operator.

For seismic interpolation we introduce the POCS (Abma and Kabir, 2006) based method. The algorithm is described in algorithm (4) in the appendix.

Monte Carlo DDTF for Seismic Data Recovery

We summarize the abbreviations for DDTF with different patch selection methods in table (1):

MC-DDTF based seismic data denoising and interpolation algorithms are described in algorithm (2) and (3) separately:

Algorithm 2 MC-DDTF based seismic data denoising algorithm

Input: Noisy data, denoising parameter.

1: Transform the raw data into patch space, and use 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 minimization (2).
3: Denoise the data with trained filter bank using equation (3).

Output: Denoised data.

Algorithm 3 MC-DDTF based seismic data interpolation algorithm

Input: Sub-sampled data, Sampling matrix.

1: Transform the raw data into patch space, and use 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 minimization (2).
3: Interpolate the data with trained filter bank using POCS method (Algorithm 4).
4: Use the interpolated data as raw data, and repeat 1, 2, 3 until meeting convergence condition.

Output: Interpolated data.

In Figure 4, we apply different patch selection methods on the model in Figure 4(b) (

SNR = 2.16). Here, SNR (signal-to-noise ratio) means the ratio of the energy of clean data to the energy of noise. 0.94% of all patches are 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 Figure 4(c)-4(f). We can see that the patches are mainly located on the events with MC-DDTF in Figure 4(e). Figure 4(g)-4(j) are the trained filters corresponding to Figure 4(c)-4(f). Notice that in Figure 4(i), there is one filter in the red ellipse, which direction is going upward from left to right, standing for the feature of the weak and non-flat event. Even with FP-DDTF, we cannot obtain such a filter. This is because in FP-DDTF, the training patches contain patches with significant noise, which will also, unfortunately, suppress significant signal. Figure 4(k)-4(n) are denoising results corresponding to Figure 4(c)-4(f), with SNR = 12.02, 11.64, 13.36, 13.97 separately. FP-DDTF achieves the highest SNR, followed by MC-DDTF. But MC-DDTF achieves the best recovery result for the weak feature, as marked in the red ellipses. The filter training time of MC-DDTF is only 0.034s, more than 15 times faster than FP-DDTF (0.53s).

In Figure 5(a), we test the filter training time of FP-DDTF and MC-DDTF with respect to different sizes of 3D seismic data. The horizontal axis means the size of the data is $n \times n \times n$ (all units are set as '1'). About 0.78% of all patches are used in MC-DDTF. We can see that MC-DDTF achieves much higher efficiency than FP-DDTF. During our tests, FP-DDTF runs out of memory (4G RAM) when the data size is over $80 \times 80 \times 80$. In Figure 5(b), we test patch selection time of RD-DDTF and MC-DDTF. We can see that MC-DDTF takes twice the time as RD-DDTF, i.e., the additional variance calculation only takes the same time as the random patch selection. In total, the filter training time of MC-DDTF is about 1.5 times of RD-DDTF for the data with the tested sizes.

NUMERICAL RESULTS

We test MC-DDTF for denoising and interpolation on 3D/5D simulated data and field data. We show that MC-DDTF accelerates the filter training progress significantly, with little damage to reconstruction quality. A comparison with RD-DDTF and RG-DDTF shows that we achieve a better reconstruction quality with MC-DDTF in terms of SNR. For the field data, we also compare MC-DDTF with three other state-of-the-art methods: the curvelet transform based method, the BM4D based method, and the MSSA based method. All tests were carried out on a personal laptop, with Matlab on Windows 7 operation system, 4G RAM and Intel Core i-7 CPU.

Denoising on shot gather data set 1:

A shot gather (Shahidi et al., 2013) is displayed in Figure 6(a), with its noise corrupted version (SNR=3.57) in Figure 6(b) (If not specially mentioned, when displaying 3D data, the center slice of each axis is shown). The data is composed of 128 (time)×128 (shot)×128 (receiver) samples. The time sampling interval is $4ms$, the spatial sampling interval in shot and receiver directions are both $0.02km$. The patch size is 8 (time)×8 (shot)×8 (receiver) samples. The sample interval and patch size are the same for the following tests if not specified. Figures 6(c)-6(f) show the denoising results. FP-DDTF achieves highest SNR=19.88, with filter training time=427s. Here only 1% of all patches are used as an approximation to FP-DDTF due to restriction of memory. For MC-DDTF, RD-DDTF and RG-DDTF, we select 0.07% of all patches as a training subset. This leads to filter training time=9.74s. MC-DDTF achieves higher SNR (17.93) than RD-DDTF (16.89) and RG-DDTF (17.10). The filtering times of these three methods are the same (96.67s). In Figure 7(a)-7(d), we show the residual between the recovered data and noisy data in a time-receiver plane, corresponding to the center slice of Figure 6(c)-6(f), which makes the results more comparable. In the red ellipses, we can see RD-DDTF and RG-DDTF leave obviously useful energy. However, in MC-DDTF, there is significantly less useful energy left. In FP-DDTF, the useful energy is hardly ever seen.

In Figure 8, we show how the size of training subset affects the filtering result. Figure 8 corresponds to the test in Figure 6. The x-axis stands for the percentage of the training subset of all patches. The y-axis stands for SNR of filtering result. The horizontal dashed line is the SNR of FP-DDTF method for comparison. When the percentage is less than 0.5%, MC-DDTF gets higher SNR than RD-DDTF and RG-DDTF. The advantage is much greater when the percentage is less than 0.1%. This is because, for seismic data, patches on events have high variance, and will be selected with higher probability in the Monte Carlo patch selection. Random or regular patch selection treats the patches the same without regard to the existence of events in the patch, so information in the events may be submerged by the other area. When the percentage is over 0.6%, MC-DDTF, RD-DDTF and RG-DDTF achieve almost the same SNR. This is, because for the simulated model in Figure 6(a), all the important information are learned by MC-DDTF, RD-DDTF and RG-DDTF when the percentage is over 0.6%.

Interpolation for migration data set 2:

A migrated marine data (courtesy of Elf Aquitaine)(Sergey Fomel, 2003) is shown in Figure 9(a). The data size is 512 (time)×512 (inline)×16 (crossline). In Figure 9(b), we decimate the data in Figure 9(a) by random 50% traces in inline-crossline plane. Nearest neighborhood interpolation is used to generate the initial training data. Four iterations are used with the MC-DDTF in Algorithm (3). The same setting is used in the following examples. 0.21% patches are used in the filter training progress. Filter training takes 5.08s. Interpolation takes 1210s. Finally we achieve SNR=37.11 with MC-DDTF , and the result is shown in Figure 9(c). For completeness, we give the results of BM4D (SNR=32.66) interpolation, the curvelet transform interpolation (SNR=30.93) and MSSA interpolation (SNR=32.14) as comparisons in Figure 9(d)- Figure 9(f). In Figure 10, we give the zoomed version (time: $0 - 0.512s$, crossline: $5.14 - 7.68km$, inline: $0.16km$) of the results in Figure 9.

Interpolation for migration data set 3:

A 3D migrated data (a test data from the software 'GEOFRAME') is displayed in Figure 11(a). The data size is 240 (time)×221 (inline)×271 (crossline) samples, and the $120th$ slice of each direction is shown. The 50% decimated version is shown in Figure 11(b). 0.027% of all patches are used in the filter training progress. Filter training takes only $6.30s$. Interpolation takes $6500s$. The interpolation results are shown in Figure 11(c)-11(f). The reconstruction errors are given in Figure 12 for better comparison of the results.

Interpolation for a 5D simulated data:

We test simultaneously 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 is shown in Figure 13(a), with the noisy and decimated version shown in Figure 13(b). 0.043% patches are used for filter bank training. The patch size is 4^5. The training subset forms a matrix with size 1024×725, which requires $2.84MB$ memory. For all patches, the size of the matrix is 1024× 1685099, which requires $6.56GB$ memory. Recovery results with MC-DDTF and RG-DDTF are shown in Figure 13(c) and 13(d). MC-DDTF achieves higher SNR (15.96) than RG-DDTF (15.55). The recovery error (times 3) sections are shown in Figure 13(e) and 13(f).

Simultaneous denoising and interpolation for a field prestack shot gather

Figure 14(a) shows a shot gather with 800 time samples and 128 traces from a field prestack seismic data used for simultaneous noise attenuation and interpolation. A zoomed version $(1.6s - 2.3s, 0.6 - 0.8km)$ is displayed in Figure 14(b). The data is regularly 50% subsampled in Figure 14(c). 1.69% of all patches are used in the training set of MC-DDTF. Figure 14(d)-14(e) are recovered shot gathers with MC-DDTF and FP-DDTF. MC-DDTF achieves higher interpolated results than FP-DDTF. FP-DDTF seems overfit the data. The results are more comparable in the recovery errors in Figure 14(f)-14(g). The filter training progress of MC-DDTF takes 0.03s, while it takes 1.28s in FP-DDTF.

CONCLUSION

Adaptive methods such as DDTF are proposed for recovery of complex structures in seismic data, but they are memory and time consuming for high dimensional situation. In this paper, we introduce a Monte Carlo based DDTF, which is designed to make the filter training progress of DDTF more efficient for high dimensional seismic data. MC-DDTF accelerates filter training progress with less damage to the recovered result than RD-DDTF and RG-DDTF. Our method also outperforms other state-of-art methods such as the BM4D based method and the curvelet based method. The disadvantage of our method is that it fails to deal with weak features and strong noise at the same time. Future work will focus on: 1, extracting patches containing weak features and strong noise 2, parallelizing the filtering progress to make the adaptive dictionary learning method handle high dimensional seismic data within a reasonable time.

ACKNOWLEDGEMENT

The authors would like to thank the editors and four reviewers for their helpful comments to improve this work, Mauricio Sacchi for providing the MSSA code and Bingbing Sun for providing the field test data from the software 'GEOFRAME'. This work is supported by NSFC (grant numbers NSFC 91330108,41374121,61327013), and the Fundamental Research Funds for the Central Universities (grant number HIT.PIRS.A201501).

APPENDIX A

ALGORITHM OF POCS

Algorithm 4 POCS Algorithm

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

1: Denoise the data with equation (3)

$$G^* = D_\lambda(G) \tag{A-1}$$

2: Reinsert original traces

$$G^* = (I - M)G^* + MG^0 \tag{A-2}$$

3: Set $G = G^*$ and repeat 1, 2 until certain convergence condition is satisfied.

Output: Interpolated data G^*.

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.

Candes, E., and D. Donoho, 2003, New tight frames of curvelets and optimal representations of objects with C2 singularities: Comm. Pure Appl. Math., **57**, no. 2, 219-266.

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**, 524–538, doi:10.1190/1.1444559.

Fomel, S., 2003, Time-migration velocity analysis by velocity continuation: Geophysics, **68**, 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**, 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.

Ma, J., and G. Plonka, 2010, The curvelet transform: IEEE Signal Processing Magazine, **27**, no. 2, 118-133.

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**, 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., and W. Yin, 2015, Block stochastic gradient iteration for convex and nonconvex optimization: SIAM J. Optimization, **25**, no. 3, 1686-1716, doi:10.1137/140983938.

Yu, S., J. Ma, X. Zhang, M. Sacchi, 2015, Denoising and interpolation of high-dimensional seismic data by learning tight frame: Geophysics, **80**, no. 5, V119–V132, doi:10.1190/geo2014-0396.1.

Zhang, R., and T. Ulrych, 2003, Physical wavelet frame denoisng: Geophysics, **68**, 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.

TABLES

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

LIST OF FIGURES

form. (f) Recovery with MSSA.

12 Reconstruction errors of time-crossline plane of the results in Figure 11. (a)-(d) correspond to MC-DDTF, BM4D, the curvelet transform and MSSA methods separately.

13 Simultaneous interpolation and denoising for 5D simulated data. (a) Simulated data. (b) 1/3 sub-sampled data. (c) Recovery with MC-DDTF (SNR=15.96). (d) Recovery with RG-DDTF (SNR=15.55). (e)(f) Recovery error $\times 3$ corresponding to (c)(d).

14 Simultaneous denoising and interpolation for a field prestack shot gather. (a) Original prestack shot gather. (b) A zoomed portion of original data. (c) 50% regularly sub-sampled data. (d) Recovered shot gather using MC-DDTF. (e) Recoverd shot gather using FP-DDTF. (f)(g) Recovery difference corresponding to (d)(e).

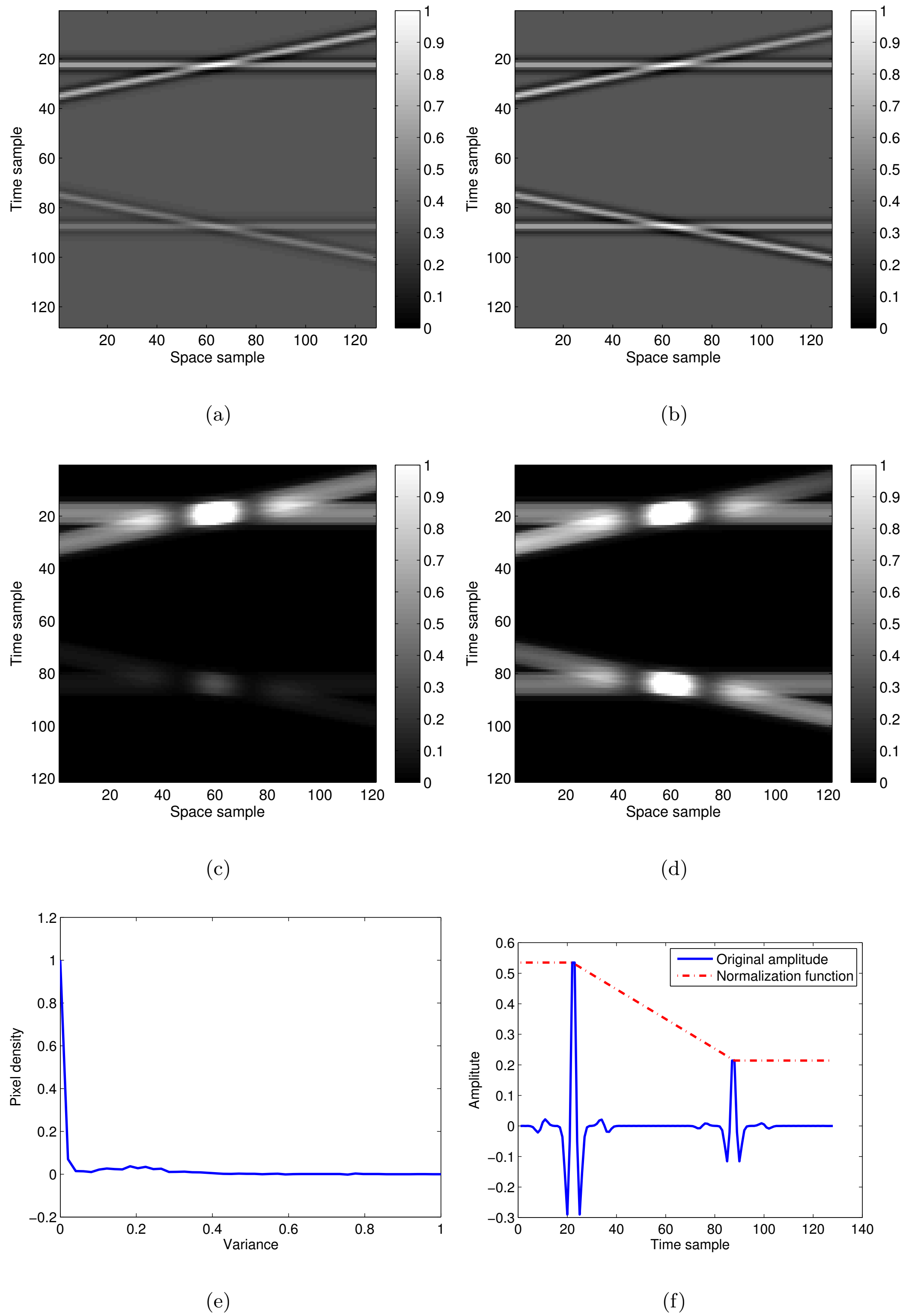
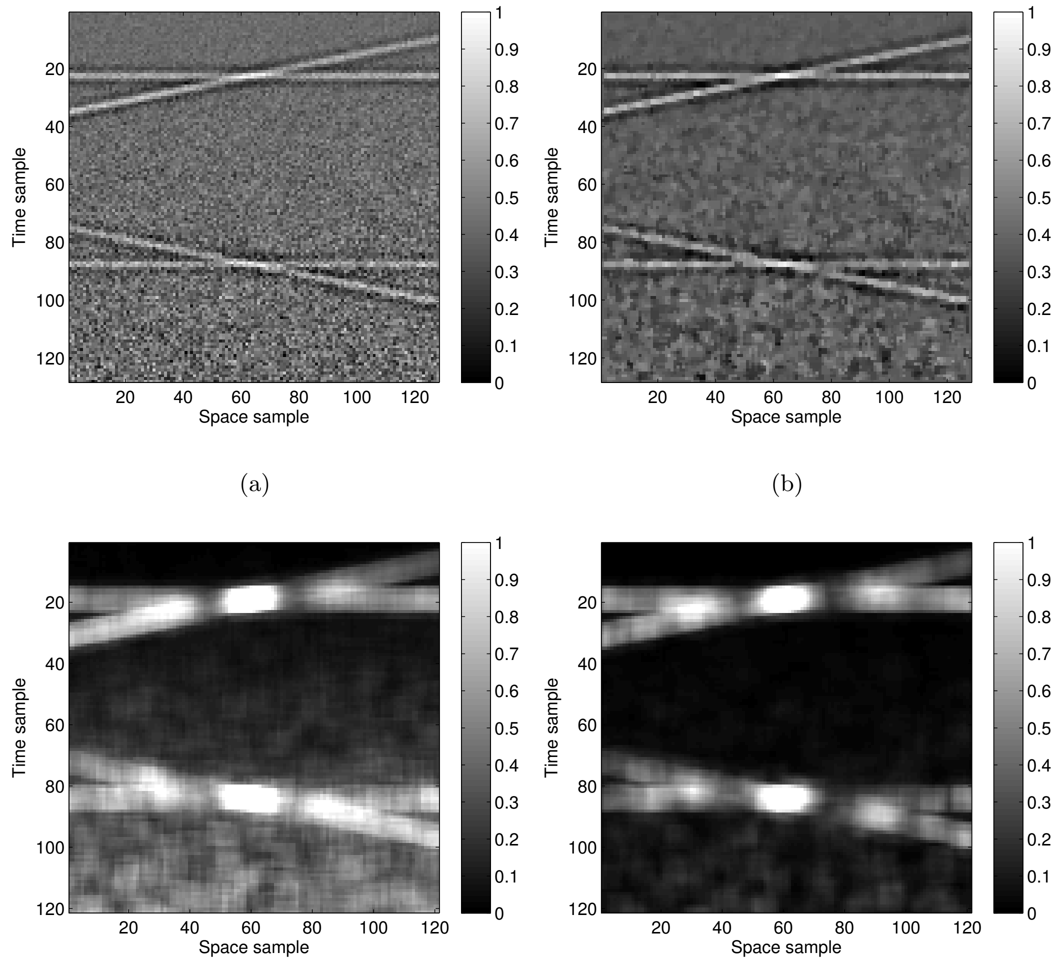


Figure 1: A demonstration of variance distribution of seismic data. (a) 2D seismic data. (b) Normalization of data in (a). (c)-(d) Variance distribution with patch size = 8 for (a)-(b). (e) Variance distribution of (d). (f) Normalization function.



(c) (d)

Figure 2: A demonstration of variance distribution of seismic data with noise. (a) 2D noisy and normalized seismic data. (b) Denoised result of (a) with median filter. (c)-(d) Variance distribution with patch size = 8 for (a)-(b).

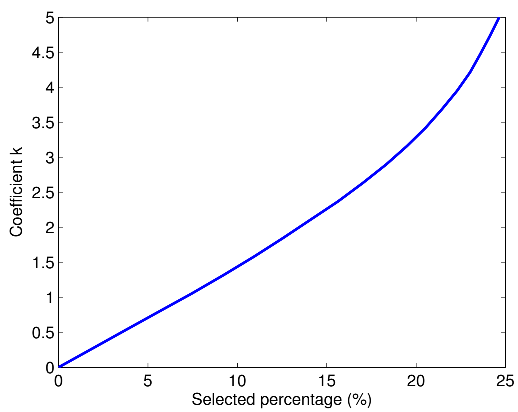
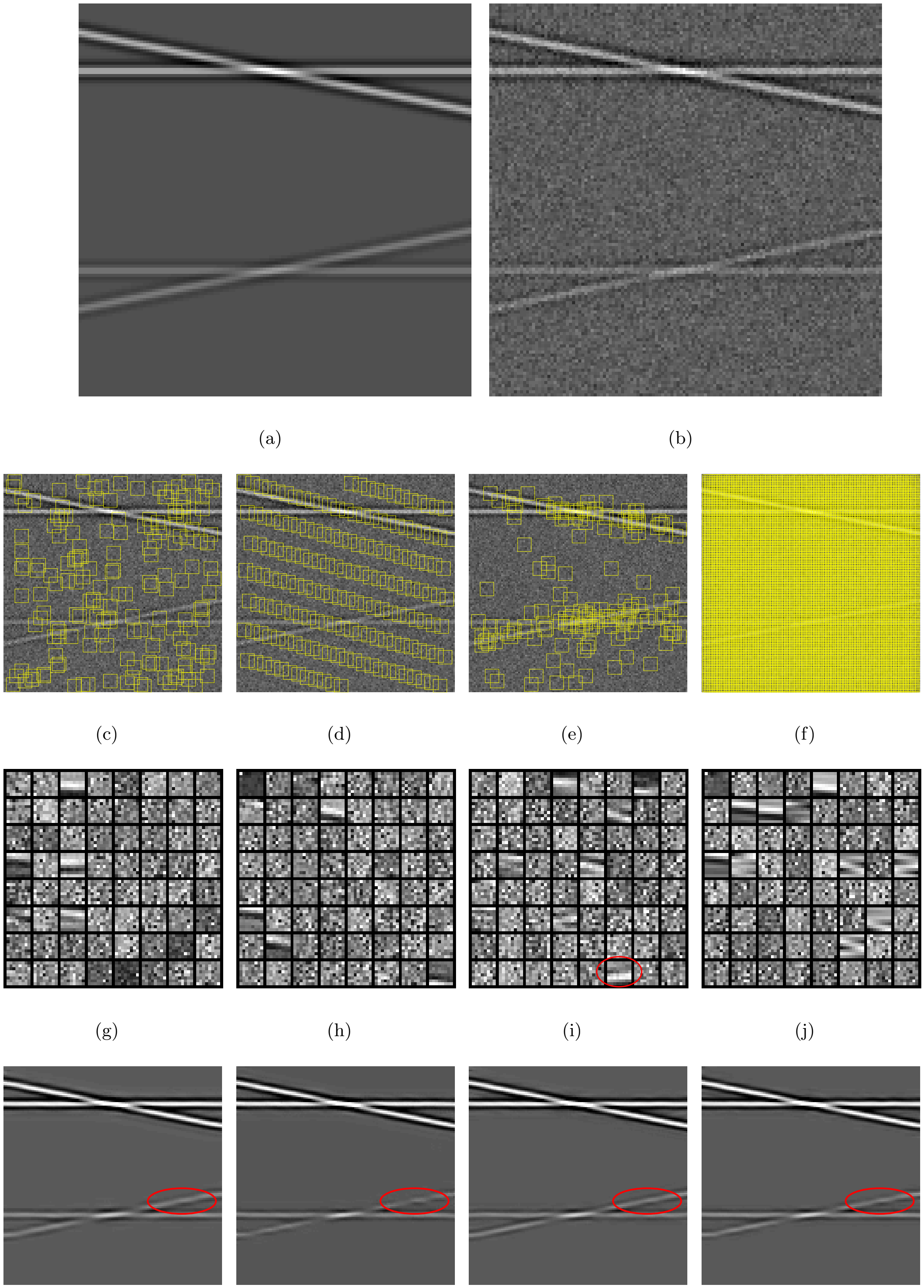


Figure 3: The percentage of selected patches from all patches versus coefficient k.



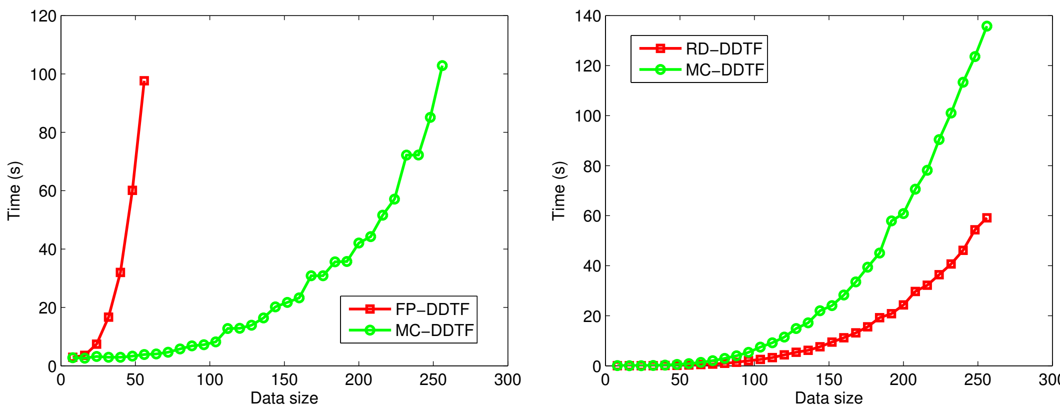
(a) (b)

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

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

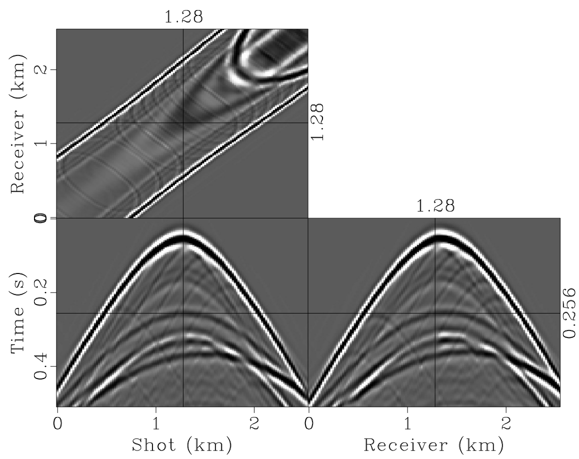
(k) (l) (m) (n)

Figure 4: A demonstration of different patch selection methods. (a) 2D seismic data. (b) Noisy version of (a). (c)-(f) Select patches randomly, regularly, with Monte Carlo method

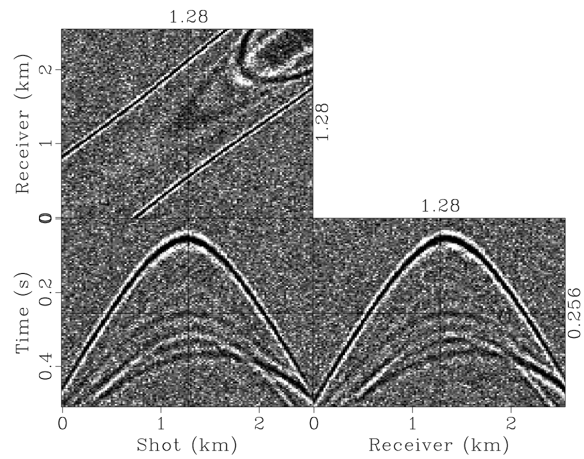


(a) (b)

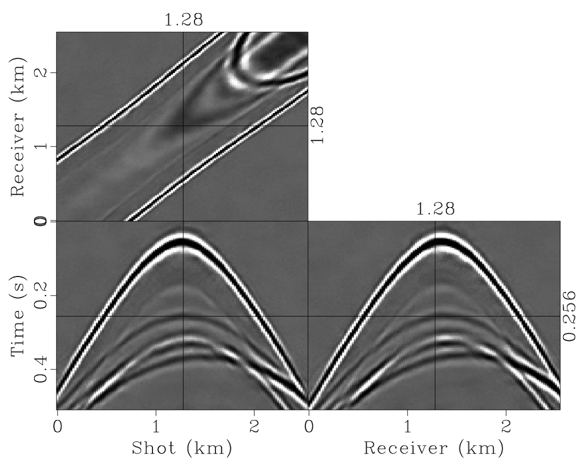
Figure 5: Time analysis on 3D seismic data. (a) Filter training time for FP-DDTF and MC-DDTF patch selection. (b) Patch selection time for RD-DDTF and MC-DDTF.



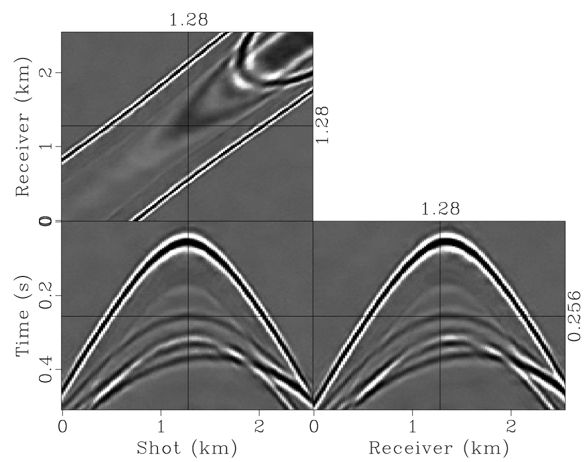
(a)



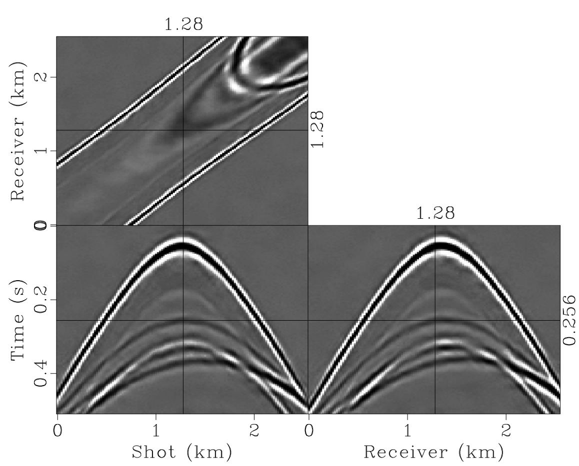
(b)



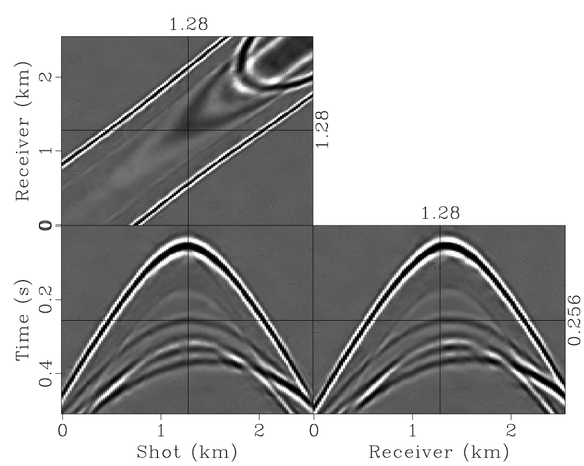
(c)



(d)



(e)



(f)

Figure 6: Denoising result for simulated data. (a) Clean data. (b) Noisy data (SNR=3.57). (c) Recovery with RD-DDTF (SNR=16.89). (d) Recovery with RG-DDTF (SNR=17.10). (e) Recovery with MC-DDTF (SNR=17.93). (f) Recovery with FP-DDTF (SNR=19.88).

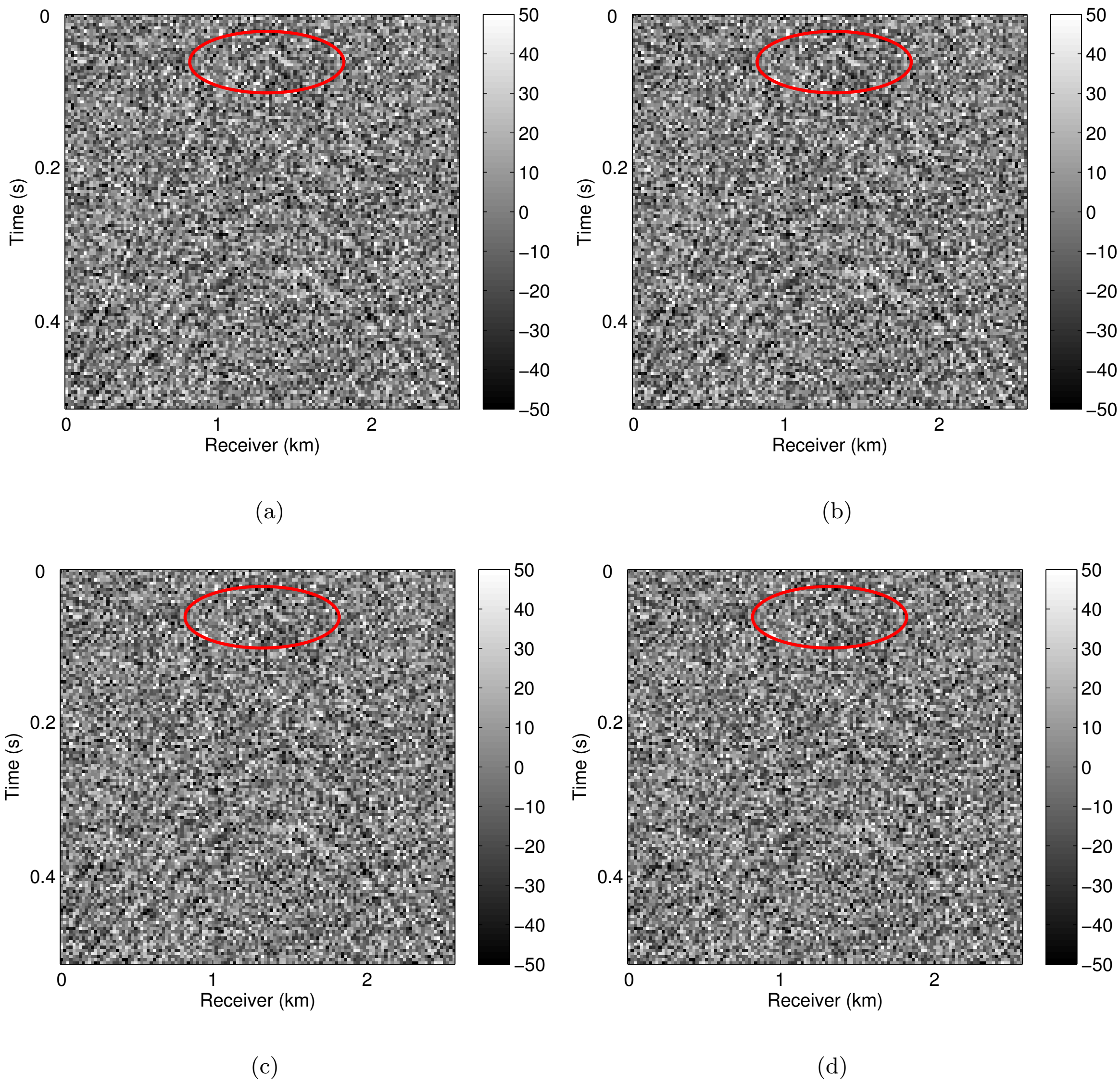


Figure 7: Recovery residual in time-receiver plane of results in Figure 6. (a)-(d) correspond to RD-DDTF, RG-DDTF, MC-DDTF and PF-DDTF separately.

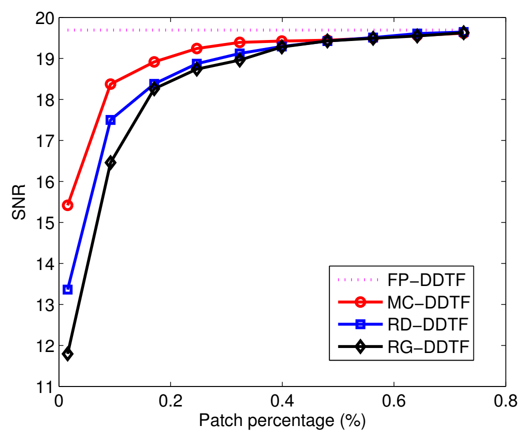
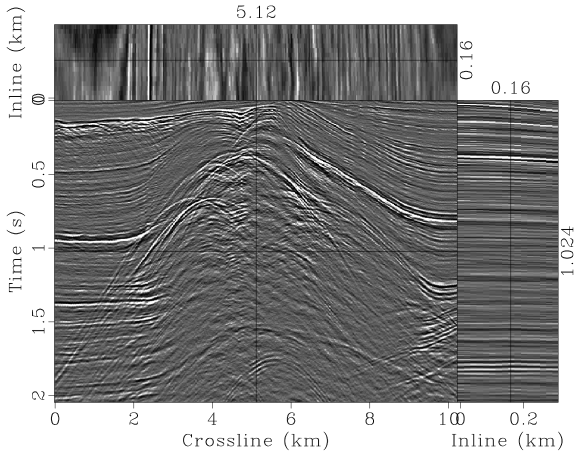
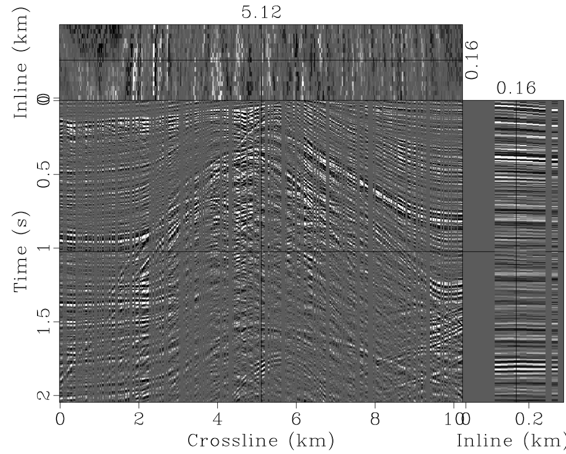
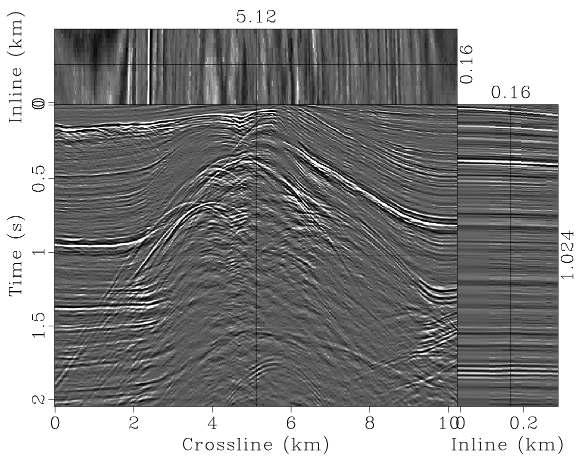


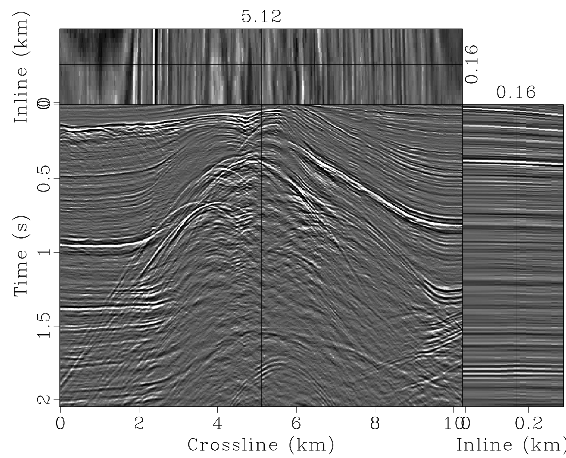
Figure 8: SNR versus patch percentage correspondding to test in Figure 6



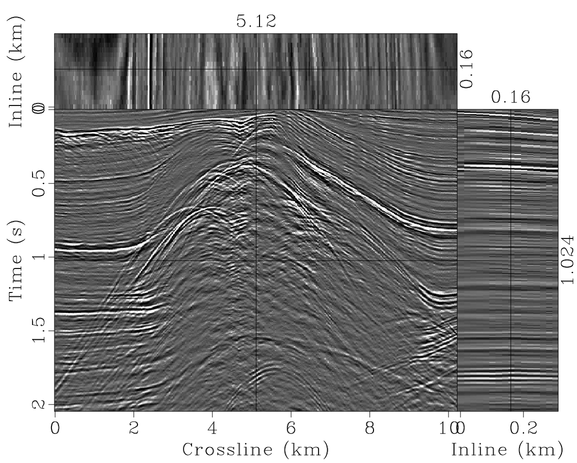


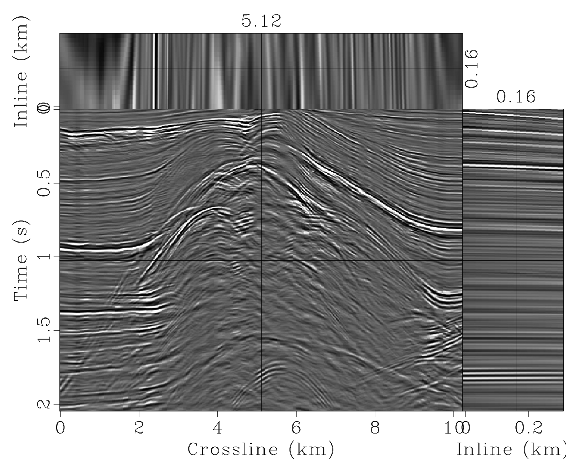
(a) (b)





(c) (d)





(e) (f)

Figure 9: Interpolation for field data. (a) Original data. (b)50% sub-sampled data. (c) Recovery with MC-DDTF (SNR=37.11). (d) Recovery with BM4D (SNR=32.66). (e) Recovery with the curvelet transform (SNR=30.93). (f) Recovery with MSSA (SNR=32.14).

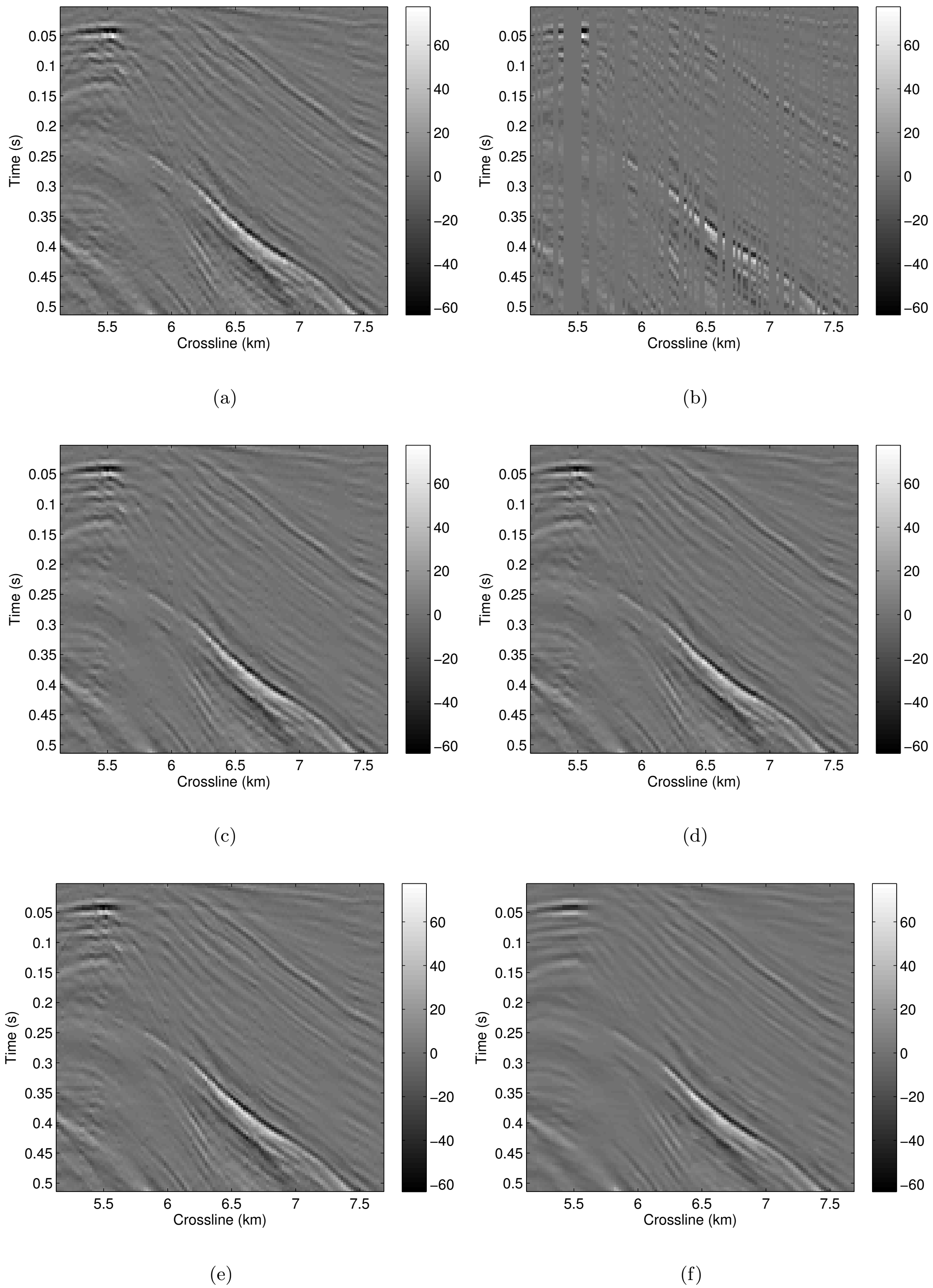


Figure 10: Zoomed version of time-crossline plane of the results in Figure 9. (a)-(b) Original and subsampled data. (c)-(f) Recovery results correspond to MC-DDTF, BM4D, the curvelet transform and MSSA separately.

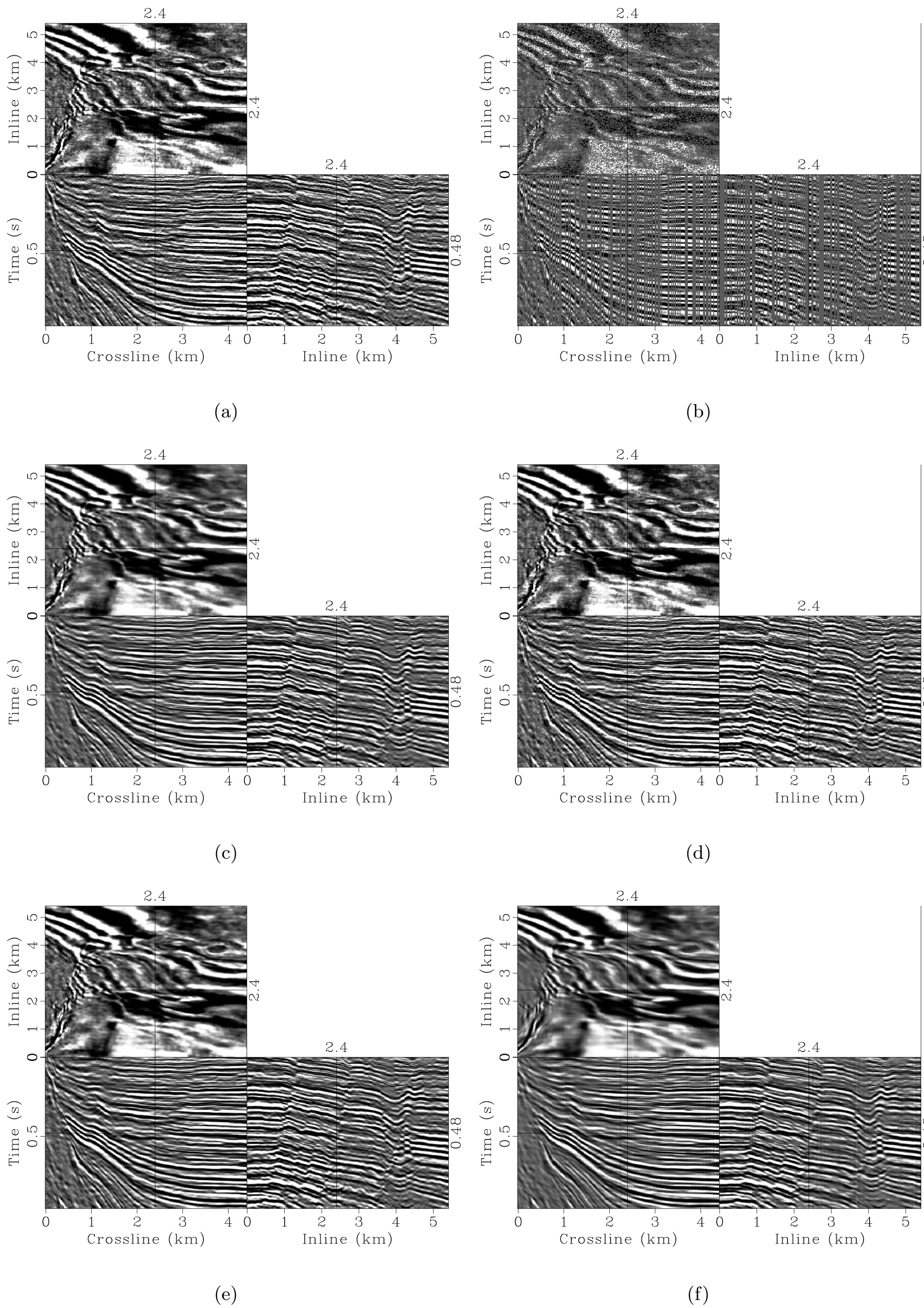


Figure 11: Interpolation for field data. (a) Original data. (b) 50% sub-sampled data (c) Recovery with MC-DDTF. (d) Recovery with BM4D. (e) Recovery with the curvelet transform. (f) Recovery with MSSA.

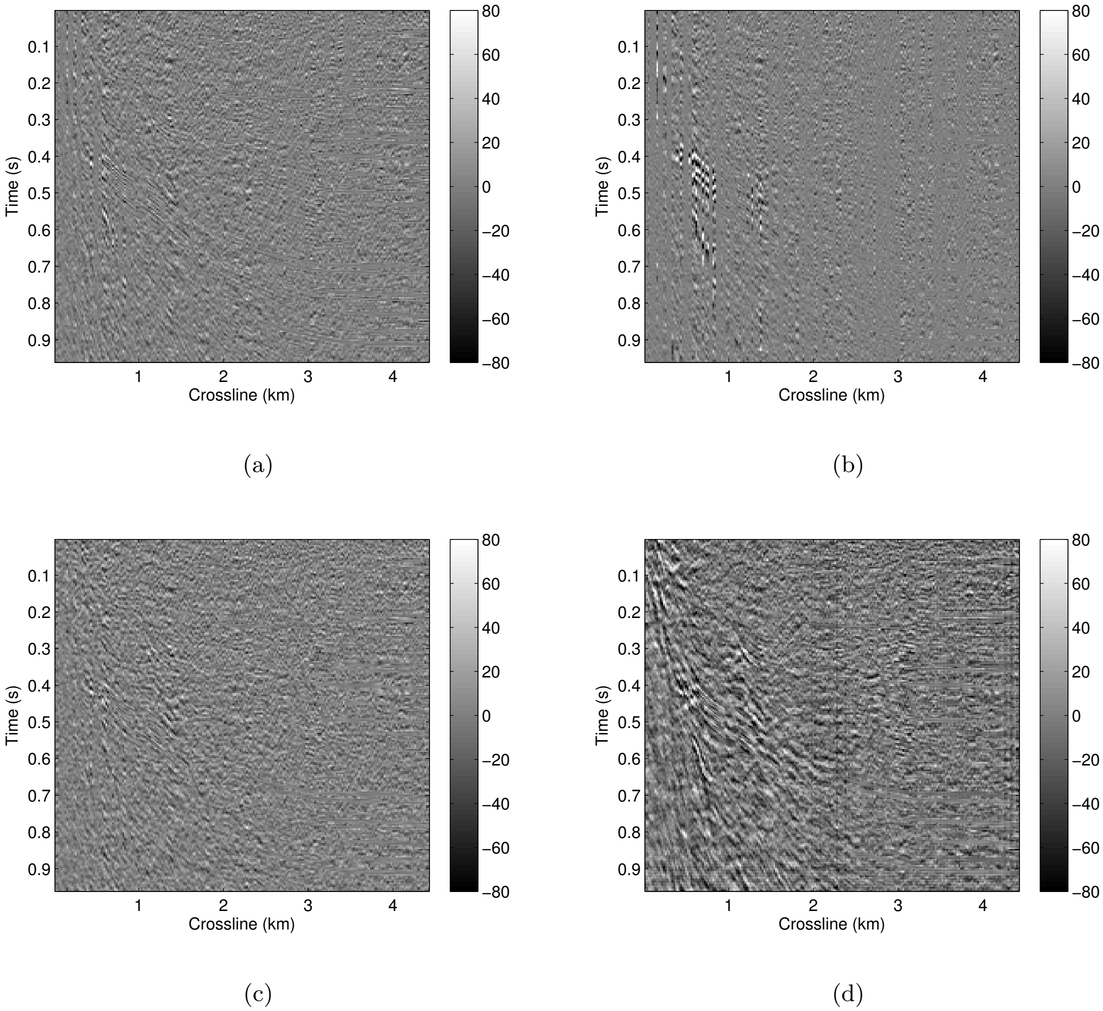


Figure 12: Reconstruction errors of time-crossline plane of the results in Figure 11. (a)-(d) correspond to MC-DDTF, BM4D, the curvelet transform and MSSA methods separately.

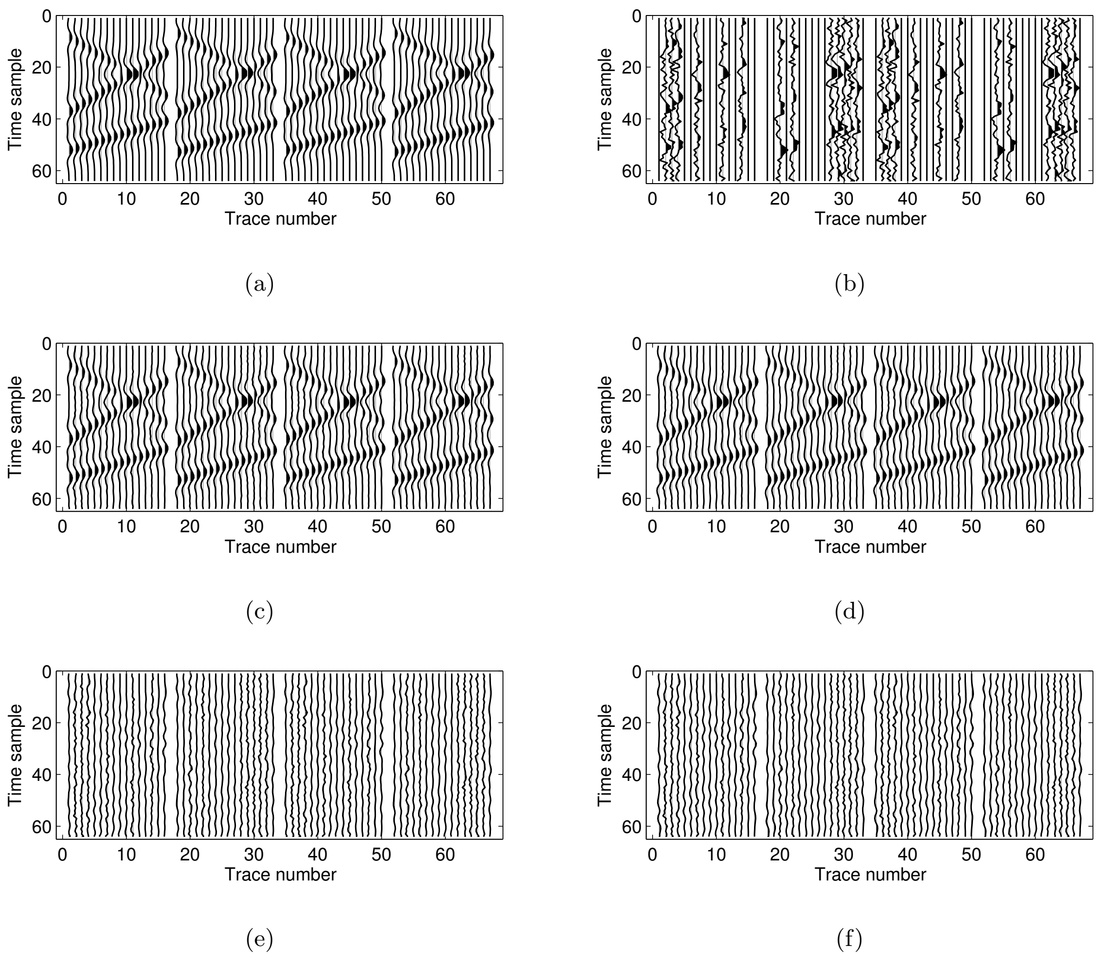
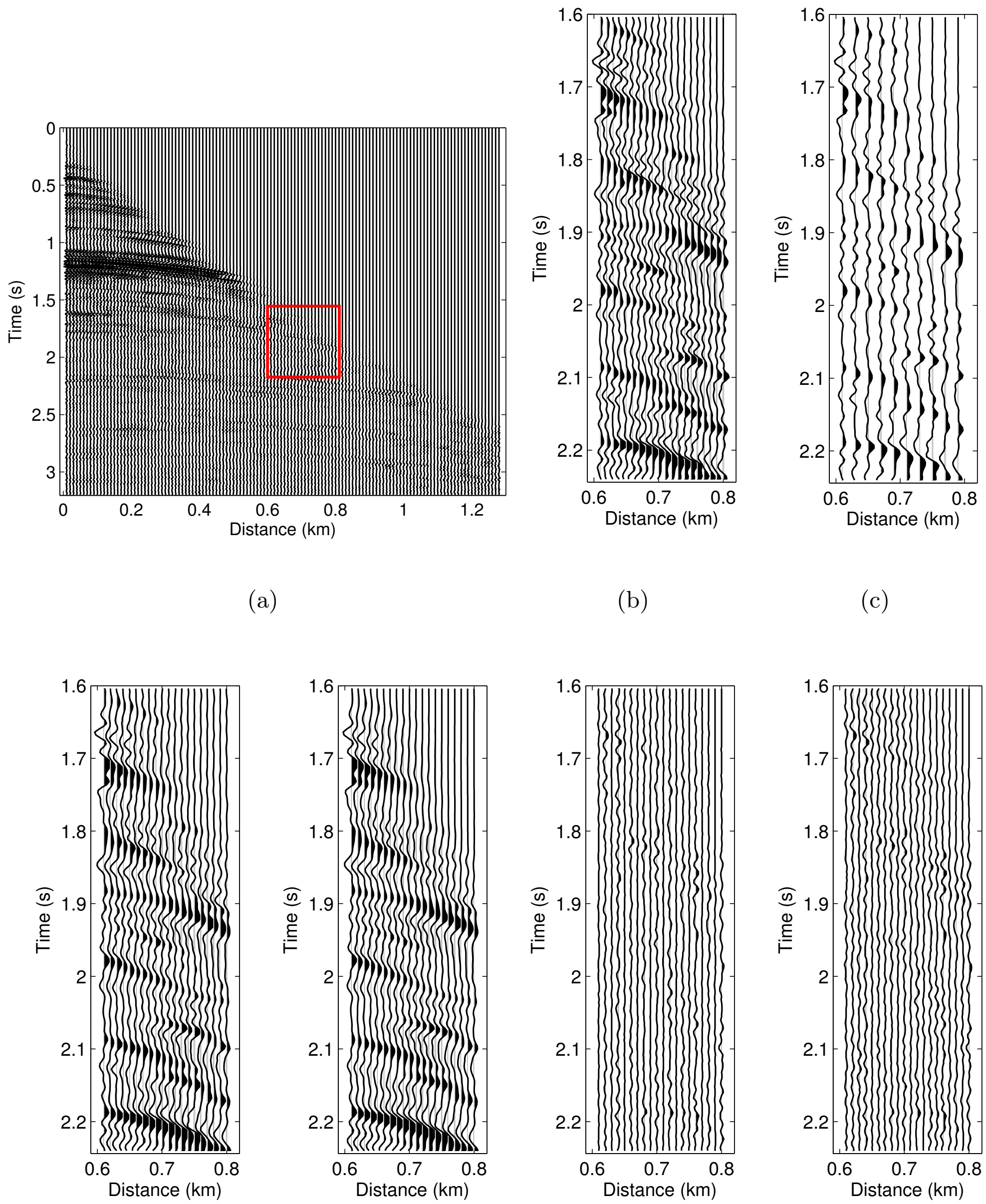


Figure 13: Simultaneous interpolation and denoising for 5D simulated data. (a) Simulated data. (b) 1/3 sub-sampled data. (c) Recovery with MC-DDTF (SNR=15.96). (d) Recovery with RG-DDTF (SNR=15.55). (e)(f) Recovery error $\times 3$ corresponding to (c)(d).



(a) (b) (c)

(d) (e) (f) (g)

Figure 14: Simultaneous denoising and interpolation for a field prestack shot gather. (a) Original prestack shot gather. (b) A zoomed portion of original data. (c) 50% regularly sub-sampled data. (d) Recovered shot gather using MC-DDTF. (e) Recoverd shot gather using FP-DDTF. (f)(g) Recovery difference corresponding to (d)(e).