EURASIP Journal on Advances in Signal Processing

SpringerOpen

Inverse polynomial reconstruction method in DCT domain

EURASIP Journal on Advances in Signal Processing 2012,
2012:133 doi:10.1186/1687-6180-2012-133

Hamid Dadkhahi (dadsy002@mymail.unisa.edu.au)
Atanas Gotchev (atanas.gotchev@tut.fi)
Karen Egiazarian (karen.egiazarian@tut.fi)

ISSN 1687-6180

Article type Research

Submission date 2 May 2011

Acceptance date 2 May 2012

Publication date 6 July 2012

Article URL http://asp.eurasipjournals.com/content/2012/1/133

EURASIP Journal on Advances in Signal Processing manuscript No.
(will be inserted by the editor)

Inverse polynomial reconstruction method in DCT domain

Hamid Dadkhahi*[1,2], Atanas Gotchev[2] and Karen Egiazarian[2]

[1]Institute for Telecommunications Research, University of South Australia, Mawson Lakes SA 5095, Australia

[2]Department of Signal Processing, Tampere University of Technology, 33101 Tampere, Finland

*Corresponding author

Email addresses:

HD: dadsy002@mymail.unisa.edu.au

AG: atanas.gotchev@tut.fi

KE: karen.egiazarian@tut.fi

Abstract

The discrete cosine transform (DCT) offers superior energy compaction properties for a large class of functions and has been employed as a standard tool in many signal and image processing applications. However, it suffers from spurious behavior in the vicinity of edge discontinuities in piecewise smooth signals. To leverage the sparse representation provided by the DCT, in this article, we derive a framework for the inverse polynomial reconstruction in the DCT expansion. It yields the expansion of a piecewise smooth signal in terms of polynomial coefficients, obtained from the DCT representation of the same signal. Taking advantage of this framework, we show that it is feasible to recover piecewise smooth signals from a relatively small number of DCT coefficients with high accuracy. Furthermore, automatic methods based on minimum description length principle and cross-validation are devised to select the polynomial orders, as a requirement of the inverse polynomial reconstruction method in practical applications. The developed framework can considerably enhance the performance of the DCT in sparse representation of piecewise smooth signals. Numerical results show that denoising and image approximation algorithms based on the proposed

framework indicate significant improvements over wavelet counterparts for this class of signals.

Keywords: sparse representation;, inverse polynomial reconstruction; discrete cosine transform; linear approximation; denoising.

1 Introduction

Many signal processing tasks take advantage of transforming the signal of interest into suitable transform domain where relevant information is concentrated into a small set of coefficients. This leads to so-called *sparse representations*, which are essential in applications such as approximation, compression [1], pattern recognition [2], and denoising [3]. Popular choices of representation bases and corresponding transforms have been the Fourier transform, the discrete cosine transform (DCT) [4], the wavelet transform [1], and more recent developments, such as curvelets [5,6], and other lets [1,7,8]. Among those transforms, the Fourier transform has been playing an important role, as the most classical one, for the conceptual interpretation of the transform domain coefficients in terms of *frequency* and *spectrum*.

The choice of suitable basis for achieving a sparse signal representation depends on the type of signal of interest, and its characteristics such as regularity and smoothness. In many cases, however, the type of the signal is not clear in advance and standard harmonic transformations (Fourier or cosine transform) are used to explore signals peculiarities. While harmonic transformations are good in terms of decorrelation and energy compaction for a large class of signals, they might also cause reconstruction artifacts. This effect is especially pronounced for the class of piecewise smooth signals, i.e. signals having smooth regions separated by (jump) discontinuities. For such signals, harmonic transformations generate spurious oscillations, in the vicinity of discontinuities at the reconstructed signals, commonly referred to as *Gibbs phenomenon*. The Gibbs phenomenon is the effect of using partial harmonic series for the reconstruction of a signal with discontinuities, where the overshoots and undershoots around the discontinuity do not disappear no matter how long the series is [9].

The problem of mitigating the Gibbs phenomenon has been thoroughly studied for the case of Fourier representation [10,11]. To overcome the Gibbs phenomenon in Fourier series expansion, in [12], a polynomial reconstruction method is proposed that successfully eliminates the oscillations with high accuracy. This is feasible through re-expanding the signal in a suitably chosen polynomial basis. In other words, the Fourier representation of the signal, which includes the Gibbs oscillations, is re-projected onto a polynomial basis, which is Gegenbauer polynomials in the case of Fourier series. Thus, this procedure is referred to as re-projection.

Roughly speaking, the re-projection procedure is a basis set transformation. The idea of a re-projection method is to determine the transformation between the two representations. In the *direct method* (originally termed as *Gegenbauer reconstruction method* in the literature) this transformation is realized as consecutive projections of the signal first onto the Fourier series basis and then onto the space of Gegenbauer polynomials [12,9]. Alternatively, the *inverse method*, also known as inverse polynomial reconstruction method (IPRM), reverses the order of projections, i.e. projects the signal first onto the polynomial basis and then onto the Fourier space [11,13,14]. It turns out that these two methods are not equivalent and the reconstruction procedure using the inverse method is not only more accurate, but also independent of the polynomial basis [11]. The latter is an important advantage of the inverse method, since there is no optimal way to select the parameter of the Gegenbauer polynomials in the direct method, automatically.

The formulation of IPRM for the case of Fourier series expansion has been studied in [11,13]. In [14] the extension of this framework to partial Fourier series expansion has been considered. Here we follow a similar formulation as the one given by [13] and take advantage of the results of [14] to derive the IPRM for the partial DCT expansion, since DCT has been adopted as a standard tool for signal and image compression and denoising, due to its excellent decorrelation properties. We specifically concentrate on the class of piecewise smooth signals where the exact locations of the discontinuity points are known *a priori* or can be estimated with suitable methods. For this class, we derive a framework for the IPRM in the DCT expansion. Limiting the order of polynomials and assuming the discontinuity locations are given, it leads to solving a

linear approximation problem. Applying this framework to both piecewise polynomial and piecewise smooth signals, we show that it is feasible to reconstruct the signals from a limited number of DCT coefficients, with high accuracy. This result can significantly improve the performance of the DCT in sparse representation of piecewise smooth signals.

In addition, we consider the problem of selecting the polynomial orders in IPRM, especially when the signal is given in the presence of noise. To this end, we use two model selection methods, namely minimum description length (MDL) [15] and cross-validation (CV) [16,17]. Finally, having been equipped with the DCT-based IPRM and polynomial order selection methods in IPRM, we will show that the use of this framework leads to efficient algorithms for denoising and linear approximation.

The article is organized as follows. In the next section, the derivation of the inverse method in DCT expansion is presented. In Section 3, we adopt order selection techniques to determine the polynomial orders in IPRM. Section 4 focuses on applications, namely, denoising and approximation. In Section 5, numerical experiments are presented, where the performance of the DCT-based IPRM is compared with other methods, especially the wavelet transform. Conclusions are given in Section 6.

2 DCT-based IPRM

Let $f(x_n)$ be the samples of a piecewise smooth signal on a uniform grid $\{x_n\}_{n=0}^{N-1}$ over the interval $\Omega = [-1, 1]$. Here we assume that the exact locations of the discontinuities (edges) are known and we have N_s smooth sub-intervals. Let $\Omega_i = \{x_n\}_{n=a_i}^{b_i}$ indicate the set of grid points of the ith smooth sub-interval, where $\{\Omega_i\}_{i=1}^{N_s}$ is a partition of Ω.[a] Furthermore, assume that n_i represents the cardinality of Ω_i, which is equal to $b_i - a_i + 1$. Designating f in each sub-interval Ω_i by f_i, $f(x_n)$ can be written as

$$f(x_n) = \sum_{i=1}^{N_s} f_i(x_n). \tag{1}$$

Note that

$$f_i(x_n) = \begin{cases} f(x_n) & x_n \in \Omega_i \\ 0 & \text{otherwise.} \end{cases}$$

The Gegenbauer polynomial expansion of f_i at the grid points is given by

$$f_i(x_n) = \sum_{\ell=0}^{\infty} g_\ell^i C_\ell^\lambda(X_i(x_n)), \tag{2}$$

where $C_\ell^\lambda(x)$ is the Gegenbauer polynomial of order ℓ with parameter λ ($\lambda > 0$), g_ℓ are the Gegenbauer polynomial coefficients. Note that since IPRM is independent of the polynomial basis [13], one may use other polynomial bases arbitrarily. Here we use Gegenbauer polynomials for consistency with the literature on IPRM. In addition, X_i is a linear map defined from $\{x_n\}_{n=a_i}^{b_i}$ to a uniform grid $\{y_n\}_{n=0}^{n_i-1}$ over the interval $[-1, 1]$ such that $y_0 = x_0$ and $y_{n_i-1} = x_{N-1}$:

$$X_i : \{x_n\}_{n=a_i}^{b_i} \to \{y_n\}_{n=0}^{n_i-1}. \tag{3}$$

Note that $\{x_n\}_{n=a_i}^{b_i}$ and $\{y_n\}_{n=0}^{n_i-1}$ are of the same cardinality. Approximating f_i with a polynomial of degree m_i yields:

$$f_i^{m_i}(x_n) = \sum_{\ell=0}^{m_i} g_\ell^i C_\ell^\lambda(X_i(x_n)). \tag{4}$$

Thus, polynomial approximation of f yields

$$f^m(x_n) = \sum_{i=1}^{N_s} \sum_{\ell=0}^{m_i} g_\ell^i C_\ell^\lambda(X_i(x_n)). \tag{5}$$

On the other hand, the DCT expansion of $f(x_n)$ is given by

$$\hat{f}_k = \beta[k] \sum_{n=0}^{N-1} f(x_n) \cos\left(\pi k \frac{(2n+1)}{2N}\right), \quad k = 0, \ldots, N-1, \tag{6}$$

where $\hat{f}_k$ are the DCT coefficients. Here we use DCT type 2, which is the most common form of DCT [4]. The normalization factors $\beta[k]$ are given by

$$\beta[k] = \begin{cases} \sqrt{\frac{1}{N}} & k = 0 \\ \sqrt{\frac{2}{N}} & k = 1, \ldots, N-1. \end{cases}$$

Given the N DCT coefficients $\{\hat{f}_k\}_{k=0}^{N-1}$, the signal $f(x_n)$ is obtained through the inverse DCT expansion as follows:

$$f(x_n) = \sum_{k=0}^{N-1} \beta[k] \hat{f}_k \cos\left(\pi k \frac{(2n+1)}{2N}\right), \quad n = 0, \ldots, N-1. \tag{7}$$

Reconstructing the signal using only the first N_d DCT coefficients results in the signal $f_{N_d}(x_n)$:

$$f_{N_d}(x_n) = \sum_{k=0}^{N_d-1} \beta[k] \hat{f}_k \cos\left(\pi k \frac{(2n+1)}{2N}\right), \quad n = 0, \ldots, N-1 \tag{8}$$

which suffers from the spurious oscillations. Instead, by substituting $f(x_n)$ in Equation (6) with $f^m(x_n)$ from Equation (5), we have

$$\begin{aligned} \hat{f}_k &= \beta[k] \sum_{n=0}^{N-1} \sum_{i=1}^{N_s} \sum_{\ell=0}^{m_i} g_\ell^i C_\ell^\lambda(X_i(x_n)) \cos\left(\pi k \frac{(2n+1)}{2N}\right), \quad k = 0, \ldots, N_d - 1, \\ &= \beta[k] \sum_{i=1}^{N_s} \sum_{\ell=0}^{m_i} g_\ell^i \sum_{n=a_i}^{b_i} C_\ell^\lambda(X_i(x_n)) \cos\left(\pi k \frac{(2n+1)}{2N}\right), \quad k = 0, \ldots, N_d - 1. \end{aligned} \tag{9}$$

By defining the matrix $\mathbf{W}^i$ for the ith sub-interval as

$$\mathbf{W}^i = [W_{k\ell}^i], \quad k = 0, \ldots, N_d - 1 \quad \text{and} \quad \ell = 0, \ldots, m_i, \tag{10}$$

with the matrix elements $W_{k\ell}^i$ given by

$$W_{k\ell}^i = \beta[k] \sum_{n=a_i}^{b_i} C_\ell^\lambda(X_i(x_n)) \cos\left(\pi k \frac{(2n+1)}{2N}\right), \tag{11}$$

Equation (9) becomes

$$\hat{f}_k = \sum_{i=1}^{N_s} \sum_{\ell=0}^{m_i} g_\ell^i W_{k\ell}^i. \tag{12}$$

This equation can also be written in matrix form as

$$\sum_{i=1}^{N_s} \mathbf{W}^i.\mathbf{G}^i = \hat{\mathbf{f}}, \tag{13}$$

where the vectors $\mathbf{G}^i$ and $\hat{\mathbf{f}}$ are

$$\mathbf{G}^i = [g_0, g_1, \ldots, g_{m_i}] \quad \text{and} \quad \hat{\mathbf{f}} = [\hat{f}_0, \ldots, \hat{f}_{N_d-1}]. \tag{14}$$

The constraint

$$\sum_{i=1}^{N_s} (1 + m_i) \leq N_d \tag{15}$$

is necessary, in order for the system of equations in (13) not to be under-determined [14]. To further simplify the representation of Equation (13), we define the matrix $\mathbf{W}$ and vector $\mathbf{G}$ as

$$\mathbf{W} = [\mathbf{W}^1 \mid \ldots \mid \mathbf{W}^{N_s}], \tag{16}$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}^1 \\ \vdots \\ \mathbf{G}^{N_s} \end{bmatrix}.$$

Thus Equation (13) can be written as

$$\hat{\mathbf{f}} = \mathbf{W}.\mathbf{G}, \tag{17}$$

which can be solved for the vector of Gegenbauer coefficients $\mathbf{G}$, through pseudo-inversion of $\mathbf{W}$

$$\mathbf{G} = \mathbf{W}^\dagger.\hat{\mathbf{f}}. \tag{18}$$

Equation (18) defines the IPRM for the discrete cosine transform. In addition to the condition in (15), since the signal is provided at a finite number of grid points, an extra set of conditions must be met in order for the matrix $\mathbf{W}$ not to be rank deficient:

$$1 + m_i \leq n_i, \quad i = 1, \ldots, N_s. \tag{19}$$

This is due to the fact that if a sub-interval Ω_i is chosen too small and thus the number of grid points in Ω_i is less than the polynomial order, i.e. $1 + m_i > n_i$, the polynomial coefficients cannot be determined uniquely.

3 Polynomial order selection

When applying IPRM, assuming that f is a smooth signal, the higher the order of the polynomial, the more accurate the reconstruction is. Specifically, if f is a polynomial of degree M, with $m \geq M$ the reconstruction is exact. As a consequence, choosing the polynomial order sufficiently large will result in an accurate reconstruction. However, one usually seeks the reconstruction of the signal using as few polynomial coefficients as possible within an acceptable accuracy (e.g. in approximation problem). Furthermore, when the signal samples are given in the presence of noise (e.g. in denoising problem), increasing the polynomial order causes over-fitting in the recovered signal. In other words, the reconstructed signal follows the random fluctuations, which are due to the noise present in the signal. Therefore, practical implementation of IPRM in many applications requires the choice of the polynomial order. We suggest employing two commonly used model selection methods, namely MDL and CV, for making such choices. Both model selection methods attempt to choose the best estimate, according to a certain criterion, among a given list of *models*. In general, this criterion is devised so as to make a compromise between a measure of goodness of fit and that of parametric complexity. The two model selection methods have been chosen as they are known to provide

nearly-unbiased estimates of model parameters and are effective in avoiding model over-fitting, which is essential for the proper selection of polynomial orders in the presence of noise.

In this section, we assume that each model $\mathcal{M}_j$ corresponds to a combination of the polynomial orders $\{m_i\}_{i=1}^{N_s}$ for different intervals. We further assume that the polynomial order in the ith interval is constrained by an upper bound M_i, i.e. $m_i \leq M_i$, $i \in \{1, \ldots, N_s\}$. As such, the number of polynomial order combinations (or models $\mathcal{M}_j$) to be chosen from is $\prod_{i=1}^{N_s} M_i$.

3.1 Minimum description length

According to MDL principle the best model for a given data set is the one that compresses the data the most and produces the shortest code length of the data. Suppose the noisy values of a piecewise smooth signal $f(x_n)$ are given:

$$y_n = f(x_n) + \epsilon_n, \quad n = 1, \ldots, N, \tag{20}$$

where the errors ϵ_n are i.i.d. Gaussian random variables with zero mean and unknown variance σ^2. The aim is to select the polynomial orders $\{\hat{m}_i\}_{i=1}^{N_s}$ with which IPRM results in the best estimate (in MDL sense) $\hat{\mathbf{y}} = f_N^{\hat{m}}$ of f. By the Gaussian distribution with unknown variance assumption, the MDL score of each model is given by [18]

$$\mathrm{MDL}(\mathcal{M}_j) = N \log\left(\frac{\mathrm{RSS}}{N}\right) + \log(N)\, K,$$

where *residual sum of squares* (RSS) is calculated as follows:

$$\mathrm{RSS} = \sum_{n=1}^{N} (y_n - \hat{y}_n)^2, \tag{21}$$

and K represents the number of free parameters in the model. Since in each interval there are $m_i + 1$, $i \in \{1, \ldots, N_s\}$ polynomial coefficients, so is the number of free parameters. Therefore, the total number of free parameters in the IPRM problem is given by $\sum_{i=1}^{N_s}(m_i + 1)$. Subtracting the constant term N_s from the

summation[b] yields $K = \sum_{i=1}^{N_s} m_i$. Hence, the MDL score for IPRM can be written as

$$\mathrm{MDL}(\mathcal{M}_j) = N \log\left(\frac{\mathrm{RSS}}{N}\right) + \log(N) \sum_{i=1}^{N_s} m_i. \tag{22}$$

Given a data set, competing models $\mathcal{M}_j$, $j \in \{1, \ldots, \prod_{i=1}^{N_s} M_i\}$ can be ranked according to their MDL scores, with the one having the lowest MDL being the best in MDL sense [15].

In addition, if the data is available in terms of DCT coefficients, instead of time domain data points, we can apply the MDL directly to the DCT coefficients to determine the polynomial orders. This will lead to the same results as in the time domain case, when all the DCT coefficients are employed. This is due to the fact that the DCT matrix is unitary and preserves L^2-norm of the data. In this case, the value of RSS in terms of DCT coefficients is

$$\mathrm{RSS} = \sum_{n=1}^{N} (a_n - \hat{a}_n)^2, \tag{23}$$

where a_i and $\hat{a}_i$ designate the DCT coefficients of the noisy and reconstructed signals, respectively. If only the first N_d terms of the DCT representation are employed in the reconstruction procedure, the MDL score can be obtained by

$$\mathrm{MDL}(\mathcal{M}_j) = N_d \log\left(\frac{\mathrm{RSS}}{N_d}\right) + \log(N_d) \sum_{i=1}^{N_s} m_i. \tag{24}$$

In this case, the RSS value is written as

$$\mathrm{RSS} = \sum_{n=1}^{N_d} (a_n - \hat{a}_n)^2. \tag{25}$$

3.2 Cross-validation

Similar to the previous section, suppose $\mathbf{y}$ is the vector of data and $\hat{\mathbf{y}}$ designates the reconstructed signal using IPRM with a combination of the polynomial orders $\{m_i\}_{i=1}^{N_s}$. If the estimate $\hat{\mathbf{y}}$ can be written in the form of

$$\hat{\mathbf{y}} = \mathbf{H}.\mathbf{y}, \tag{26}$$

where $\mathbf{H}$ is a matrix that does not depend on the data $\mathbf{y}$, the fitting method is called *linear* [16], and the matrix $\mathbf{H}$ is referred to as *hat matrix*. In *leave-one-out cross-validation* [17], for a linear fitting problem, the CV score can be obtained using the following equation:

$$\mathrm{CV}(\mathcal{M}_j) = \frac{1}{N}\sum_{n=1}^{N}\left(\frac{y_n - \hat{y_n}}{1 - H_{nn}}\right)^2, \tag{27}$$

where H_{nn} is the nth diagonal element of $\mathbf{H}$. This score is calculated for all the competing models $\mathcal{M}_j$, $j \in \{1, \ldots, \prod_{i=1}^{N_s} M_i\}$, and the best model is the one minimizing the CV score. By replacing the quantity $1 - H_{nn}$ in the denominator with its average $1 - \frac{1}{N}tr(\mathbf{H})$, one can obtain the *generalized cross-validation* score [17]:

$$\mathrm{GCV}(\mathcal{M}_j) = \frac{1}{N}\frac{\sum_{n=1}^{N}(y_n - \hat{y_n})^2}{\left(1 - \frac{1}{N}tr(\mathbf{H})\right)^2}. \tag{28}$$

Since IPRM can be expressed in the form of a linear fitting model, we can take advantage of Equations (27) and (28) to calculate CV and GCV scores, respectively. The hat matrix of IPRM is given by

$$\mathbf{H} = \mathbf{P}.\mathbf{W}^{\dagger}.\mathbf{D}, \tag{29}$$

where $\mathbf{W}$ is the transformation matrix of IPRM from Equation (16); $\mathbf{D}$ and $\mathbf{P}$ are the DCT and polynomial basis matrices, respectively. The polynomial basis matrix $\mathbf{P}$ is the direct sum of the polynomial basis matrices of sub-intervals $\mathbf{P}_i$, i.e.

$$\mathbf{P} = \bigoplus_{i=1}^{N_s}\mathbf{P}_i \tag{30}$$

$$= \begin{bmatrix} \mathbf{P}_1 & & & 0 \\ & \mathbf{P}_2 & & \\ & & \ddots & \\ 0 & & & \mathbf{P}_{N_s} \end{bmatrix}. \tag{31}$$

The rationale behind derivation of the hat matrix of IPRM in the form given in Equation (29) is as follows. From Equation (18) the polynomial coefficients are related to DCT coefficient via $\mathbf{G} = \mathbf{W}^{\dagger} \cdot \hat{\mathbf{f}}$. Replacing $\hat{\mathbf{f}}$ by $\mathbf{D} \cdot \mathbf{y}$ results in $\mathbf{G} = \mathbf{W}^{\dagger} \cdot \mathbf{D} \cdot \mathbf{y}$. Finally, left matrix multiplication of both sides by the polynomial basis matrix given in Equation (30) results in the time domain reconstructed signal $\hat{\mathbf{y}}$, i.e. $\hat{\mathbf{y}} = \mathbf{P} \cdot \mathbf{G} = \mathbf{P} \cdot \mathbf{W}^{\dagger} \cdot \mathbf{D} \cdot \mathbf{y}$.

Similar to MDL, if the signal is provided in terms of its DCT coefficients, by making use of the unitary property of the DCT matrix, it is feasible to find the GCV score[c] as follows:

$$\mathrm{GCV}(\mathcal{M}_j) = \frac{1}{N} \frac{\sum_{n=1}^{N} (a_n - \hat{a_n})^2}{\left(1 - \frac{1}{N} tr(\mathbf{H})\right)^2}. \tag{32}$$

Moreover, if only a part of DCT coefficients (first N_d-terms) are used for the reconstruction, $\mathbf{D}$ is the first N_d rows of the DCT matrix. In this case, the GCV score is given by

$$\mathrm{GCV}(\mathcal{M}_j) = \frac{1}{N_d} \frac{\sum_{n=1}^{N_d} (a_n - \hat{a_n})^2}{\left(1 - \frac{1}{N_d} tr(\mathbf{H})\right)^2}. \tag{33}$$

3.3 Numerical experiments

In this section, we assess and compare the performance of the order selection methods introduced above. Root mean square error (RMSE) is used as a quantitative measure for comparison purposes:

$$\mathrm{RMSE} = \| f_N^{\hat{m}}(x_n) - f(x_n) \|_2. \tag{34}$$

Here we use two test signals; a piecewise polynomial and a piecewise smooth signal. The first test signal is given by

$$f(x) = \begin{cases} -1 - x & x \leq 0 \\ (1-x)^5 & x > 0. \end{cases}$$

The sample size is set at $N = 256$. The signal-to-noise ratio (SNR) is changed from 10 to 20 dB (which in linear scale corresponds to SNR = 3.16 to 10) with step size of 1 dB. For each SNR, 100 sets of noisy data

were simulated and for each SNR, RMSE is averaged over all the data sets. The polynomial order in each interval is iterated from $m_i = 1$ to 10. For each combination of orders, we perform IPRM to reconstruct the signal. For each reconstructed signal, we calculate the RSS given in Equation (21). The values of RSS and polynomial orders are used to compute the MDL and CV scores given in Equations (22) and (28), respectively. In each case, the minimizer of the resulting score is chosen as the combination of estimated orders, which is used to compute RMSE of the reconstructed signal. The plot of variations of the RMSE values versus SNR is depicted in Figure 1a, in which we compare the RMSE performance of MDL and CV with that of the "oracle", which is basically the IPRM reconstruction with the optimal orders so as to minimize the RMSE criterion. The orders employed in the oracle method are obtained using the original signal f.

We repeat the same experiment for the heavisine signal, using the same set of parameters. As demonstrated in Figure 1b, at low SNR values MDL outperforms CV, whereas at higher SNR values (larger than around 12 dB) CV exhibits a better performance compared to MDL.

The simulation results of this section suggest that the performance of the two model selection methods differs slightly depending on both the signal and SNR level.

4 Applications

In this section, we concentrate on denoising and linear approximation as two main applications for which DCT has been applied successfully.

4.1 Denoising

Applying the framework derived in Section 2, in this section we show that inverse reconstruction method offers excellent de-noising properties. As a result, a new de-noising algorithm for piecewise smooth signals using inverse method in the DCT domain is presented. This de-noising algorithm is particularly applicable

when the signal is provided in terms of its first N_d DCT coefficients, where recovering the signal using ordinary inverse DCT would result in spurious oscillations in the reconstructed signal.

Suppose f is a piecewise smooth discrete signal. The problem is formulated as follows: one observes $y_n = f(x_n) + e_n$ where the errors e_i are independent and identically distributed zero-mean Gaussian random variables. Alternatively, the observation may be available in terms of the first N_d DCT coefficients of a noisy signal. The original signal is supposed to be deterministic and independent of the noise. Our goal is to find an estimate of the original signal using the DCT IPRM framework derived in Section 2.

The proposed de-noising algorithm works in the following way:

(1) Compute the DCT coefficients of the noisy signal y. (This step is omitted in the case the noisy signal is given in terms of its DCT coefficients.)
(2) Find the regions of smoothness in the signal. Here we employ edge detection techniques based on concentration factors for noisy data [19], which works on Fourier coefficients of the signal. The Fourier coefficients are found from either the noisy signal or through a transformation on the DCT coefficients of the signal.
(3) Find the best polynomial orders for each interval using the order selection techniques introduced in Section 3.
(4) Reconstruct the signal using DCT-based IPRM with the polynomial orders calculated in step 3.

The performance and accuracy of this algorithm depends on the following factors: the number of DCT coefficients utilized in IPRM procedure, precise knowledge of the intervals of smoothness in the signal (i.e. edge detection), and the polynomial orders determined as a result of order selection techniques. In the sequel, we investigate the effect of the number of DCT coefficients used for reconstruction.

Let f be a piecewise smooth discrete signal of length N. Taking DCT of f results in N DCT coefficients. We consider applying IPRM on f when the first N_d ($N_d \leq N$) DCT terms of f are used for reconstruction. When the signal is noise free, the number of DCT coefficients N_d is required to satisfy the minimum requirement given by inequality (15).

In order to examine the accuracy of the reconstruction, as the number of DCT coefficients varies, we consider two examples. The first example is a piecewise polynomial signal of length $N = 256$ given by

$$f_1(x) = \begin{cases} -1 - x & x \leq 0 \\ (1-x)^6 & x > 0, \end{cases}$$

which is illustrated in Figure 2a. The DCT spectrum of f_1 is plotted in Figure 2b. Figure 3a depicts the RMSE of the reconstructed signal as a function of the number of DCT coefficients N_d. As can be observed in this figure, although the signal f_1 has more significant DCT coefficients than the lower bound specified by inequality (15), the reconstruction is exact (within machine accuracy) and RMSE is of order 10^{-14}, when the above constraint is met.

Next, we consider *Heavisine* signal f_2 of the same length, which is piecewise smooth and is depicted in Figure 4a. From the RMSE graph of this piecewise smooth signal, it is clear that the higher the number of DCT coefficients used in IPRM, the more accurate the reconstructed signal is. Note that since f_2 is not a piecewise polynomial signal, the error does not converge to zero.

Subsequently, we examine the RMSE performance of the above signals in the presence of noise at different number of DCT coefficients. We assume the signal-to-noise-ratio of the signals to be SNR $= 10\,\text{dB}$. For each signal, we conduct a Monte Carlo experiment of 100 sets of noisy data, and average the RMSE result over all data sets. The resulting RMSE for signals $f_1(x)$ and $f_2(x)$ are demonstrated in Figure 5.

For the piecewise polynomial signal $f_1(x)$, when N_d is equal or close to the minimum requirement, the RMSE of the reconstruction is high compared to the case where higher number of DCT coefficients is used. Above this range, including more DCT coefficients in IPRM, does not improve the reconstructed signal. For the piecewise smooth signal $f_2(x)$, we virtually encounter the same scenario as in the noise free case. In other words, as we increase the number of DCT coefficients used for IPRM, we achieve a more accurate reconstruction.

4.2 Approximation

In [13], the extension of the one-dimensional (1D) inverse method to images has been studied, where they have shown that when using the tensor product of the 1D case, the transformation matrix in the two-dimensional (2D) case, even for sub-domains with a simple geometry, easily becomes singular. Therefore, they have suggested to instead apply IPRM on images in a slice-by-slice[d] manner [13], which is not efficient for approximation purposes.

One way to overcome the restriction of IPRM in 2D scenarios, is to convert the image into a 1D signal. The aim is to find a path through all data points of the image such that the resulting signal is as smooth as possible, while considering the costs required for storing the path data. In this way we can apply IPRM on the resulting 1D signal efficiently and achieve a sparse DCT representation of the image.

Recently, an adaptive wavelet transform, referred to as Easy Path Wavelet Transform (EPWT),[e] was proposed in [20] for a sparse representation of images. The EPWT works along pathways of image data points and exploits the correlations among neighboring pixels. In a nutshell, it consists of two parts: first, finding the pathways according to the difference in value among the neighboring pixels and second, applying a 1D wavelet transform along the resulting pathways. Here, we refer to the first part as *easy path algorithm*.

The EPWT algorithm is shown to be highly efficient for representation of real-world images, outperforming the tensor-product wavelet transform. However, in this article we show that applying IPRM on the result of the easy path algorithm can improve the approximation performance for piecewise smooth images, significantly.

The easy path algorithm works as follows. Let $f(i,j)$ be a discrete image of size N_1-by-N_2 and $I = \{(i,j) : i = 1,\dots,N_1, j = 1,\dots,N_2\}$ be the corresponding index set. We are interested in forming the path vector $\mathbf{p}$, which is a permutation of the 1D representation of the index set, i.e. $(1,\dots,N_1N_2)$. Having determined one point of the image as an element of the path vector $\mathbf{p}(\ell)$, we seek the neighborhood of this point for the neighbor that minimizes the absolute difference $|f(\mathbf{p}(\ell)) - f(\mathbf{p}(\ell+\mathbf{1}))|$ of the function values.

The neighborhood of an index $(i, j) \in I$ is defined by

$$N(i,j) = \{(i_1, j_1) \in I - \{(i,j)\} : |i - i_1| \leq 1, |j - j_1| \leq 1\}. \tag{35}$$

According to this definition, the indices located at a vertex and at a boundary but not a vertex, have three and five neighbors, respectively. Otherwise, $N(i, j)$ comprises eight elements. However, considering that the path passes through every point exactly once, it is necessary to exclude the indices that already have been used in the path.

We carry on this procedure as long as the set of admissible neighbors is not empty. This sequence of neighboring points is referred to as a *pathway*, and the union of pathways forms a *path* through the index set. Moreover, the transition from one pathway to the next in the path is called an *interruption*.

In the case of an interruption, we need to pick the next index $\mathbf{p}(\ell + 1)$ from the remaining free indices in the index set. Among different possibilities to start the next pathway [20], we select the index minimizing the absolute difference of the function values.

In addition, it can happen that we encounter multiple choices for the next index in the sense that more than one neighbor attain the minimum cost given by the absolute difference criterion. In [20], it is suggested to fix a favorite direction in advance to find a unique pathway. However, we propose a modified procedure to not only avoid the occurrences of interruptions the most, but also to reduce the number of edges in the path substantially. The latter consequence is particularly important in IPRM, since we shall rely on the edge locations in the reconstruction process.

The proposed modification in the easy path algorithm is as follows. Suppose the recent element of the path is $\mathbf{p}(\ell)$ and the easy path algorithm is to choose the next element of the path $\mathbf{p}(\ell+1)$ out of K potential indices $\{(i_k, j_k)\}_{k=1}^{K}$, tied at the minimum of the absolute difference criterion. For each of such indices, we count the number of choices for $\mathbf{p}(\ell + 2)$, if it were chosen as the next index $\mathbf{p}(\ell + 1)$ in the path. In other words, we find the number of degrees of freedom for the path element $\mathbf{p}(\ell + 2)$. Obviously, the index with minimum degrees of freedom is more susceptible to interruption in the future, if not selected in the path at

this step of the algorithm. For instance, indices located at a boundary or the vicinity of an edge naturally have less degrees of freedom and are more likely to form a singleton pathway. As a result, we give priority to the index with the least degrees of freedom, and choose it as element $\mathbf{p}(\ell+1)$ of the path.

Example 1 Here we consider a toy example to illustrate how the proposed modification of the easy path algorithm works. We consider a lower triangle of a simple linear image [f] of size 6×6. We use 1D numbering to show the direction of the path. Applying the easy path algorithm, without and with the modification of the algorithm (as illustrated in Figures 6a and 6b respectively), the path vector reads

$$\mathbf{p}_1 = \{1, 3, 5, 9, 14, 21, 13, 29, 12, 18, 11, 17, 10, 16, 15|6, 7, 4, 5, 2|8\},$$

$$\mathbf{p}_2 = \{1, 3, 5, 9, 14, 21, 20, 13, 8, 5, 2, 4, 7, 12, 18, 17, 11, 6, 10, 16, 15\},$$

where the sign | indicates interruption in the path. Note that the path $\mathbf{p}_1$ not only has two interruptions but also incorporates more edges along the pathways, since it changes the gray level values more often compared to the path $\mathbf{p}_2$. For instance, the transition $21 \longrightarrow 13$ in $\mathbf{p}_1$ is due to the fact that in the original version of the easy path algorithm, when there are two options, the one which is closer to the favorite direction is taken, where the favorite direction is the same as the previous direction of the path, which is to the right. Since, neither of the options are on the right, starting from the favorite direction, the pixel which is first, turning clockwise, is chosen as the next index in the path. On the other hand, in $\mathbf{p}_2$ the transition $21 \longrightarrow 20$ is chosen, since if the index 20 is chosen, it offers one option for the next index in the path, i.e. index 13, whereas if the index 13 is chosen, it offers two options for the next index in the path, i.e. indices 8 and 20.

Considering that the storage cost of the path vector is not negligible in comparison with that of the approximation coefficients, efficient coding schemes are required to reduce the cost of storing this vector. Here, we employ the coding strategy introduced in [20]. Since each pathway connects neighboring indices, one can store the direction of the next element rather than the index itself. Obviously, having eight neighbors

at most, the direction of the next index can be represented by three bits in the worst case. The coding algorithm that maps the vector of path indices $\mathbf{p}$ into the vector $\tilde{\mathbf{p}}$ works in the following steps:

(1) Let $\tilde{\mathbf{p}}(0) = 0$ since $\mathbf{p}(0) = 0$ is the starting point.

(2) For finding $\tilde{\mathbf{p}}(1)$, look clockwise through the neighbors of $\mathbf{p}(0)$ and insert

$$\tilde{\mathbf{p}}(1) = \begin{cases} 0 & \mathbf{p}(1) = N_1 \\ 1 & \mathbf{p}(1) = N_1 + 1 \\ 2 & \mathbf{p}(1) = 1. \end{cases}$$

(3) Having found the path code $\tilde{\mathbf{p}}(\ell)$, the subsequent path code $\tilde{\mathbf{p}}(\ell+1)$ is zero if it follows the previous direction of the path. More precisely, if

$$\tilde{\mathbf{p}}(\ell+1) = 2\tilde{\mathbf{p}}(\ell) - \tilde{\mathbf{p}}(\ell-1), \tag{36}$$

then $\tilde{\mathbf{p}}(\ell+1) = 0$. This direction is regarded as the *favorite direction* in the path. On the other hand, if the direction of the path is changed, then look clockwise through the admissible neighbors (starting with the favorite direction) and determine $\tilde{\mathbf{p}}$ accordingly.

Due to the fact that each index appears only once in the path, as we proceed with the algorithm, the number of admissible neighbors decreases and the coding algorithm tends to assign smaller codes to $\tilde{\mathbf{p}}$ [21]. This in turn results in a lower entropy of the path information.

From the coding strategy described above, it is clear that for a better storage of the path vector, it is preferable to continue the pathway in the direction which leads to smaller values of the path code $\tilde{\mathbf{p}}$. However, this is in contrast with the procedure explained for the multiple options scenario. In other words, there is a trade-off between the entropy of the pathways and the number of edges and interruptions in the path. This can be tackled by making a compromise through setting a predetermined upper bound θ on the number of degrees of freedom. More precisely, if the minimum number of degrees of freedom is lower than or

equal to θ, the neighbor with the minimum degrees of freedom is selected as $\mathbf{p}(\ell+1)$. Otherwise, the next index $\mathbf{p}(\ell+1)$ is the one closest to the favorite direction.

The proposed approximation scheme using DCT-based IPRM and the easy path algorithm works in the following way. First, we apply the easy path algorithm with the settings described above, in order to find the vector path $\mathbf{p}$ through the image. Having determined the 1D signal $f(\mathbf{p})$, we compute the DCT expansion of $f(\mathbf{p})$. Next, we find the discontinuity locations, using the edge detection from spectral data algorithm suggested in [22]. Specifically, we employ trigonometric concentration factors accompanied with the nonlinear enhancement algorithm proposed in [23]. The Fourier coefficients are found either from the time domain signal itself or through a transformation from the DCT coefficients of the signal [24]. The K-term approximation is obtained by storing the first K DCT coefficients, the path vector $\mathbf{p}$ and the edge locations.

The reconstruction of f from $\{\hat{f}_k\}_{k=1}^{K}$ is given in the following steps:

(1) Find the polynomial orders using the model selection methods explained in Section 3 (employing either MDL or CV through Equations (24) and (33), respectively).
(2) Apply DCT-based IPRM, with the polynomial orders determined in step 1, to obtain the approximate signal f_p.
(3) Apply the permutation $f(\mathbf{p}) = f_p$ in order to recover the function f (i.e. reciprocate the operation of the easy path algorithm).

In order to be able to compare the storage costs of our scheme with that of EPWT, we need to review how its algorithm works. In EPWT, after finding the complete path vector $\mathbf{p}$, a 1D wavelet transform is applied to the vector of function values along the path. As a result of a one-level decomposition, low-pass and high-pass wavelet coefficients are derived. The down-sampled low-pass part is used to form a low-pass filtered image with the same size as the original image and consisting of pairs of identical data points. The easy path algorithm is applied to the resulting image to find a path through the pairs of data points. The wavelet transform is again applied to the resulting path vector. The same strategy is repeated in further

levels up to L decomposition levels, the extent of which depends on the data. After a sufficient number of iterations, a shrinkage procedure is applied to the wavelet coefficients at different decomposition levels. In order to reconstruct the image, one needs the wavelet coefficients, location of the wavelet coefficients (since it is a nonlinear approximation scheme), and the path vectors in each iteration step. According to this algorithm, the length of the path vector at decomposition level ℓ is $\frac{N_1 N_2}{2^{\ell-1}}$. As a result, the cost of storing the path indices for the further levels usually doubles that of the first decomposition level [20].

5 Numerical experiments

5.1 Denoising

In this section, we analyze and assess the performance of the de-noising algorithm, introduced in Section 4. RMSE is used as a quantitative measure for comparison purposes:

$$\text{RMSE} = \| f_N^{\hat{m}}(x_n) - f(x_n) \|_2. \tag{37}$$

Here, we use all the DCT coefficients of the signal for reconstruction in IPRM, following the discussion on the effect of the number of coefficients N_d, presented in the previous section. Recall that for piecewise polynomial signals a wide range of N_d values produce identical results, whereas for piecewise smooth signals, as we increase the number of coefficients N_d, the reconstructed signal improves gradually.

We compare the result of our algorithm with the following de-noising methods:

(1) Translation-invariant (TI) wavelet de-noising (cycle-spinning) [25]

(2) De-noising based on wavelet transform footprints [26, 27]

(3) Direct method [28, 29]

The first test signal is a piecewise constant signal given in Figure 7, which is known as *Blocks* in de-noising literature. We use the same specifications (length of the signal $N = 2,048$, SNR $= 7$) as those provided

in *Wavelab* [30]. For this signal, cycle-spinning gives the best result in terms of RMSE, when Haar wavelet is employed. When applying the wavelet footprints algorithm, it is assumed that the signal is piecewise constant. We also consider the case where polynomial orders are not known. Considering the de-noising procedure for the direct method, in [28] it is suggested to use the following formula:

$$\lambda_i = m_i = \epsilon_i \, \beta \, N, \tag{38}$$

for the ith interval $[a_i, b_i]$, where $\beta = \frac{2\pi}{27} \approx 0.2327$ and $\epsilon_i = \frac{b_i - a_i}{2}$. In case the result of this formula is not an integer, $\lceil m_i \rceil$ is considered as the polynomial order [9]. However, the polynomial orders obtained through this formula do not result in an acceptable reconstruction of the signal. Here, we assume that the signal is piecewise constant (thus setting the polynomial orders to zero) and find the parameters λ_i from Equation (38). In the case of our algorithm, we consider two scenarios for the polynomial orders: first, we use an oracle method, where an oracle tells us which are the best polynomial orders (zero order polynomials for Blocks signal). Second, we use one of the model selection methods introduced in Section 3, to find the polynomial orders automatically. Here, we use MDL for polynomial order selection.

The simulation results in terms of RMSE for one sample noisy signal are given in Table 1. According to this table, de-noising using IPRM significantly outperforms other de-noising algorithms. Note that Table 1 indicates the same RMSE values for IPRM with and without oracle. This is attributed to the fact that the MDL has estimated the correct polynomial orders. Figure 7 shows the reconstructed signals using different de-noising algorithms.

The second test signal is a piecewise polynomial signal given by

$$f(x) = \begin{cases} -1 - x & x \leq 0 \\ (1-x)^2 & x > 0. \end{cases}$$

The simulation results in terms of RMSE for one sample noisy signal are given in Table 2. Figure 8 illustrates the reconstructed signals using different de-noising algorithms.

The next test signal is a piecewise smooth signal, referred to as *Heavisine* in de-noising literature. The RMSE results of different de-noising algorithms are given in Table 3. We use MDL for selecting the polynomial orders, which are $m_1 = 6$, $m_2 = 6$, and $m_3 = 5$. In case of the direct method, we set the parameter $\lambda = 1$, which was found by trial and error, since the values found through the above formula do not lead to an acceptable reconstruction of the signal. Figure 9 shows the reconstructed signals using different de-noising algorithms at SNR $= 16.91$ dB.

The last experimental signal is *Doppler* signal, as illustrated in Figure 10a. The simulation results in terms of RMSE for a noisy signal with SNR $= 16.91$ dB are given in Table 4. Figure 10 illustrates the reconstructed signals using different de-noising algorithms. Here we use CV to find the polynomial order. The resulting polynomial order is $m = 207$ (oracle gives $m = 203$) which is relatively high, and causes over-fitting at the low frequency section of the signal. However, this is inevitable due to the nature of Doppler signal, which requires a high polynomial order for the high frequency part, and a low polynomial order for the low frequency one. One way to further improve the performance of IPRM algorithm for this signal is to divide it into two parts successively and find the polynomial orders for the resulting sub-intervals, separately. After one step of such division, it is feasible to reduce the RMSE of the recovered signal from 13.6757 to 11.1579. In this case, the polynomial orders used in the IPRM procedure are $m_1 = 140$, and $m_2 = 15$, respectively. As depicted in Figure 10f, the accuracy of the recovered signal is significantly improved in the right sub-interval of the signal.

In order to investigate the performance of the denoising algorithms over a wider range of SNR values, here we present Monte Carlo experiments for the blocks and $f(x)$ signals at different values of SNR. In the case of the blocks signal, 100 sets of noisy signals are simulated, where we vary SNR from 10 to 20 dB by a step-size of 2 dB. The graph in Figure 11a illustrates the average RMSE results for different algorithms. For the de-noising algorithm based on wavelet footprints, two scenarios are considered: first, the signal is assumed to be piecewise constant, i.e. polynomial orders are known in advance or given by an oracle, and second, polynomial orders are not known in advance. For de-noising based on the direct method, we assume

that the polynomial orders are known. In the case of IPRM, we find the polynomial orders using the MDL algorithm. In addition, Figure 11b demonstrates the RMSE performance of the denoising algorithms with the same set of parameters for the piecewise smooth signal $f(x)$.

5.2 Approximation

In this section we present some numerical examples of the proposed approximation scheme applied to gray-scale piecewise-smooth images. We compare the performance of the DCT-based IPRM with the approximation results of both EPWT and the tensor product wavelet transform using Haar, Daubechies db4 and 9/7 bi-orthogonal filter banks. This comparison is carried out in terms of the number of the coefficients used for the sparse representation, storage costs and the SNR given by

$$\text{SNR} = 20 \log \left(\frac{\|f\|_2}{\|f - \tilde{f}\|_2} \right), \tag{39}$$

where $\|.\|_2$ denotes the L^2-norm, and f and $\tilde{f}$ are the original and reconstructed sparse images, respectively.

The storage cost of our algorithm consists of the costs of storing the coefficients, edge locations and the path information. The entropy of the path is composed of the entropy of the pathways and the cost of storing the location of interruptions. The entropy of the pathways is given by

$$\text{entropy} = -\sum_{i=0}^{n-1} \frac{h_i}{N} \log_2 \left(\frac{h_i}{N} \right), \tag{40}$$

where $n = 8$ is the number of possible options in the coded path vector $\tilde{\mathbf{p}}$ and $h_0, \ldots, h_{n-1}$ designate the frequencies of their occurrence. $N = N_1 \times N_2$ stands for the total number of coefficients. Moreover, the cost of storing the location of either an edge or an interruption is considered as $\log_2(N)$.

In order to roughly compare the storage costs of the proposed approximation algorithm with that of EPWT and the tensor product wavelet transform, the following simplified scheme is assumed [20,31]. For the tensor product wavelet transform, the cost of storing the non-zero coefficients as well as that of encoding

their positions constitute the storage cost. Here, we assume that the parameter b represents the number of bits used for encoding of one wavelet coefficient. In addition, the cost of encoding the position of M non-zero wavelet coefficients is calculated through $-\frac{M}{N}\log_2\frac{M}{N}-\frac{N-M}{N}\log_2\frac{N-M}{N}$. In the case of the EPWT, the storage cost of the path is also required to be taken into account.

In the first example we consider the Shepp-Logan phantom image of size 256×256, as depicted in Figure 12a. To gain an insight into the effect of the parameter θ, in Table 5, we have compared the number of edges in the outcome of the easy path algorithm for different values of θ. This table reflects the trade-off between the entropy of the path and the number of edges that appear in the resulting signal. Throughout this section, we favor the choice $\theta = 2$, so as to avoid the occurrence of singleton pathways, while keeping the entropy of the path as small as possible.

In Table 6, we present some approximation results of our algorithm, EPWT and tensor product wavelet transform, for the phantom image. In this table, we have obtained the storage costs for both $b = 8$ and $b = 16$, and the number of iterations for EPWT and tensor product wavelet transform are considered $L = 9$ and $L = 5$, respectively, which usually results in the best approximation scenario for the phantom image. Please note that as long as the number of both iterations and coefficients are constant, cost of the path in EPWT is independent of the choice of the wavelet transform.

According to Table 6, as expected, the performance of both DCT-based IPRM and EPWT is especially superior to tensor product wavelet transform for higher values of the parameter b. In addition, the DCT-based IPRM algorithm outperforms the wavelet-based algorithms, in terms of the storage costs as well as the SNR of the approximation result. Figures 12b and 13a illustrate the approximation results of tensor product wavelet transform and EPWT, respectively, using the Haar transform with the specifications provided in Table 6. In addition, Figure 13b shows the reconstruction result of the proposed DCT-based IPRM algorithm with 175 approximation coefficients.

For the second toy example, we consider the piecewise linear image, shown in Figure 14a. Akin to the previous example, we compare the approximation results of the proposed algorithm with EPWT and tensor

product wavelet transform, as depicted in Table 7. As can be observed from this table, the DCT-based IPRM algorithm significantly reduces the number of coefficients as well as the storage costs of the approximation, at similar SNR values. The reconstruction results of this image using tensor product wavelet transform and EPWT, using Haar filters, are illustrated in Figures 14b and 15a, respectively. The 40-term approximation of the piecewise linear image using the proposed algorithm, is shown in Figure 15b.

Finally, we consider the approximation of disparity map images. Such images encode the shifts between stereoscopic views of the same scene and can serve for determining the depth map associated with a certain view [32]. The color view (Figure 16a) is augmented by disparity map (Figure 16b) encoded in gray-scale levels. The comparison of the approximation performance of the proposed algorithm with that of EPWT and tensor product wavelet transform is given in Table 8. In addition, the 130-term approximation of the DCT-based IPRM (as depicted in Figure 17) offers a similar SNR to 500-term and $3,000$-term approximations using EPWT and tensor product wavelet transform (using Haar), with less storage cost. The approximation results of tensor product wavelet transform and EPWT using Haar are depicted in Figures 18a and 18b, respectively.

6 Conclusions

In this article, a mathematical formulation for the IPRM in the DCT expansion has been derived. The proposed framework aims at overcoming the oscillatory behavior of a signal with discontinuities reconstructed from a limited number of its DCT coefficients. It also leads to a sparser representation if the underlying signal is piecewise smooth. Assuming the discontinuity locations are known and limiting the polynomial order for each smooth piece, the framework essentially leads to the solution of a linear approximation problem. Furthermore in the article, denoising and linear approximation algorithms for piecewise-smooth signals based on their DCT-based IPRMs have been developed. Simulation results have demonstrated significant improvements over wavelet-based methods, namely, TI wavelet denoising, denoising based on wavelet transform footprints, as well as recently introduced EPWT based method.

The linear approximation problem of re-projection of a signal represented in one basis onto another basis as addressed in this article, correlates with the problem considered in the compressive sensing (CS) literature [33]. As the CS theory postulates, if a signal is known to be sparse with respect to a certain dictionary of atoms, it can be reconstructed exactly from measurements in another dictionary, incoherent with the first one. In our case, the sparse representation is sought based on the (Gegenbauer) polynomials while the measurements are taken with respect to the dictionary of DCT atoms. The CS formalism may lead to more general results, i.e. when the discontinuity locations are not given or when the DCT coefficients used for reconstruction are selected arbitrary rather than taking the first K terms. Such results might be quite beneficial for e.g. compression and are object of our future research.

Competing interests

The authors declare that they have no competing interests.

7 Author's contribution

All authors read and approved the final manuscript.

Acknowledgment

This work was supported by the Academy of Finland, Project No. 213462 (Finnish Centre of Excellence program 2006–2011). The authors thank Prof. Jae-Hun Jung for stimulating discussions on IPRM. Thanks to Prof. Anne Gelb and Prof. Shlomo Engelberg for helpful comments on edge detection algorithms. Furthermore, the authors would like to thank Dr. Pier Luigi Dragotti for providing us the codes of Wavelet Footprints, and Prof. Gerlind Plonka for providing us a version of codes for the EPWT algorithm.

Endnotes

[a]According to the notation used here, each transition from x_{b_i} to $x_{a_{i+1}}$ corresponds to an edge. [b]This operation is permissible since the constant term appears in the MDL score of all the competing models. [c]We employ GCV (which is a close approximation of CV) instead of CV, in order to make use of unitary property of the DCT matrix and extend the result to the case where the signal is provided in terms of its DCT coefficients. [d]In the slice-by-slice approach, the 1D IPRM is applied to each row and each column of the image independently. In other words, the image is considered as a collection of independent 1D signals. [e]Here we use the so-called *rigorous* version of EPWT, since we are seeking a 1D representation of the image, which is as smooth as possible. This, in turn, will reduce the number of edges in the resulting signal. [f]In other words, the hypotenuse is assumed to be adjacent to an edge. This toy example is chosen just to show the effectiveness of the proposed modification in the vicinity of edges and boundaries of an image.

References

1. S Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. (Academic Press, Orlando, 2008)
2. J Wright, Y Ma, J Mairal, G Sapiro, T Huang, S Yan, Sparse representation for computer vision and pattern recognition, *the Proceedings of the IEEE*, vol. 98, 1031-1044 (2010)
3. AK Fletcher, S Rangan, VK Goyal, K Ramchandran, Denoising by sparse approximation: error bounds based on rate-distortion theory. EURASIP J. Appl. Signal Process. **2006**, 19 (2006)
4. KR Rao, P Yip, V Britanak, *Discrete Cosine Transform: Algorithms, Advantages, Applications* (Academic Press, Orlando, 2007)
5. EJ Cands, DL Donoho, New tight frames of curvelets and optimal representations of objects with piecewise c singularities. Commun. Pure Appl. Math. **57**, 219–266 (2004)
6. J-L Starck, EJ Candes, DL Donoho, The curvelet transform for image denoising. IEEE Trans. Image Process. **11**(6), 670–684 (2002)
7. DL Donoho, Wedgelets: nearly-minimax estimation of edges. Ann. Stat. **27**, 859–897 (1999)

8. EL Pennec, S Mallat, Bandelet image approximation and compression. SIAM J. Multiscale Model. Simul. **4**(4), (2005)

9. D Gottlieb, C-W Shu, On the Gibbs phenomenon and its resolution. SIAM Rev. **39**(4), 644–668 (1997)

10. JS Hesthaven, S Gottlieb, D Gottlieb, in *Spectral Methods for Time-Dependent Problems* ed. by G Anastassiou. Cambridge Monographs on Applied and Computational Mathematics, ch. 9 (Cambridge University Press, Cambridge, 2007), pp. 153–187

11. BD Shizgal, J-H Jung, Towards the resolution of the Gibbs phenomena. J. Comput. Appl. Math. **161**(1), 41–65 (2003)

12. D Gottlieb, C-W Shu, A Solomonoff, H Vandeven, On the Gibbs phenomenon I: recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function. J. Comput. Appl. Math. **43**(1–2), 81–98 (1992)

13. J-H Jung, BD Shizgal, Inverse polynomial reconstruction of two dimensional Fourier images. J. Sci. Comput. **25**(3), 367–399 (2005)

14. T Hrycak, K Gröchenig, Pseudospectral Fourier reconstruction with the modified inverse polynomial reconstruction method. J. Comput. Phys. **229**, 933–946 (2010)

15. P Grnnwald, A Tutorial Introduction to the Minimum Description Length Principle, in *Advances in Minimum Description Length: Theory and Applications*, Edited by P. Grnwald and I. J. Myung and M. A. Pitt (MIT Press, Cambridge, MA, 2005)

16. T Hastie, R Tibshirani, J Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics (Springer, New York, 2001)

17. S Konishi, G Kitagawa, *Information Criteria and Statistical Modeling* (Springer, New York, 2007)

18. MH Hansen, B Yu, Model selection and the principle of minimum description length. J. Am. Stat. Assoc. **96**, 746–774 (1998)

19. S Engelberg, E Tadmor, Recovery of edges from spectral data with noise—a new perspective. SIAM J. Numer. Anal. **46**(5), 2620–2635 (2008)

20. G Plonka, The easy path wavelet transform: a new adaptive wavelet transform for sparse representation of two-dimensional data. Multiscale Model. Simul. **7**(3), 1474–1496 (2009). [Online] Available: http://link.aip.org/link/?MMS/7/1474/1

21. J Ma, G Plonka, H Chauris, New sparse representation of seismic data using adaptive easy-path wavelet transform. IEEE Geosci. Remote Sens. Lett. **7**, 540–544 (2010)
22. A Gelb, E Tadmor, Detection of edges in spectral data. Appl. Comput. Harmon. Anal. **7**, 101–135 (1999)
23. A Gelb, E Tadmor, Detection of edges in spectral data II. nonlinear enhancement. SIAM J. Numer. Anal. **38**(4), 1389–1408 (2000)
24. AV Oppenheim, RW Schafer, *Discrete-Time Signal Processing* (Prentice Hall Press, Upper Saddle River, 2009)
25. RR Coifman, DL Donoho, Translation-invariant de-noising, Department of Statistics, Tech. Rep. (1995). Available online: citeseer.ist.psu.edu/coifman95translationinvariant.html
26. PL Dragotti, M Vetterli, Shift-invariant Gibbs free denoising algorithm based on wavelet transform footprints, in *Proc. SPIE Wavelet Applications in Signal and Image Processing VIII*, vol. 4119, San Diego, CA, 2000, pp. 821–830
27. PL Dragotti, M Vetterli, Wavelet footprints: theory, algorithms and applications. IEEE Trans. Signal Process. **51**(5), 1306–1323 (2003)
28. R Archibald, A Gelb, Reducing the effects of noise in image reconstruction. J. Sci. Comput. **17**(1–4), 167–180 (2002)
29. C MacInnes, The reconstruction of discontinuous piecewise polynomial signals. IEEE Trans. Signal Process. **53**(7), 2603–2607 (2005)
30. RR Coifman, DL Donoho, Wavelab 850, http://www-stat.stanford.edu/ wavelab/
31. G Plonka, S Tenorth, D Rosca, A New Hybrid Method for Image Approximation Using the Easy Path Wavelet Transform. IEEE Trans. Image Process. **20**(2), 372–381 (2011)
32. D Scharstein, R Szeliski, A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. Int. J. Comput. Vis. **47**(1–3), 7–42 (2002), see also http://vision.middlebury.edu/stereo/
33. E Candes, M Wakin, An introduction to compressive sampling. IEEE Signal Process. Mag. **25**(2), 21–30 (2008)

Figure 1.: **Comparison of the performance of different model selection methods at SNR** $= 10$ **to** $20\,$**dB for: (a)** piecewise polynomial signal $f(x)$, and **(b)** Heavisine signal.

Figure 2.: **Effect of the number of DCT coefficients used in IPRM for signal** $f_1(x)$. **(a)** Signal $f_1(x)$. **(b)** DCT coefficients of $f_1(x)$.

Figure 3.: **Effect of the number of DCT coefficients used in IPRM**. **(a)** RMSE of the reconstructed signal from signal $f_1(x)$ versus the number of coefficients N_d. **(b)** RMSE of the reconstructed signal from signal $f_2(x)$ versus the number of coefficients N_d.

Figure 4.: **Effect of the number of DCT coefficients used in IPRM for signal** $f_2(x)$. **(a)** Signal $f_2(x)$. **(b)** DCT coefficients of $f_2(x)$.

Figure 5.: **Effect of the number of DCT coefficients used in IPRM in the presence of noise**. **(a)** RMSE of the reconstructed signal from signal $f_1(x)$ with SNR $= 10\,$dB versus the number of coefficients N_d. **(b)** RMSE of the reconstructed signal from signal $f_2(x)$ with SNR $= 10\,$dB versus the number of coefficients N_d.

Figure 6.: **Easy path algorithm: (a)** without, and **(b)** with the proposed modification.

Figure 7.: **Comparison of different de-noising methods for Blocks signal (blue: original signal, red: reconstructed signal)**. **(a)** Noisy signal with SNR $= 16.91\,$dB. **(b)** Cycle-spinning de-noising. **(c)** Wavelet footprints de-noising. **(d)** Wavelet footprints de-noising with oracle. **(e)** Direct method de-noising with oracle. **(f)** IPRM de-noising.

Figure 8.: **Comparison of different de-noising methods for a piecewise polynomial signal (blue: original signal, red: reconstructed signal)**. **(a)** Original signal. **(b)** Noisy signal with SNR = 16.91 dB. **(c)** Cycle-spinning de-noising. **(d)** Wavelet footprints de-noising. **(e)** Direct method de-noising with oracle. **(f)** IPRM de-noising.

Figure 9.: **Comparison of different de-noising methods for Heavisine signal (blue: original signal, red: reconstructed signal)**. **(a)** Original Heavisine signal. **(b)** Noisy signal with SNR = 16.91 dB. **(c)** Cycle-spinning de-noising. **(d)** Direct method de-noising with oracle. **(e)** IPRM de-noising.

Figure 10.: **Comparison of different de-noising methods for Doppler signal (blue: original signal, red: reconstructed signal)**. **(a)** Original Doppler signal. **(b)** Noisy signal with SNR = 16.91 dB. **(c)** Cycle-spinning de-noising. **(d)** Direct method de-noising with oracle. **(e)** IPRM de-noising. **(f)** IPRM de-noising after one step of division.

Figure 11.: **Comparison of RMSE of IPRM with other de-noising algorithms at different SNR values for: (a)** blocks signal, and **(b)** piecewise polynomial signal $f(x)$.

Figure 12.: **Approximation of the phantom image: (a)** Original phantom image, **(b)** Approximation using Haar wavelets ($N_d = 3300$ coefficients).

Figure 13.: **Approximation of the phantom image: (a)** Approximation using the EPWT with Haar ($N_d = 650$ coefficients), **(b)** Approximation using the easy path algorithm and DCT-based IPRM ($N_d = 175$ coefficients).

Figure 14.: **Approximation of a piecewise linear image: (a)** Original image, **(b)** Approximation using Haar wavelets ($N_d = 2800$ coefficients).

Figure 15.: **Approximation of a piecewise linear image: (a)** Approximation using the EPWT with Haar ($N_d = 200$ coefficients), **(b)** Approximation using the easy path algorithm and DCT-based IPRM ($N_d = 40$ coefficients).

Figure 16.: **Approximation of the disparity map image: (a)** Color texture image, **(b)** Original disparity map image corresponding to (a).

Figure 17.: **Approximation of the disparity map image:** Approximation using the easy path algorithm and DCT-based IPRM ($N_d = 130$ coefficients).

Figure 18.: **Approximation of the disparity map image: (a)** Approximation using Haar wavelets ($N_d = 3000$ coefficients), **(b)** Approximation using the EPWT with Haar ($N_d = 500$ coefficients).

Table 1.: Comparison of de-noising algorithms for blocks signal.

Method	RMSE
Cycle-spinning	7.7701
Direct method + Oracle	8.2528
Wavelet footprints	21.5279
Wavelet footprints + Oracle	7.2656
IPRM	4.1078
IPRM + Oracle	4.1078

Table 2.: Comparison of de-noising algorithms for signal $f(x)$.

Method	RMSE
Cycle-spinning	2.6876
Direct method + Oracle	1.2733
Wavelet footprints	1.2859
IPRM	0.1236
IPRM + Oracle	0.1236

Table 3.: Comparison of de-noising algorithms for Heavisine signal.

Method	RMSE
Cycle-spinning	9.0905
Direct method + Oracle	6.7298
IPRM	4.9491
IPRM + Oracle	4.9491

Table 4.: Comparison of de-noising algorithms for Doppler signal.

Method	RMSE
Cycle-spinning	17.5325
Direct method + Oracle	31.5494
IPRM	13.6757
IPRM + Oracle	13.6417

Table 5.: Effect of the parameter θ for the phantom image.

$\boldsymbol{\theta}$	Number of edges	Entropy of $\tilde{\mathbf{p}}$
1	251	0.19
2	86	0.36
5	6	0.89

Table 6.: Approximation results of DCT-based IPRM, EPWT and tensor product wavelet transform for the phantom image.

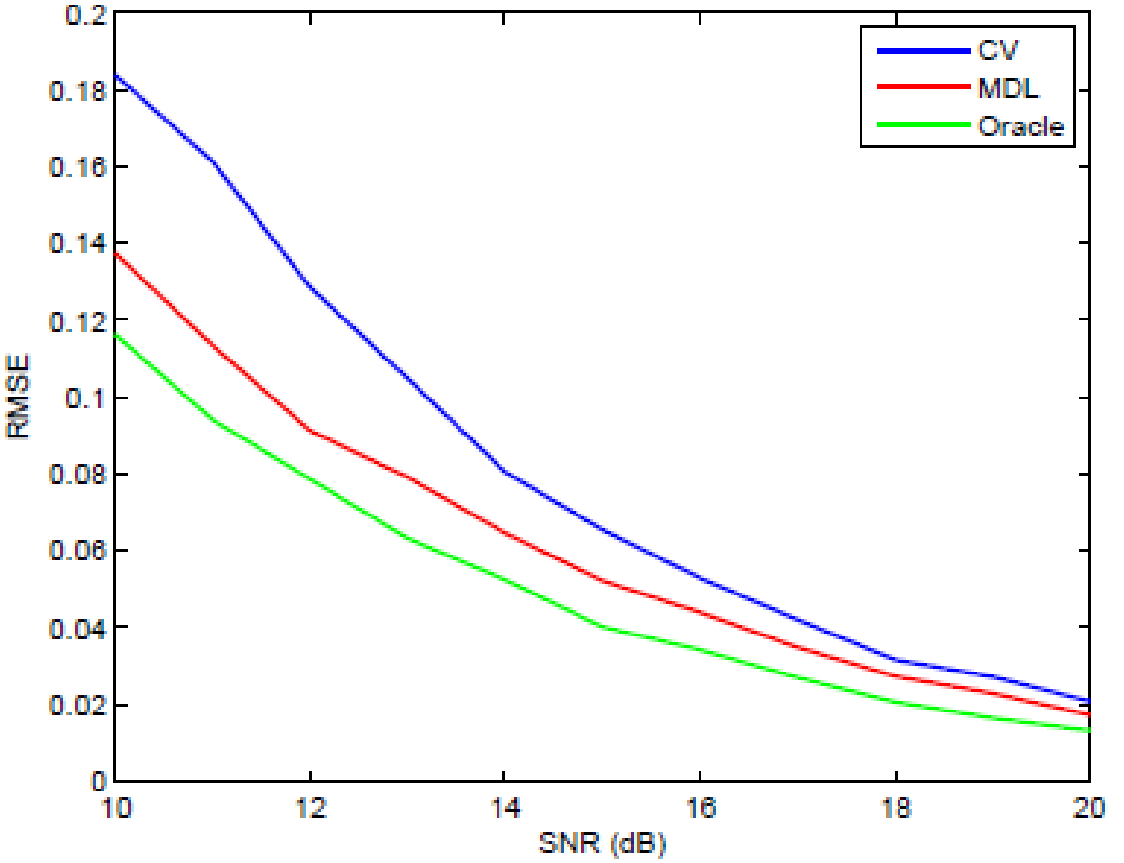
Method	L	N_d	SNR (dB)	Entropy of $\tilde{\mathbf{p}}$	Storage cost (bpp)	
					$b = 8$	$b = 16$
DCT IPRM	–	175	68.39	0.36	0.40	0.43
EPWT Haar	9	650	35.59	0.38	0.54	0.61
EPWT db4	9	1500	35.73	0.38	0.71	0.89
EPWT 7/9	9	2700	35.56	0.38	0.94	1.27
tensor prod. Haar	5	3300	36.23	–	0.69	1.09
tensor prod. db4	5	9000	35.79	–	1.67	2.77
tensor prod. 7/9	5	8000	35.30	–	1.51	2.48

Table 7.: **Approximation results of DCT-based IPRM, EPWT and tensor product wavelet transform for the piecewise linear image.**

Method	L	N_d	**SNR (dB)**	**Entropy of** $\tilde{\mathrm{p}}$	**Storage Cost (bpp)**	
					$b = 8$	$b = 16$
DCT IPRM	–	40	34.58	0.14	0.15	0.16
EPWT Haar	9	200	35.25	0.33	0.39	0.41
EPWT db4	9	200	35.75	0.33	0.39	0.41
EPWT 7/9	9	500	35.18	0.33	0.46	0.52
tensor prod. Haar	5	2800	35.31	–	0.59	0.93
tensor prod. db4	5	3200	35.26	–	0.67	1.06
tensor prod. 7/9	5	1800	36.00	–	0.41	0.62

Table 8.: **Approximation results of DCT-based IPRM, EPWT and tensor product wavelet transform for the disparity map image.**

Method	L	N_d	**SNR (dB)**	**Entropy of** $\tilde{\mathrm{p}}$	**Storage cost (bpp)**	
					$b = 8$	$b = 16$
DCT IPRM	–	130	30.16	0.54	0.57	0.60
EPWT Haar	9	500	30.29	0.57	0.69	0.76
EPWT db4	9	750	29.81	0.57	0.75	0.84
EPWT 7/9	9	900	29.89	0.57	0.78	0.89
tensor prod. Haar	5	3000	30.37	–	0.63	1.00
tensor prod. db4	5	5000	29.80	–	0.99	1.60
tensor prod. 7/9	5	3500	29.85	–	0.72	1.15


CV
MDL
Oracle
RMSE
SNR (dB)
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
10
12
14
16
18
20

(a)

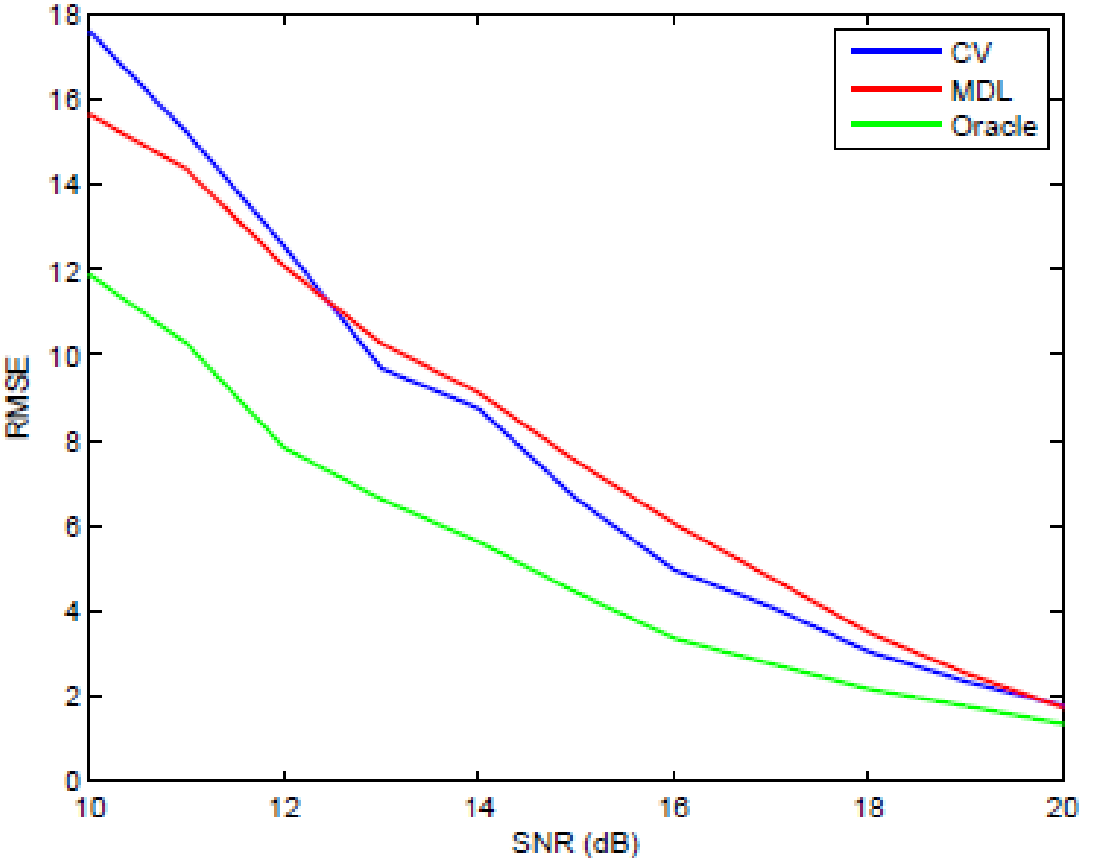

CV
MDL
Oracle
RMSE
SNR (dB)
18
16
14
12
10
8
6
4
2
0
10
12
14
16
18
20

Figure 1

(b)

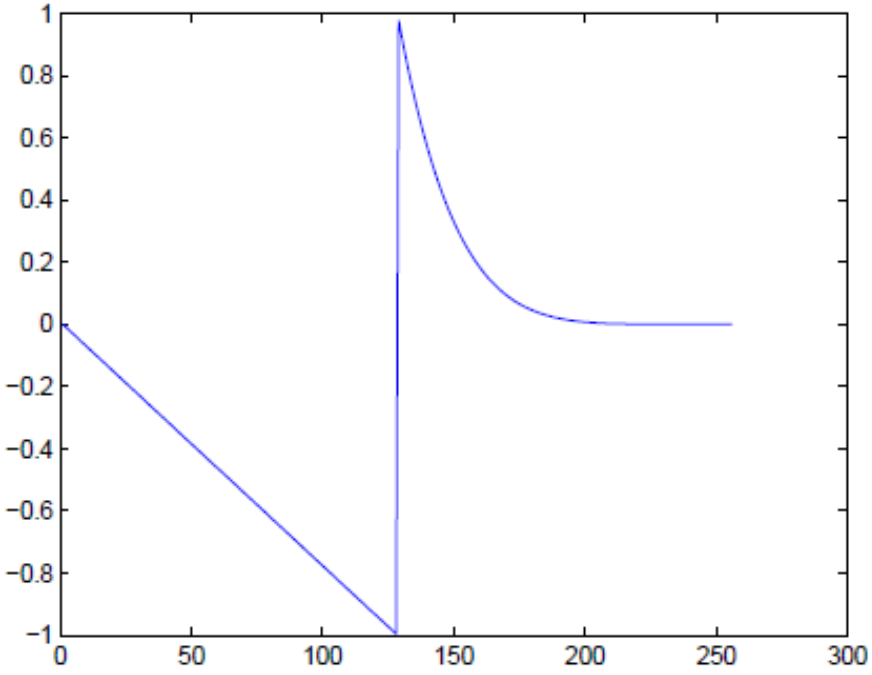
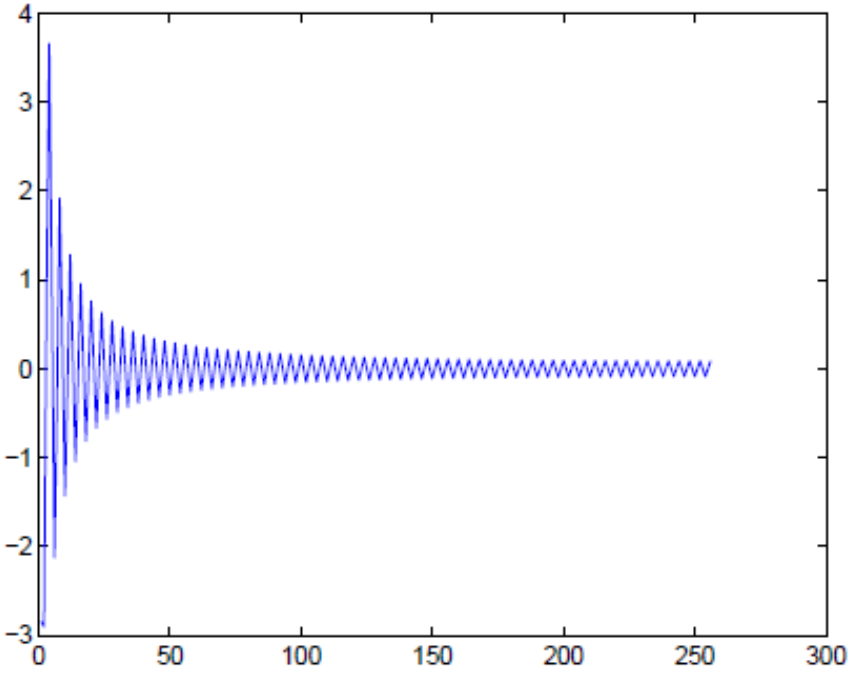

1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
0
50
100
150
200
250
300

Figure 2

(a)


4
3
2
1
0
−1
−2
−3
0
50
100
150
200
250
300

(b)

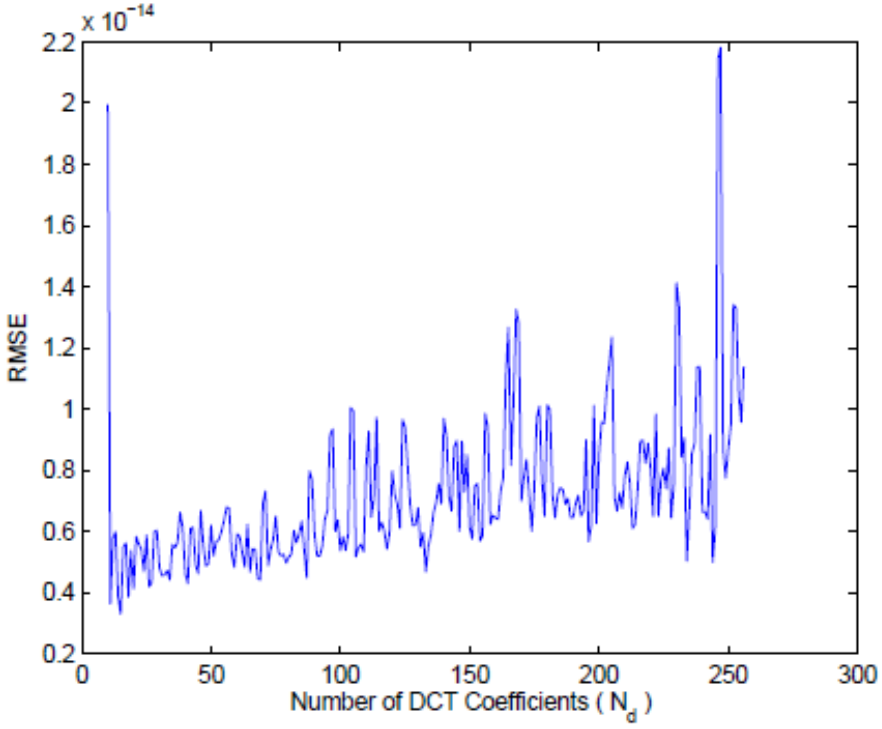
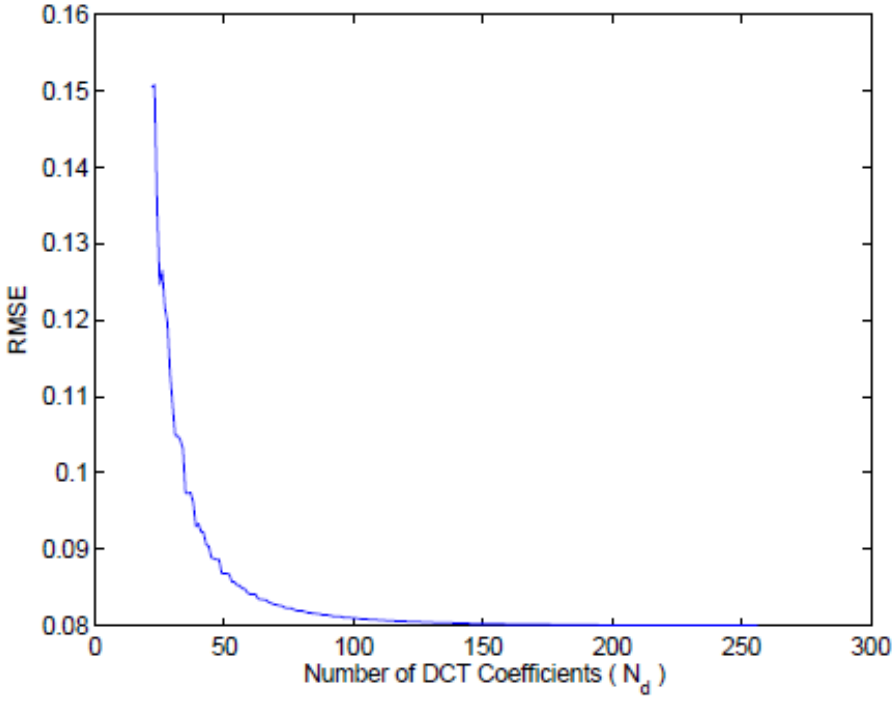
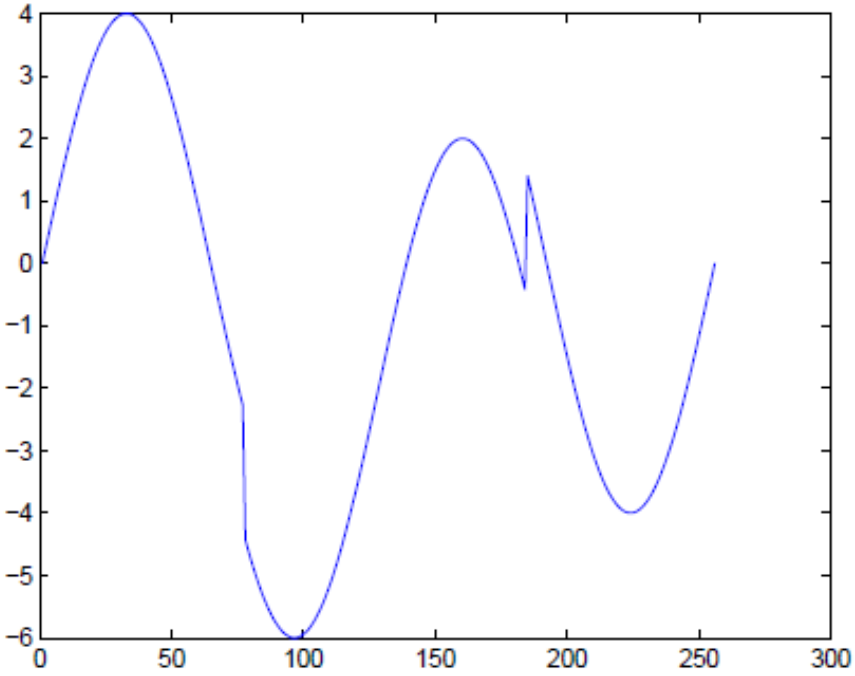

x 10^-14
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
RMSE
0
50
100
150
200
250
300
Number of DCT Coefficients (N_d)

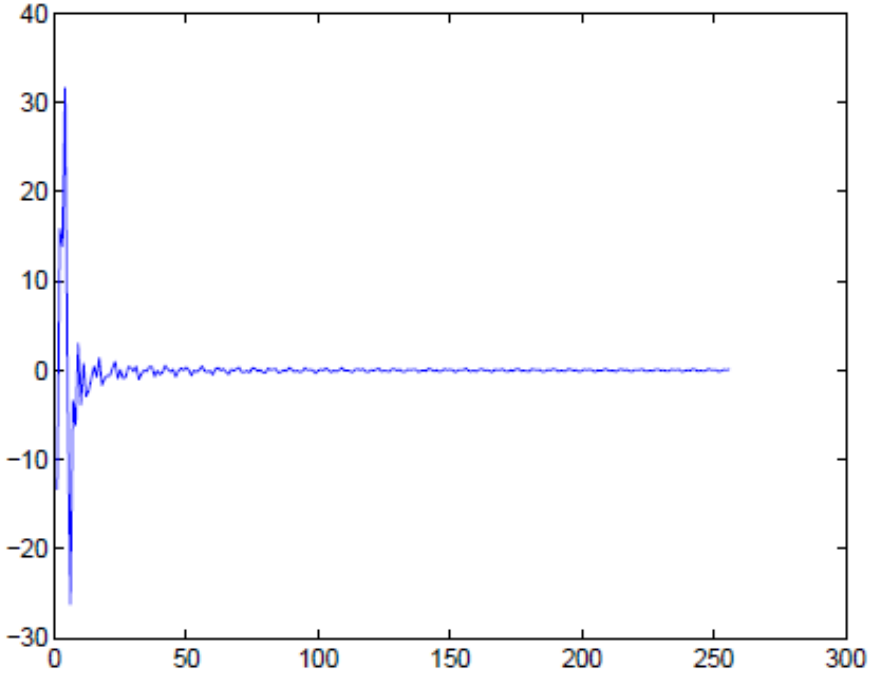
Figure 3 (a)


0.16
0.15
0.14
0.13
0.12
0.11
0.1
0.09
0.08
RMSE
0
50
100
150
200
250
300
Number of DCT Coefficients (N_d)

(b)

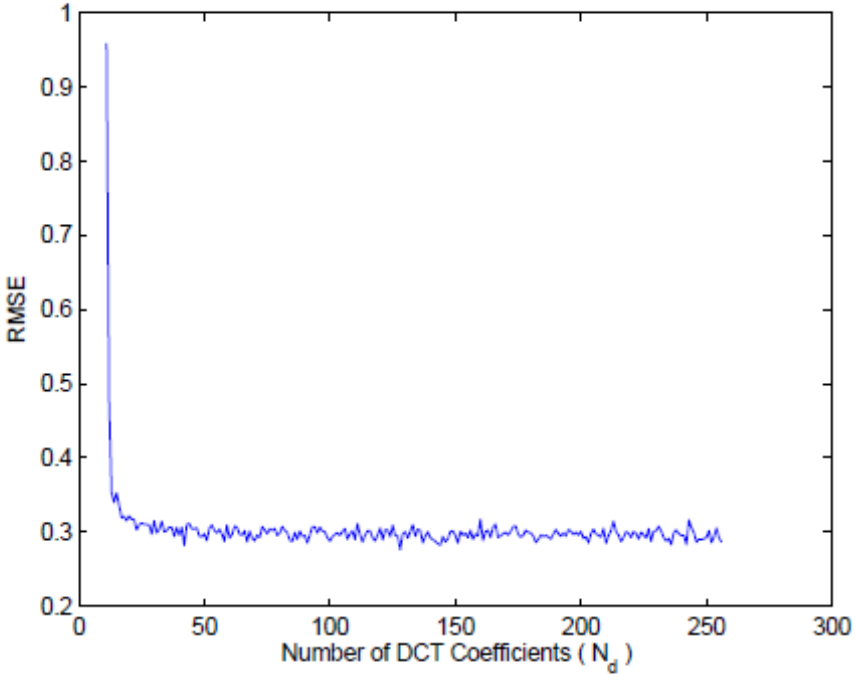

4
3
2
1
0
−1
−2
−3
−4
−5
−6
0
50
100
150
200
250
300

(a)

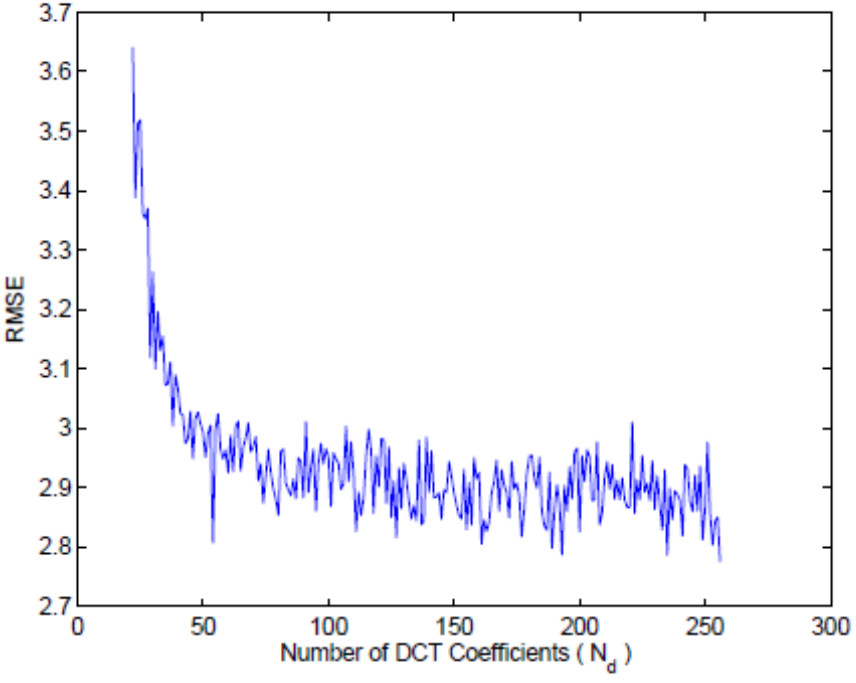

40
30
20
10
0
−10
−20
−30
0
50
100
150
200
250
300

(b)

Figure 4

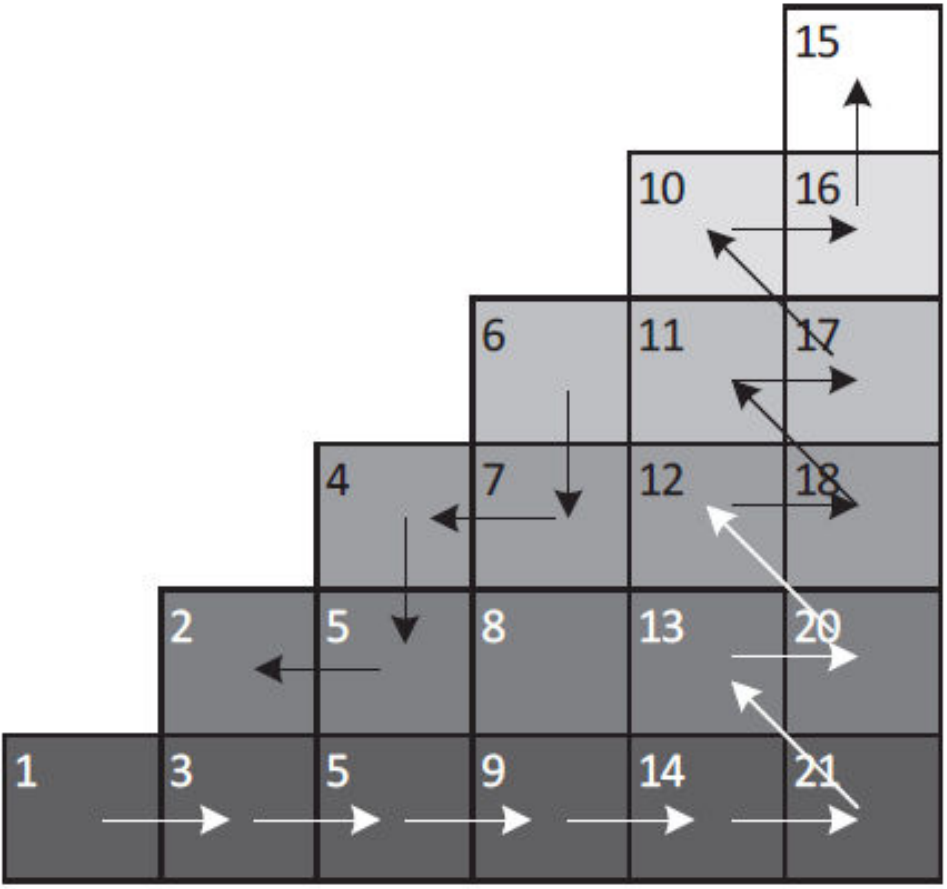

RMSE
Number of DCT Coefficients (N_d)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
50
100
150
200
250
300

(a)

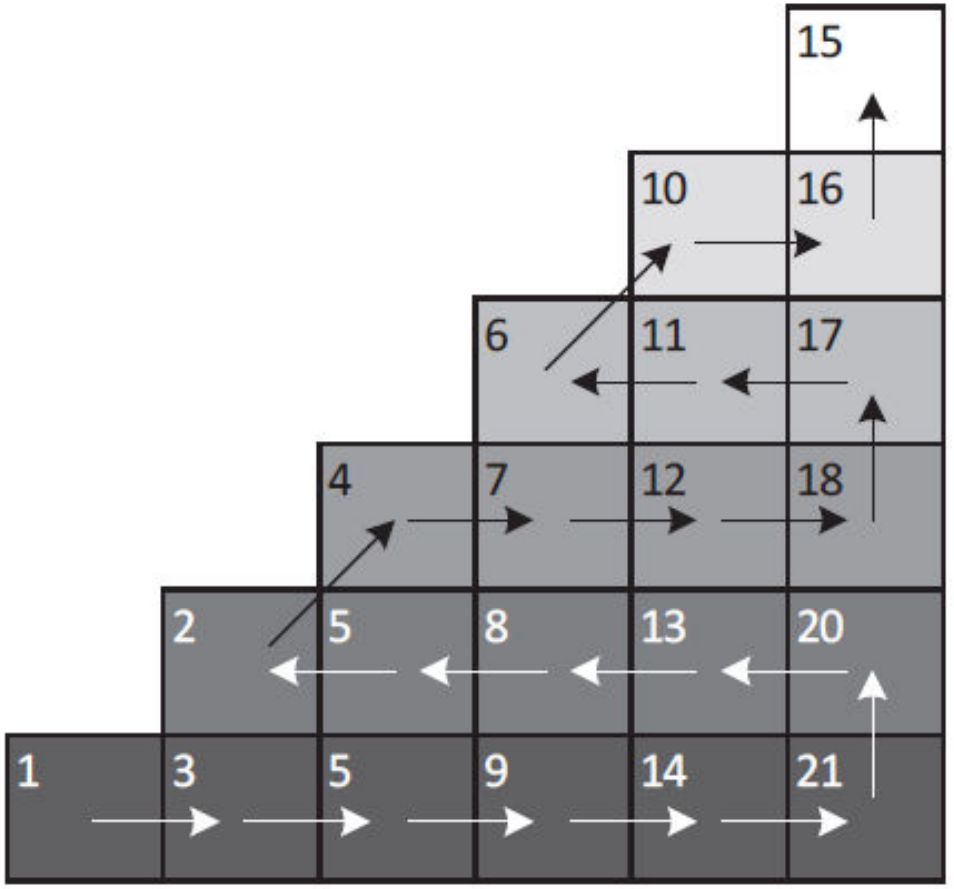

RMSE
Number of DCT Coefficients (N_d)
3.7
3.6
3.5
3.4
3.3
3.2
3.1
3
2.9
2.8
2.7
0
50
100
150
200
250
300

(b)

Figure 5

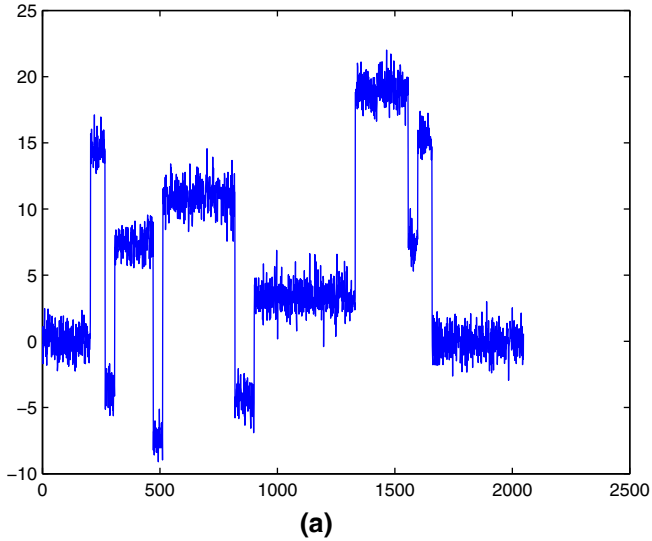

15
10
16
6
11
17
4
7
12
18
2
5
8
13
20
1
3
5
9
14
21

(a)

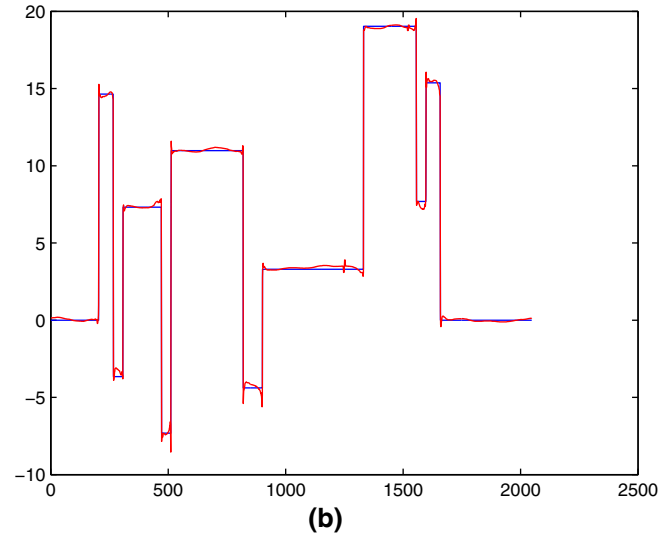

15
10
16
6
11
17
4
7
12
18
2
5
8
13
20
1
3
5
9
14
21

(b)

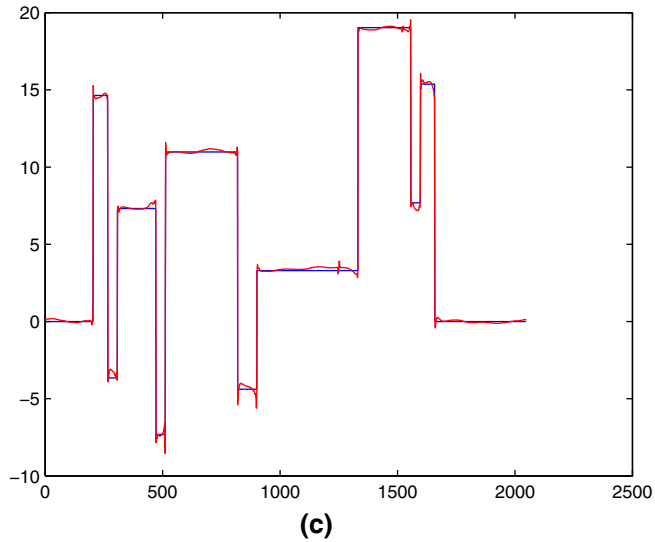
Figure 6



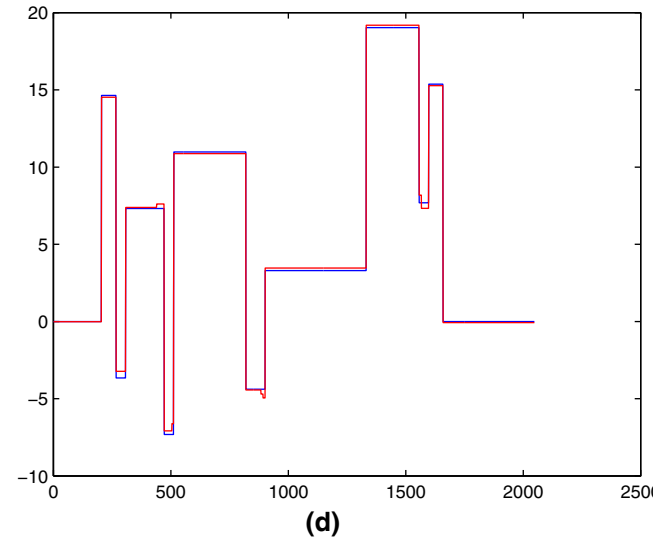
(a)



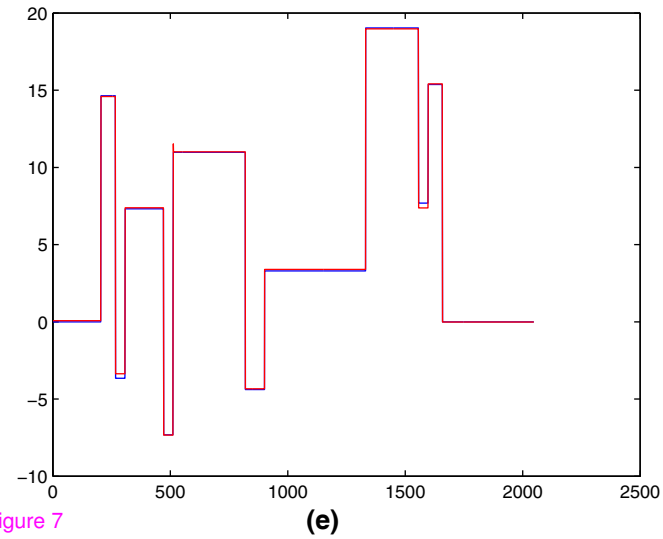
(b)



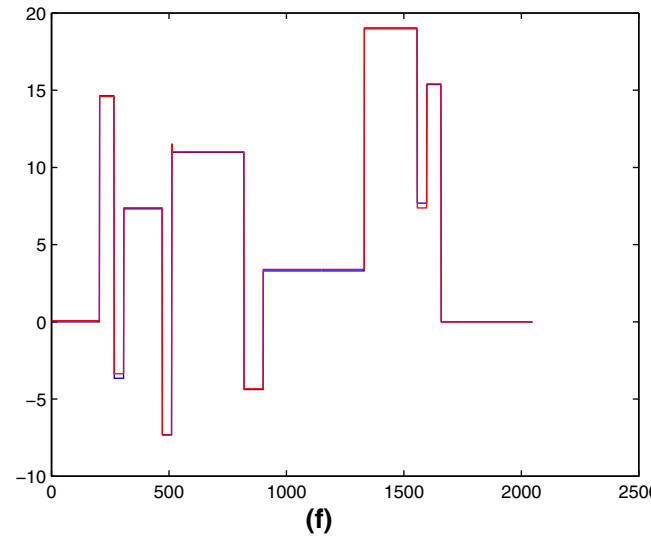
(c)



(d)

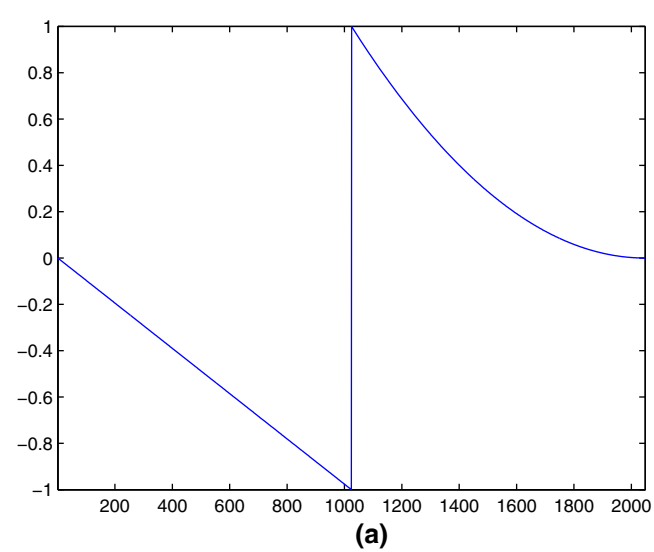


(e)

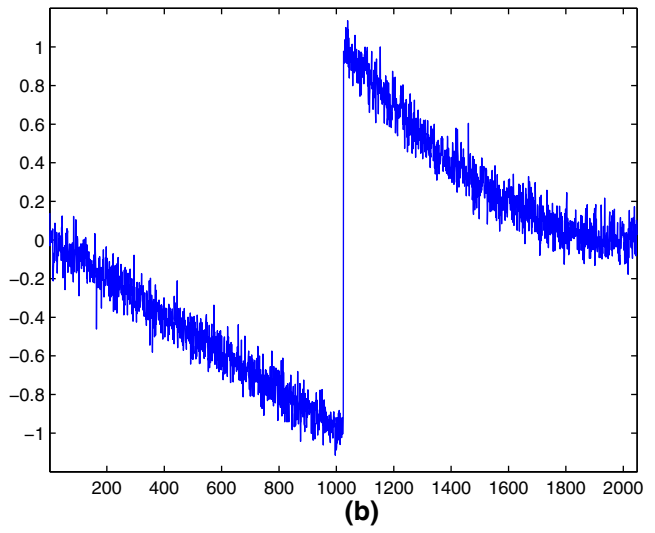


(f)

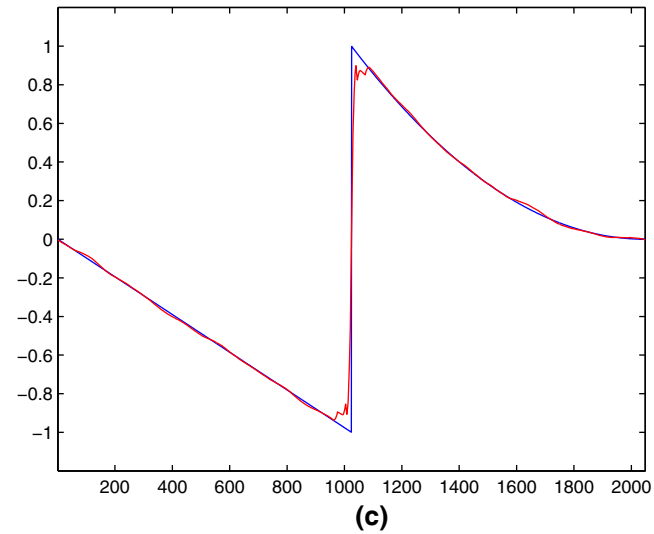
Figure 7



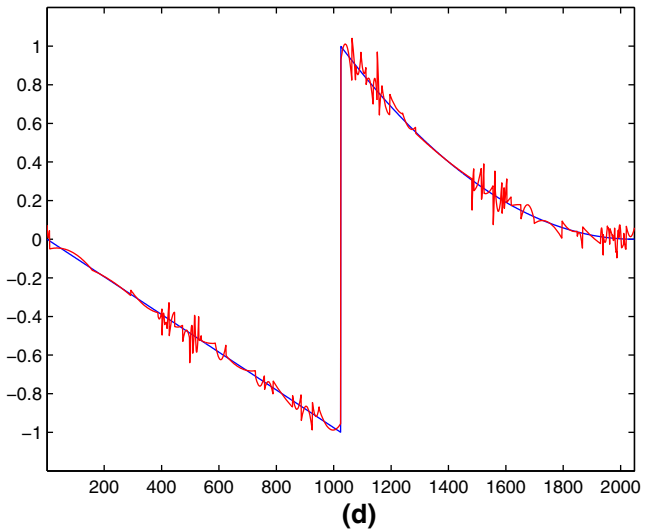
(a)



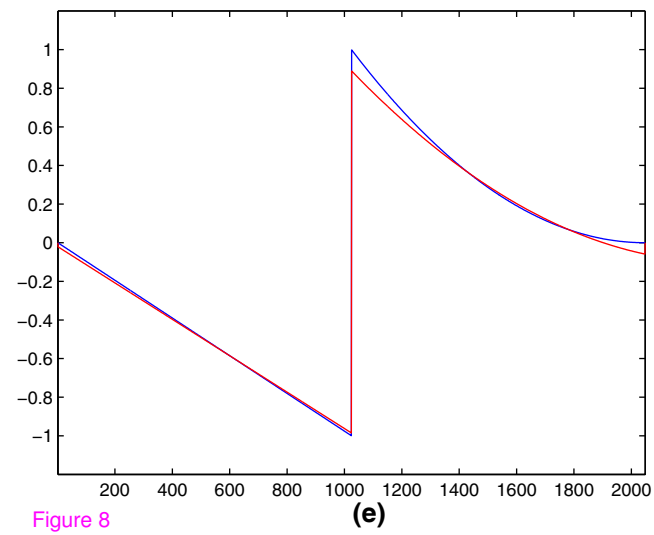
(b)



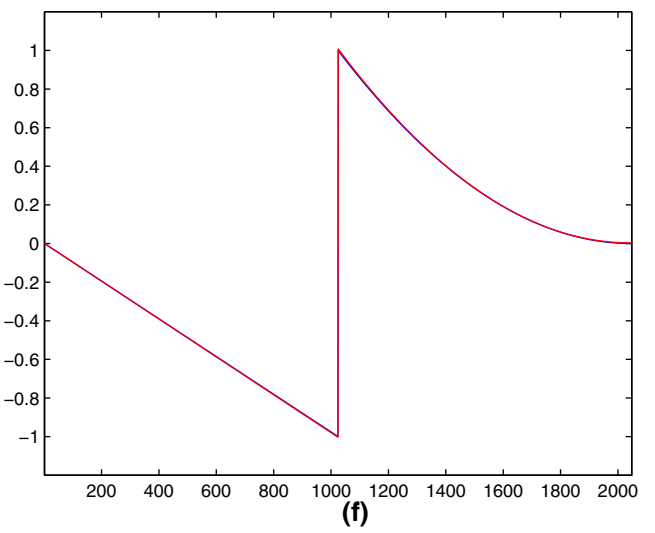
(c)



(d)

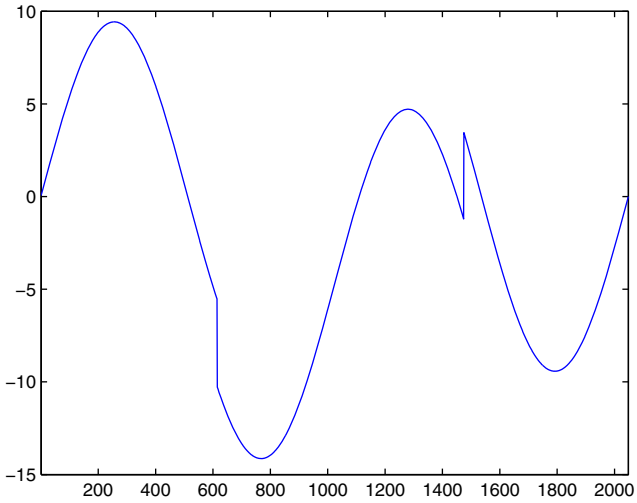


(e)

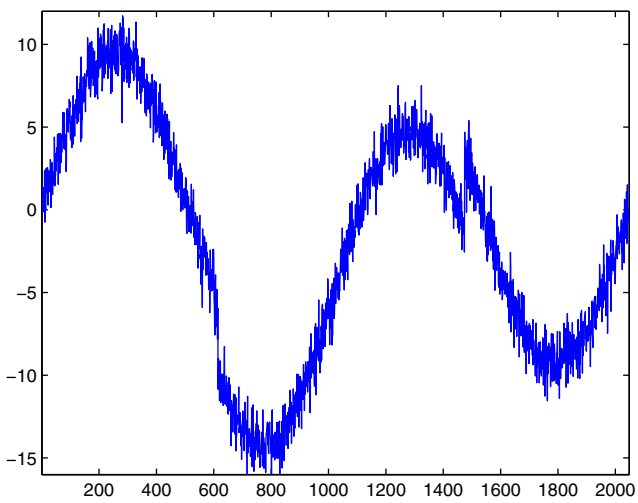


(f)

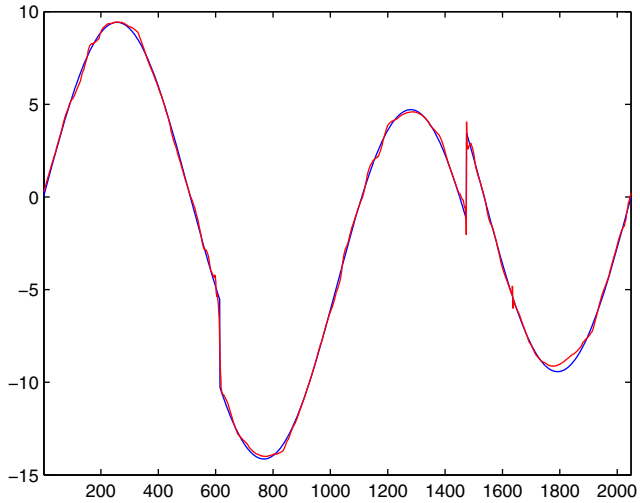
Figure 8



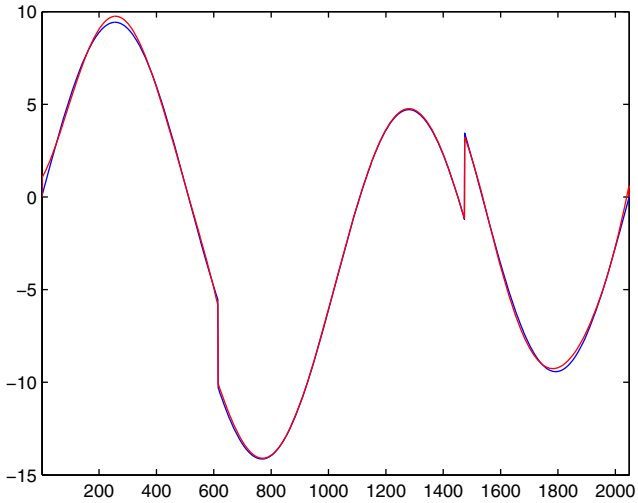
(a)



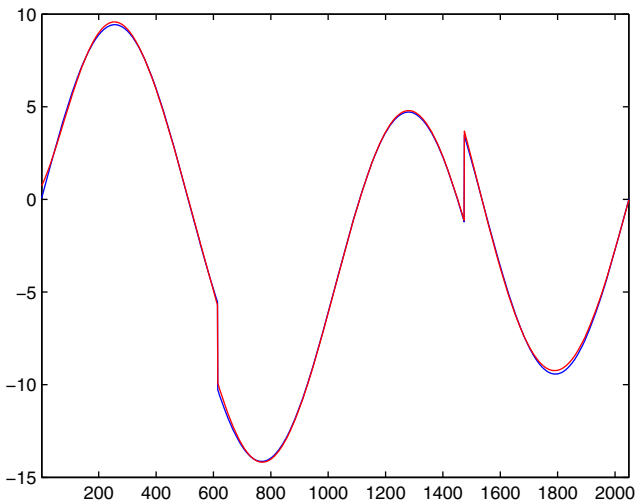
(b)



(c)

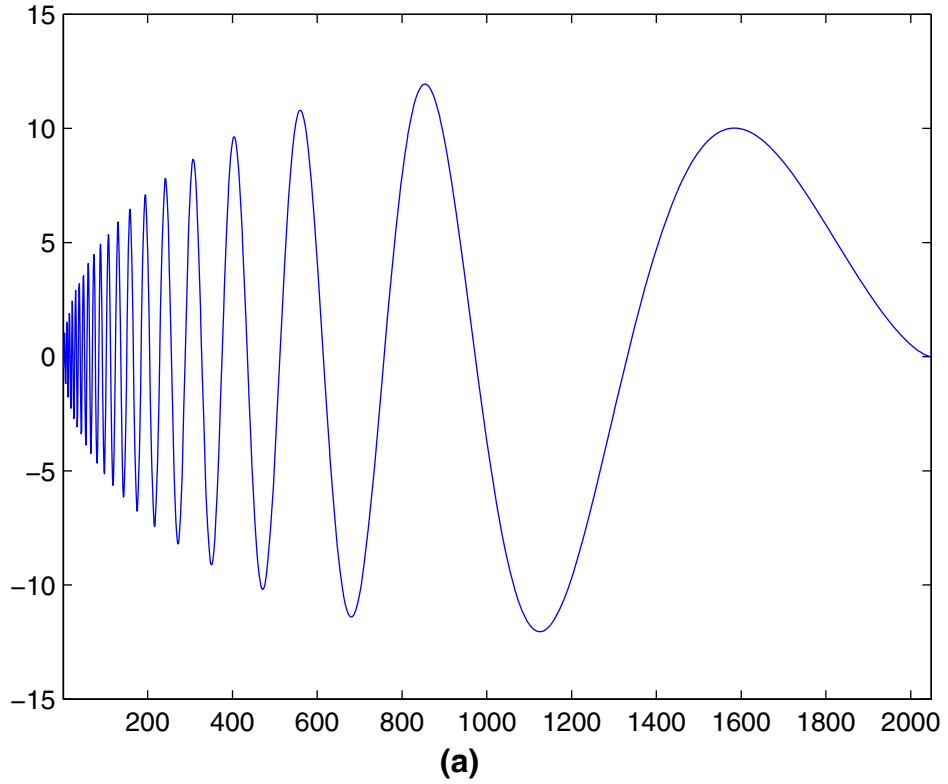


(d)

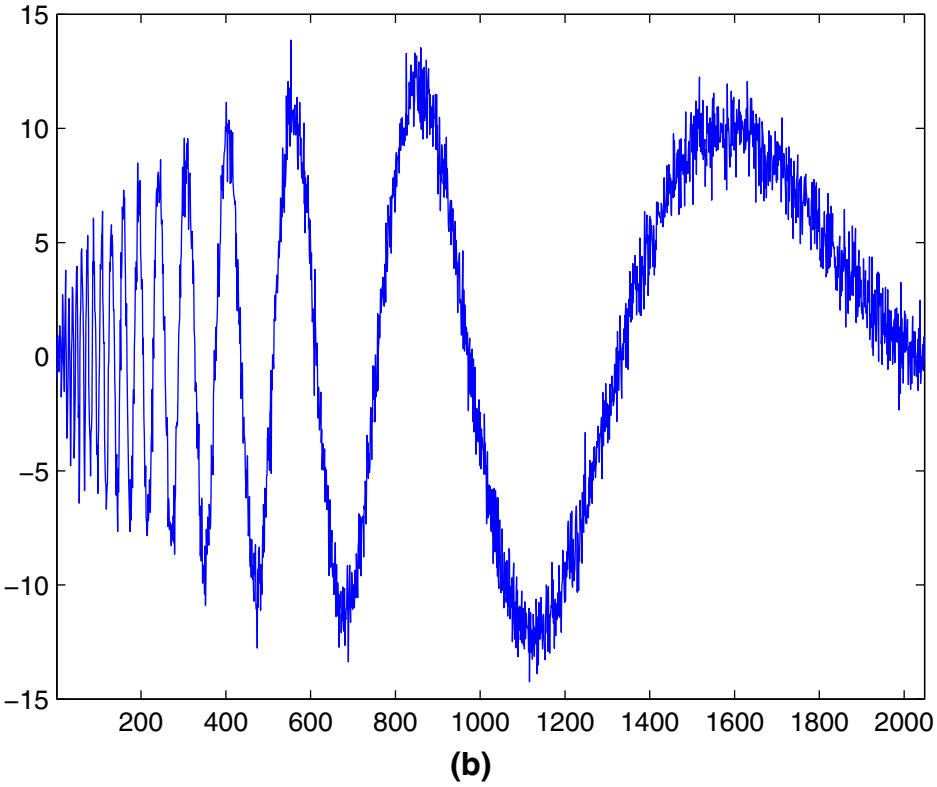


(e)

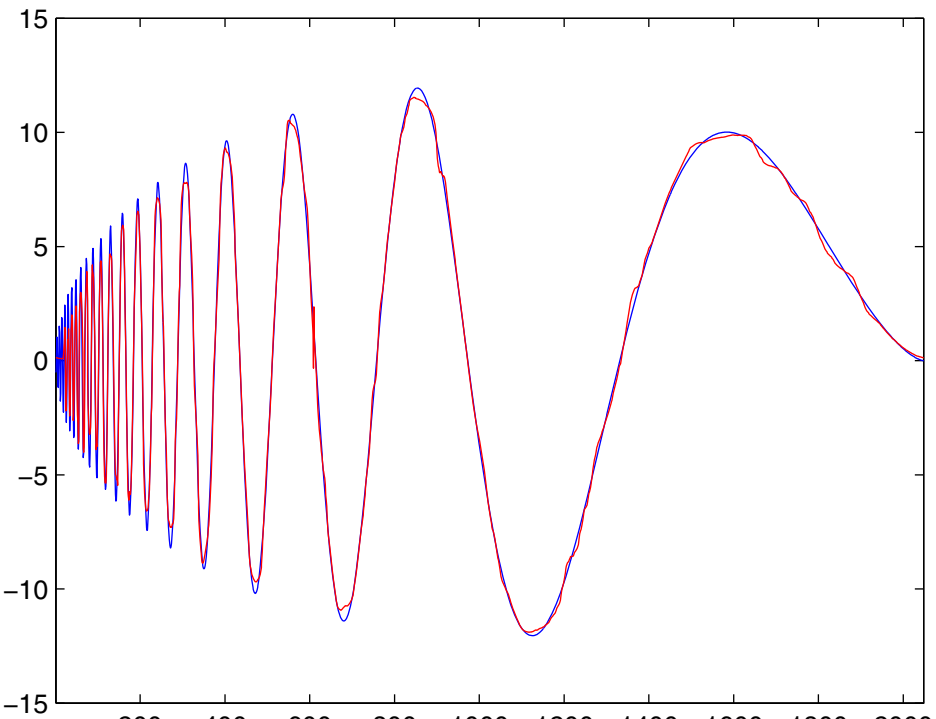
Figure 9



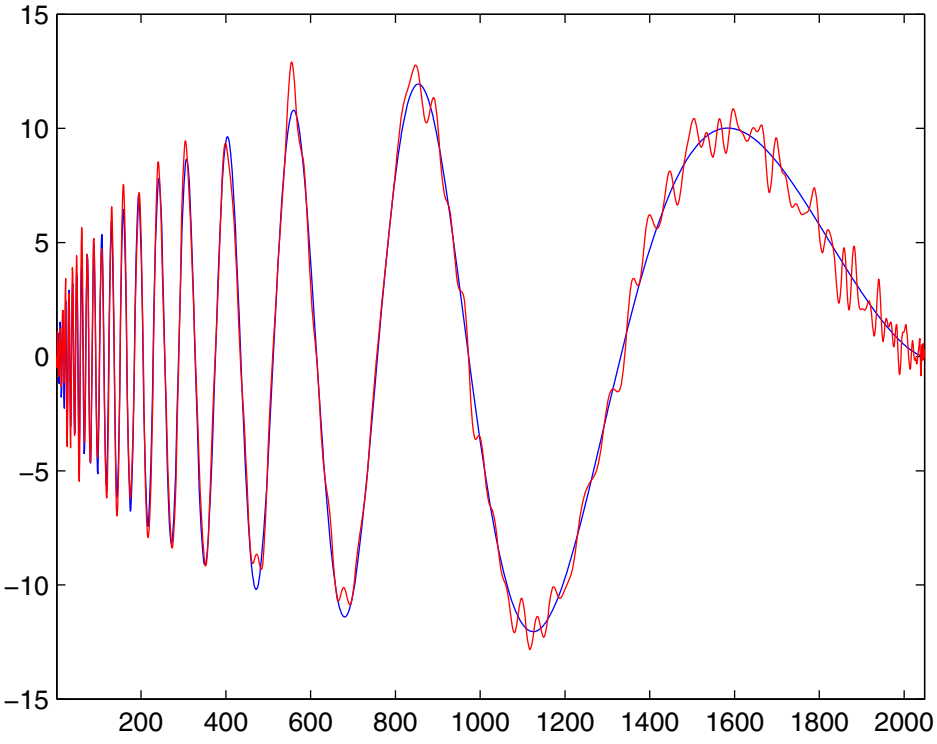
(a)



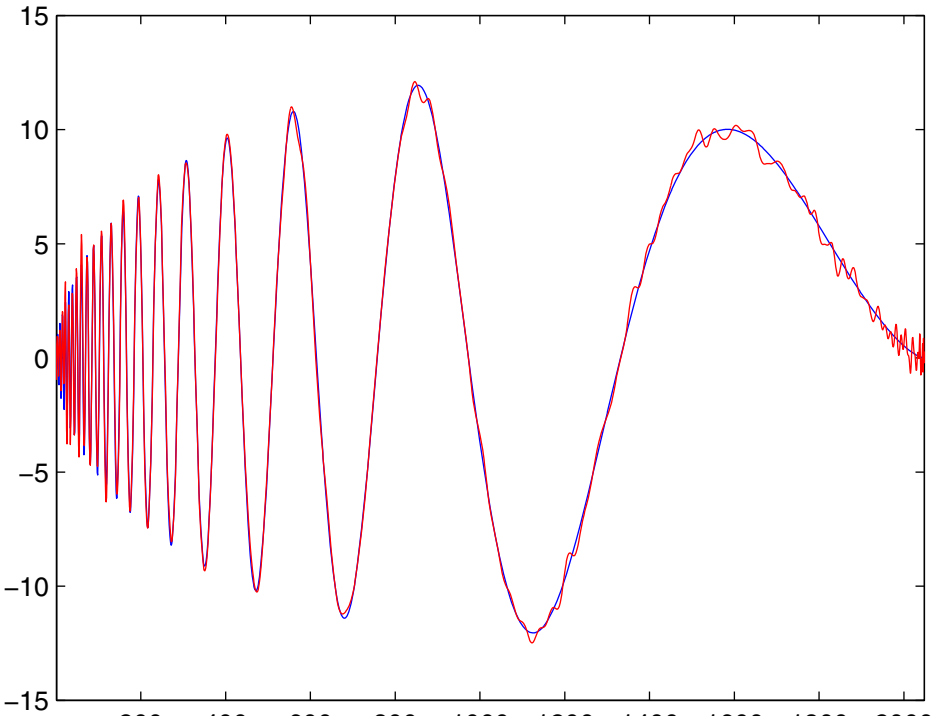
(b)



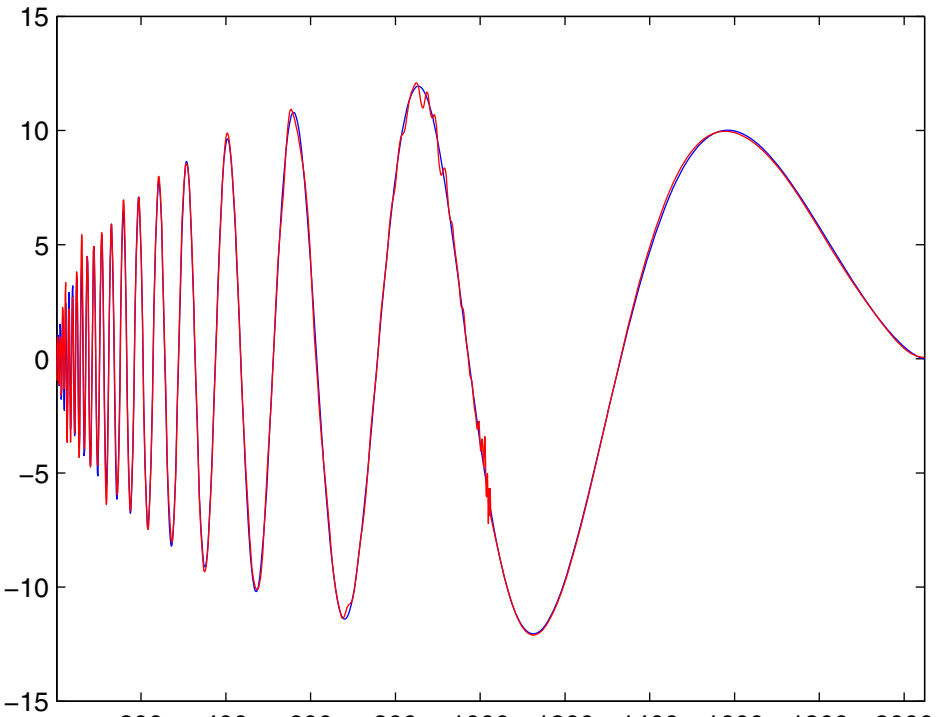
(c)



(d)

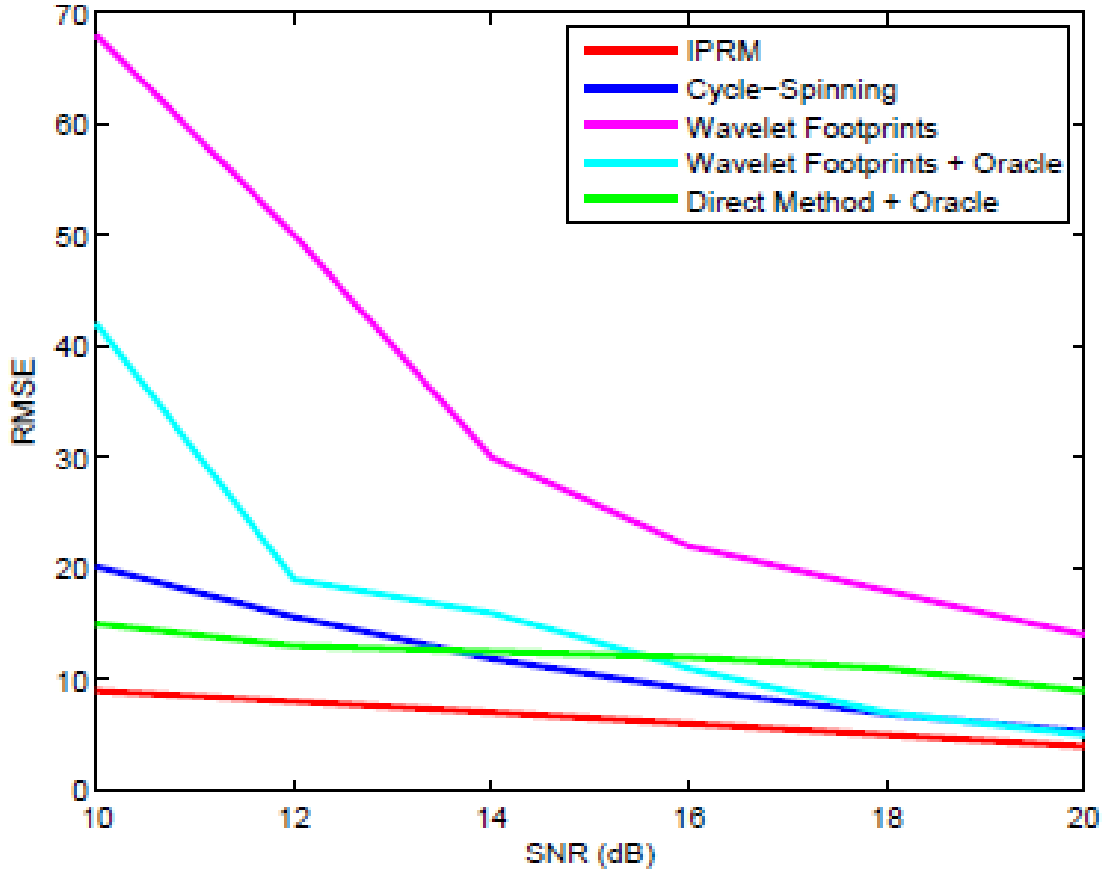


(e)



(f)

Figure 10


IPRM
Cycle−Spinning
Wavelet Footprints
Wavelet Footprints + Oracle
Direct Method + Oracle
RMSE
SNR (dB)
0
10
20
30
40
50
60
70
10
12
14
16
18
20

(a)

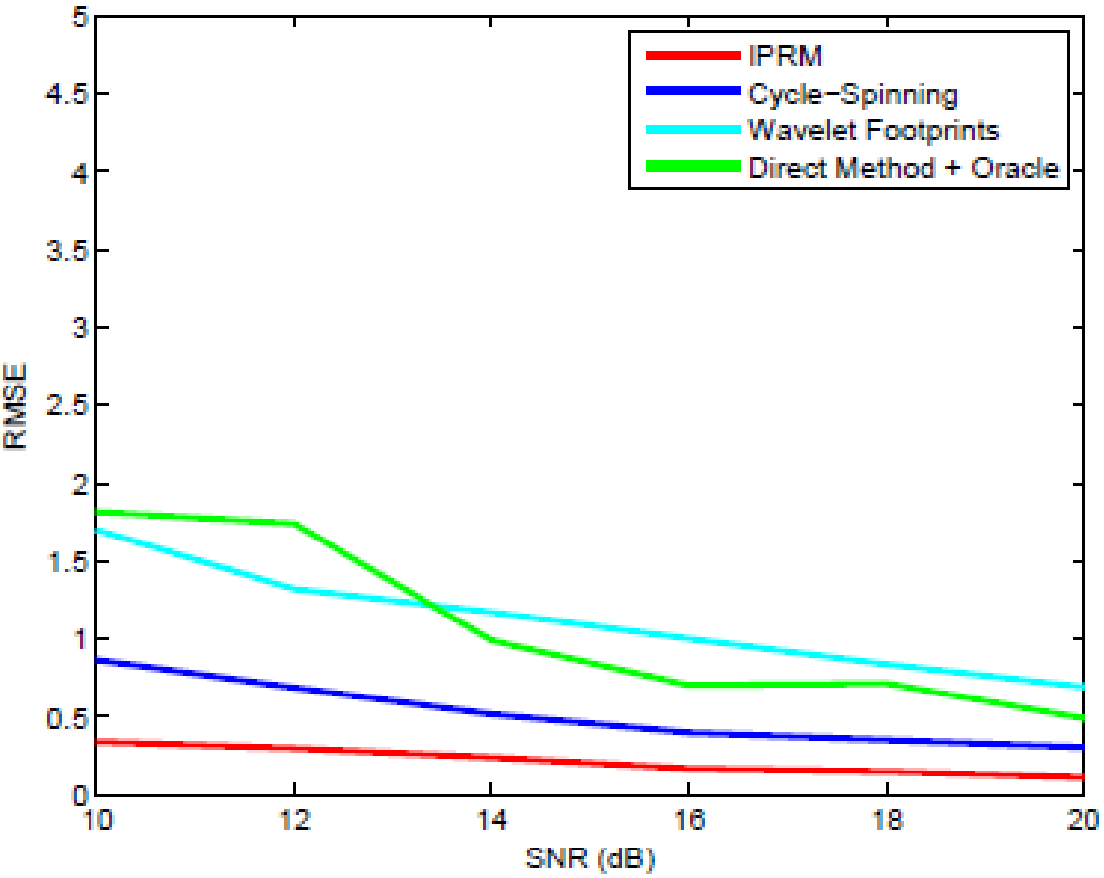

IPRM
Cycle−Spinning
Wavelet Footprints
Direct Method + Oracle
RMSE
SNR (dB)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
10
12
14
16
18
20

Figure 11

(b)

(a)

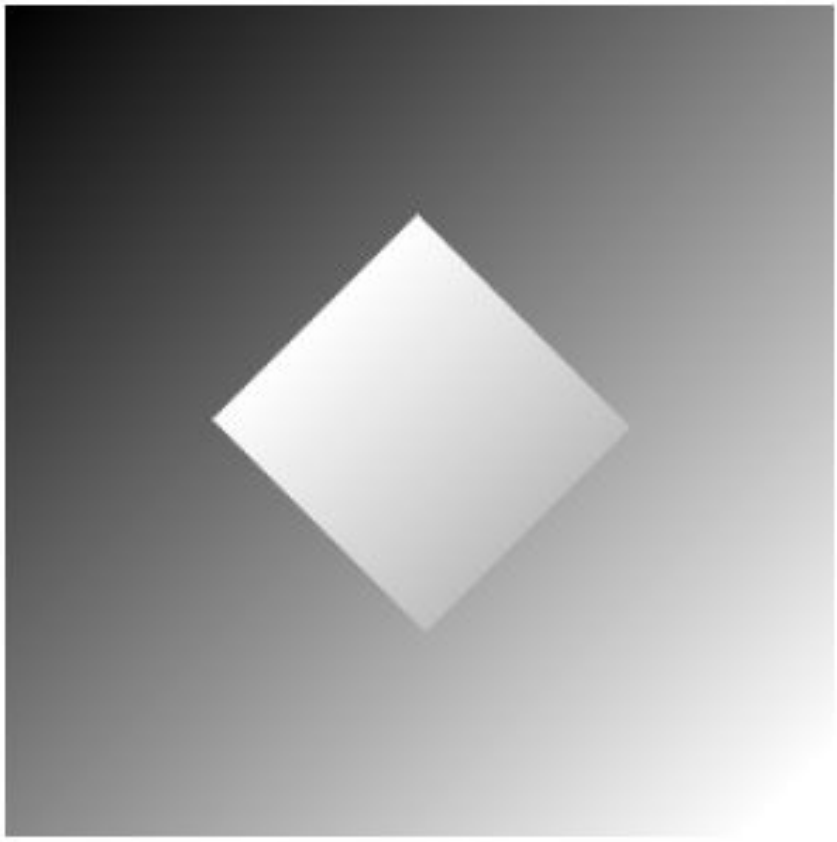
(b)

Figure 12

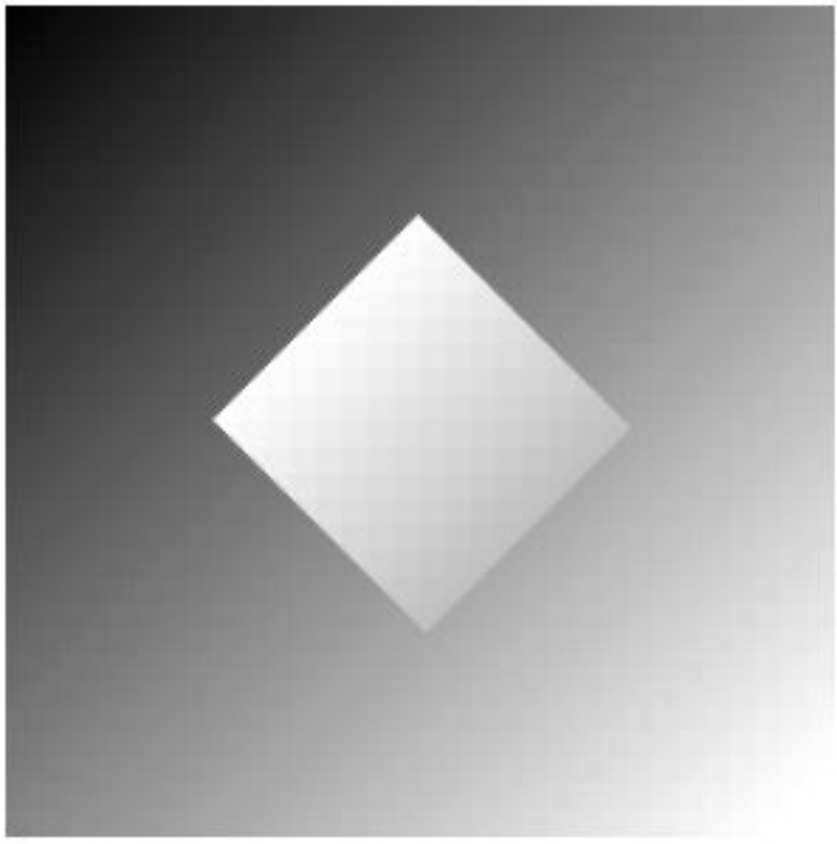
(a)

(b)

Figure 13

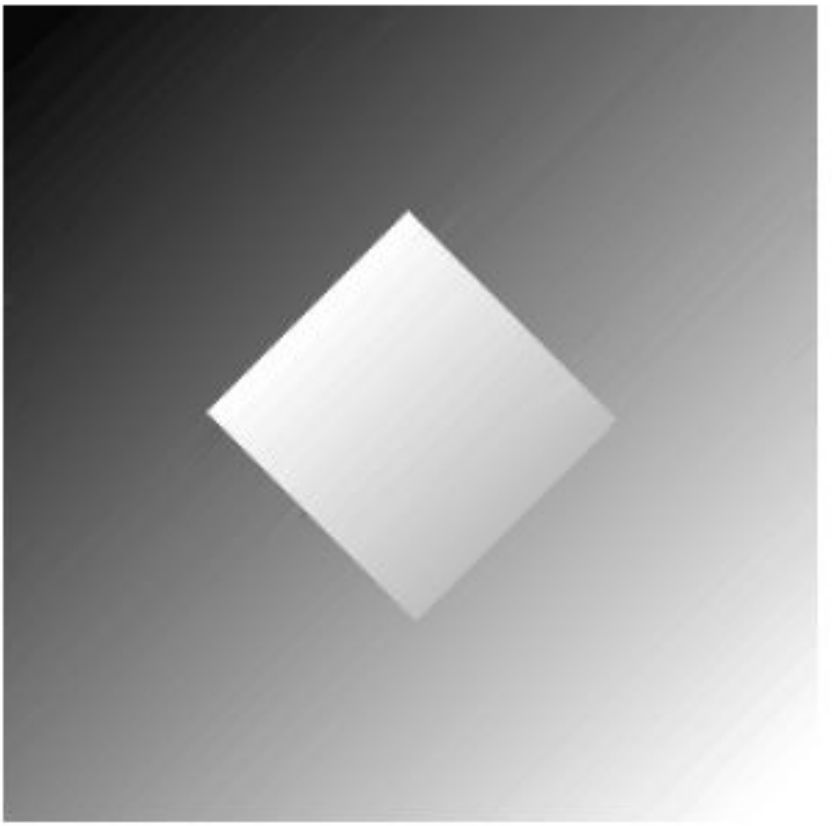


(a)

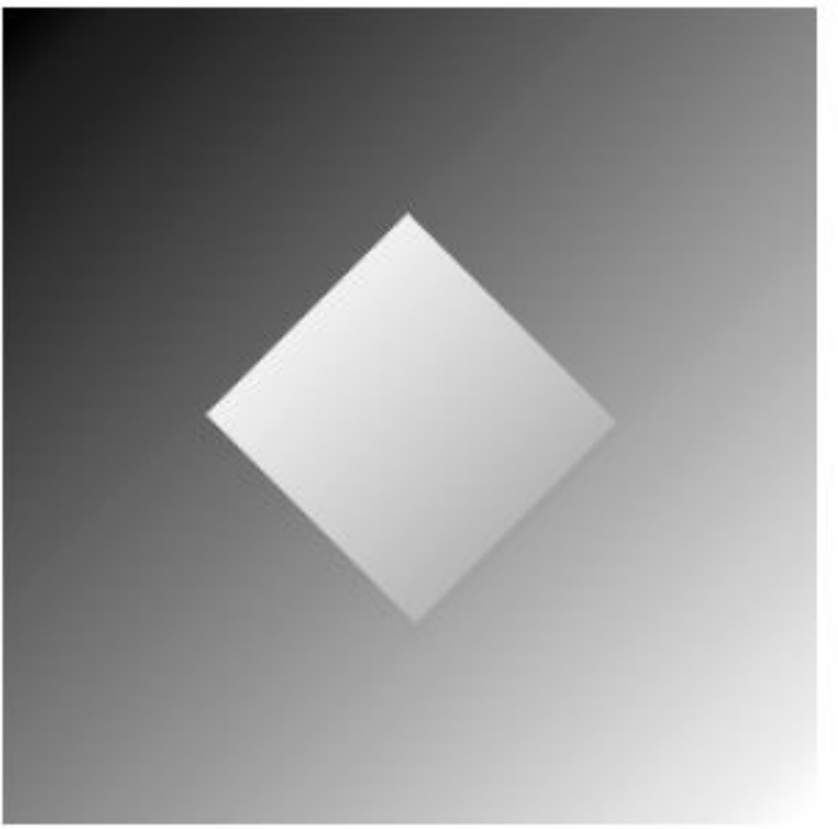


(b)

Figure 14



(a)



(b)

Figure 15

(a)

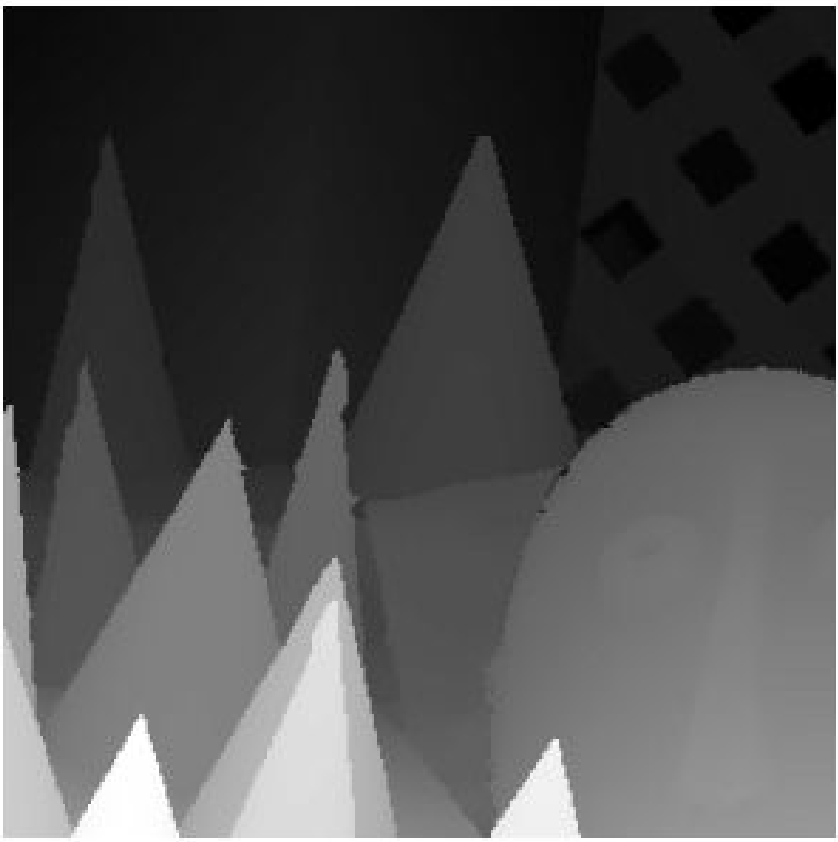


Figure 16

(b)

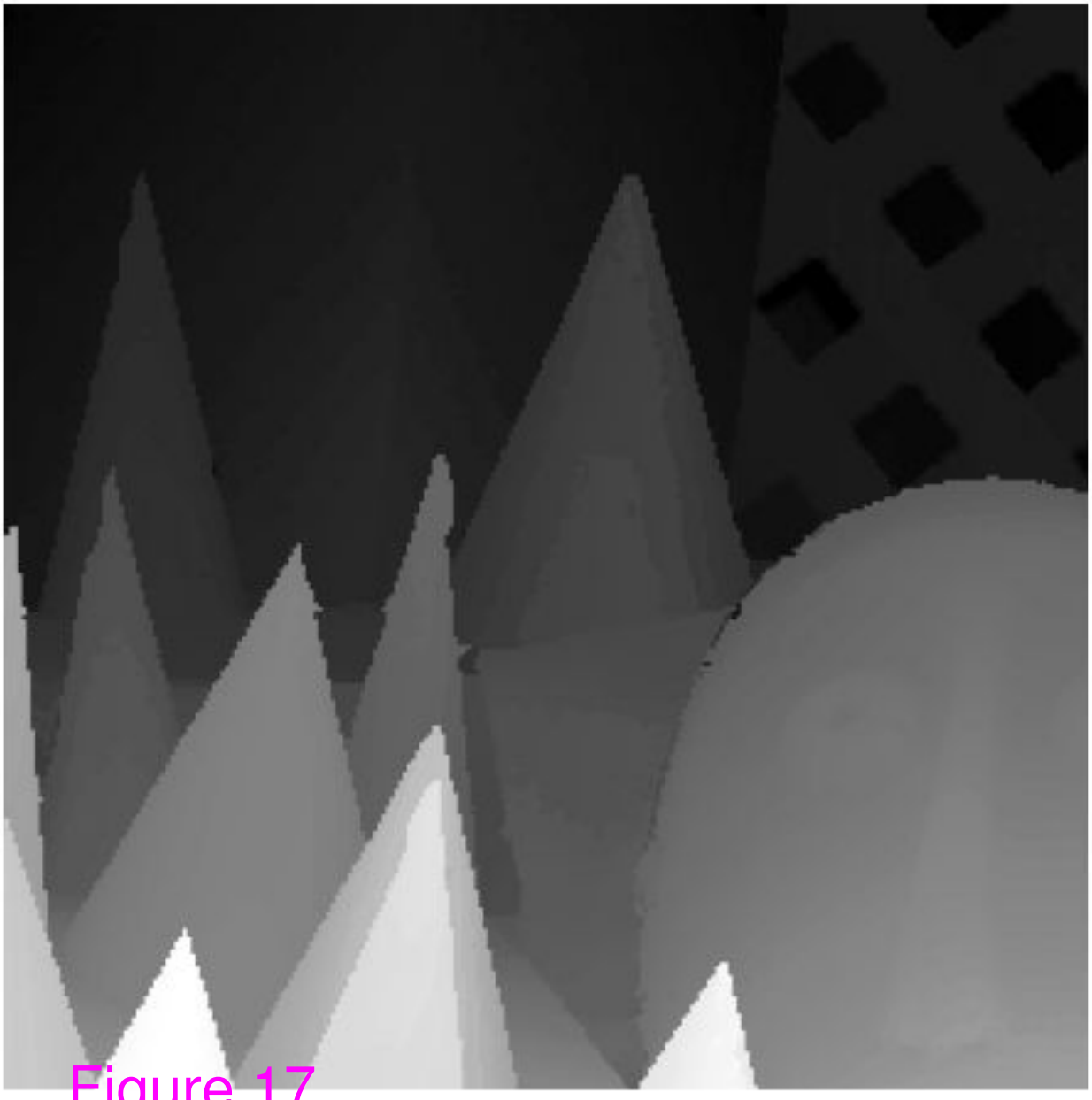
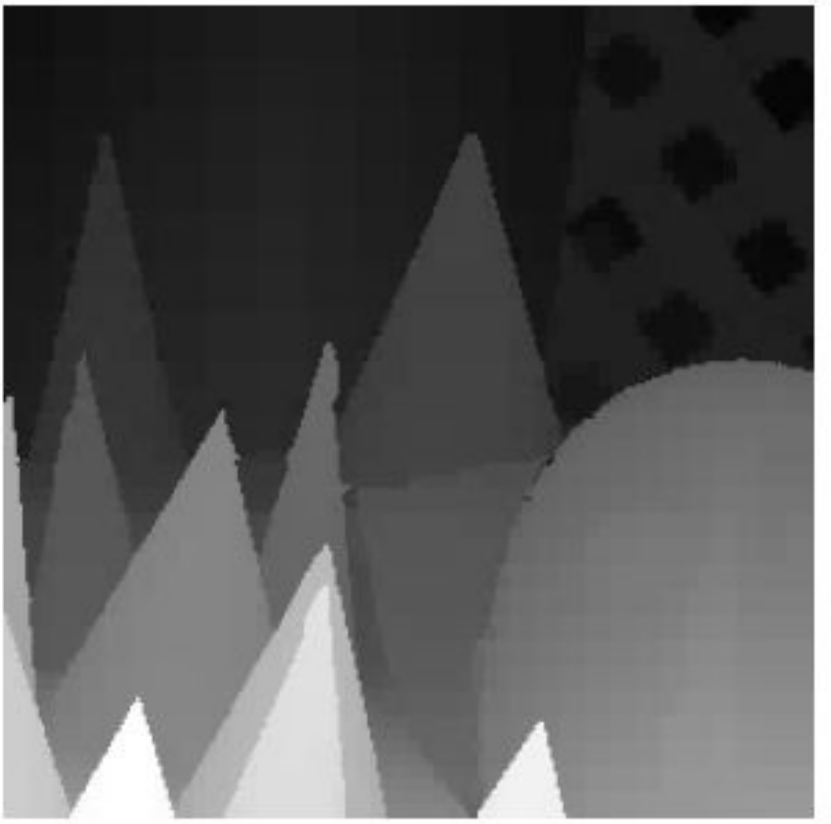
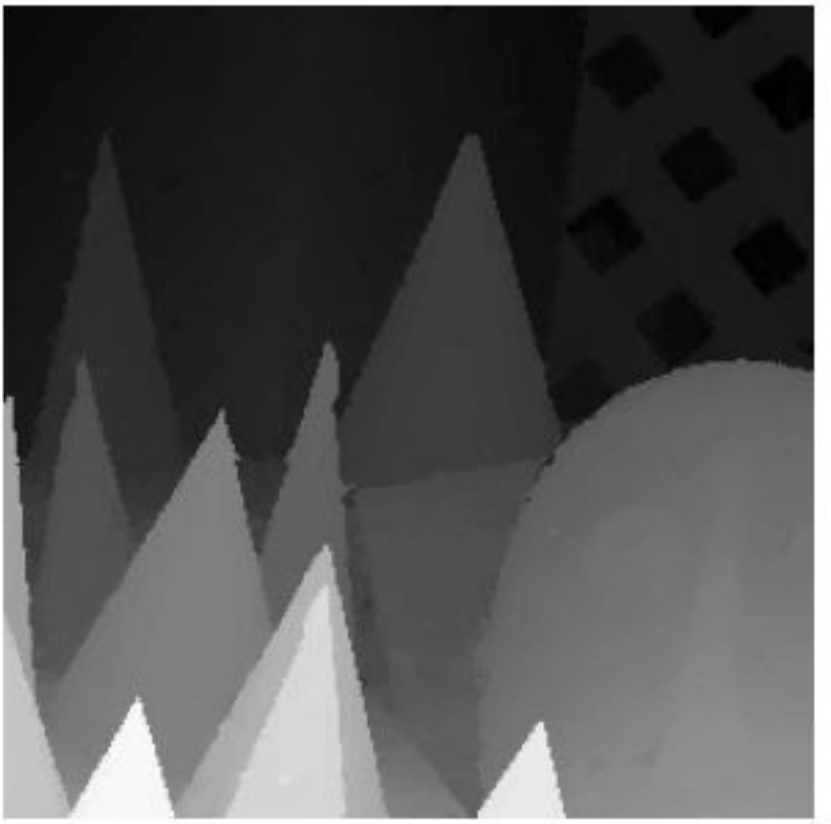


Figure 17



(a)



(b)

Figure 18