IMAGE RESTORATION AND RECONSTRUCTION USING PROJECTIONS ONTO EPIGRAPH SET OF CONVEX COST FUNCTIONS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

By

Mohammad Tofighi

July, 2015

Image Restoration and Reconstruction Using Projections onto Epigraph Set of Convex Cost Functions
By Mohammad Tofighi
July, 2015

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. A. Enis Çetin (Advisor)

Assoc. Prof. Dr. Sinan Gezici

Assist. Prof. Dr. B. Uğur Töreyin

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

ABSTRACT

IMAGE RESTORATION AND RECONSTRUCTION USING PROJECTIONS ONTO EPIGRAPH SET OF CONVEX COST FUNCTIONS

Mohammad Tofighi
M.S. in Electrical and Electronics Engineering
Advisor: Prof. Dr. A. Enis Çetin
July, 2015

This thesis focuses on image restoration and reconstruction problems. These inverse problems are solved using a convex optimization algorithm based on orthogonal Projections onto the Epigraph Set of a Convex Cost functions (PESC). In order to solve the convex minimization problem, the dimension of the problem is lifted by one and then using the epigraph concept the feasibility sets corresponding to the cost function are defined. Since the cost function is a convex function in $\mathbb{R}^N$, the corresponding epigraph set is also a convex set in $\mathbb{R}^{N+1}$. The convex optimization algorithm starts with an arbitrary initial estimate in $\mathbb{R}^{N+1}$ and at each step of the iterative algorithm, an orthogonal projection is performed onto one of the constraint sets associated with the cost function in a sequential manner. The PESC algorithm provides globally optimal solutions for different functions such as total variation, ℓ_1-norm, ℓ_2-norm, and entropic cost functions. Denoising, deconvolution and compressive sensing are among the applications of PESC algorithm. The Projection onto Epigraph Set of Total Variation function (PES-TV) is used in 2-D applications and for 1-D applications Projection onto Epigraph Set of ℓ_1-norm cost function (PES-ℓ_1) is utilized.

In PES-ℓ_1 algorithm, first the observation signal is decomposed using wavelet or pyramidal decomposition. Both wavelet denoising and denoising methods using the concept of sparsity are based on soft-thresholding. In sparsity-based denoising methods, it is assumed that the original signal is sparse in some transform domain such as Fourier, DCT, and/or wavelet domain and transform domain coefficients of the noisy signal are soft-thresholded to reduce noise. Here, the relationship between the standard soft-thresholding based denoising methods and sparsity-based wavelet denoising methods is described. A deterministic soft-threshold estimation method using the epigraph set of ℓ_1-norm cost function is presented. It is

demonstrated that the size of the ℓ_1-ball can be determined using linear algebra. The size of the ℓ_1-ball in turn determines the soft-threshold. The PESC, PES-TV and PES-ℓ_1 algorithms, are described in detail in this thesis. Extensive simulation results are presented. PESC based inverse restoration and reconstruction algorithm is compared to the state of the art methods in the literature.

Keywords: Convex optimization, epigraph of a convex cost functions, projection onto convex sets, total variation function, ℓ_1-norm function, denoising, deconvolution, compressive sensing.

ÖZET

DIŞBÜKEY MALİYET FONKSIYONLARI'NIN EPİGRAF KÜMESİNE DİK İZDÜŞÜMLER KULLANAN İMGE RESTORASYONU VE YENİDEN İNŞA ALGORİTMASI

Mohammad Tofighi
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Prof. Dr. A. Enis Çetin
Temmuz, 2015

Bu tez, imge restorasyonu ve yeniden inşası ile alakalı problemler üzerinedir. İmge restorasyonu ve yeniden inşa problemleri, Dışbükey Maliyet Fonksiyonları'nın Epigraf Kümesine Dik İzdüşümleri (PESC) ile çözülür. Dışbükey küçültme problemini çözmek için ilk adımda problemin boyutu bir artırılır ve ardından epigraf fikri kullanılarak maliyef fonksiyonlarının fizibilite kümeleri tanımlanır. Maliyet fonksiyonu $\mathbb{R}^N$ içerisinde olduğundan dolayı, ona karşılık gelen epigraf seti dışbükey de $\mathbb{R}^{N+1}$ içerisindedir. Dışbükey küçültme algoritması $\mathbb{R}^{N+1}$ içerisinde rastgele bir tahmin ile başlar ve yinelemeli algoritmanın her adımında birbirini takip eden şekilde, maliyet fonksiyonlarını kısıtlayan kümeler üzerine dik izdüşümler gerçekleştirir. PESC algoritması, tam değişim, ℓ_1-norm, ℓ_2-norm, entropik maliyet fonksiyonu gibi değişik bir çok fonksiyon için global en iyi çözümler verir. Tam Değişim Fonksiyonunun Epigraf Kümesi Üzerine İzdüşüm (PES-TV) 2 boyutlu uygulamalar için, ℓ_1-norm Fonksiyonunun Epigraf Kümesi Üzerine İzdüşüm (PES-ℓ_1) ise 1 boyutlu uygulamalar için değerlendirilmiştir.

PES-ℓ_1 algoritmasında, gözlemlenen sinyal ilk adımda dalgacık ve ya piramit ayrışımı kullanılarak dağılmıştır. Dalgacık tabanlı gürültüden arındırma ve diğer seyreklik tabanlı gürültüden arındırma teknikleri yumuşak eşiklendirmeye dayalıdır. Seyreklik tabanlı gürültüden arındırma metodlarında, asıl sinyalin, Fourier, DCT, ve ya dalgacık gibi herhangi bir dönüşüm uzayında, seyrek oldukları varsayılmaktadır ve gürültülü sinyalin dönüşüm uzayındaki katsayılarına yumuşak eşiklendirme uygulanır. Burada, standart yumuşak eşiklendirmeye dayalı gürültüden arındırma metodları ile seyreklik tabanlı dalgacık kullanarak gürültüden arındırma metodları açıklanmıştır. ℓ_1-norm maliyet

fonksiyonunun epigraf kümesini kullanan bir yumuşak eşik tahmin metodu sunulmuştur. Doğrusal cebir kullanarak ℓ_1 topunun büyüklüğünün belirlenebileceği gösterilmiştir. Yumuşak eşiği ℓ_1 topunun büyüklüğü belirlemektedir. PESC, PES-TV ve PES-ℓ_1 algoritmaları detaylı olarak anlatılmıştır. Kapsamlı benzetim sonuçları sunulmuştur. PESC tabanlı ters restorasyon ve yeniden inşa algoritması, edebiyattaki en gelişmiş tekniklerle karşılaştırılmıştır.

Anahtar sözcükler: Dışbükey optimizasyon, dışbükey maliyet fonksiyonları'nın epigrafı, dışbükey kümeler üzeri'ne izdüşüm, tam değişim fonksiyonu, ℓ_1-norm fonksiyonu, gürültüden arındırma, ters evrişi, sıkıştırılmış algılama.

Acknowledgement

First of all, I would like to express my gratitude to my supervisor, Prof. A. Enis Çetin for supporting me in all possible ways.

I am also very grateful to Prof. Orhan Arıkan, Prof. Sinan Gezici, Prof. Serdar Kozat, and Prof. Selim Aksoy for their valuable supports during my education in Bilkent.

I would also like to express my sincere thanks to Prof. Sinan Gezici and Prof. B. Uğur Töreyin for accepting to review this thesis and providing useful comments.

I want to thank Mürüvet Parlakay for her tireless and great manner in every moment in Electrical and Electronics Engineering Department.

I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for my scholarship under project grant 113E069 of TÜBİTAK.

I would like to thank

- Akbar Alipour, Ecem Bozkurt, Mehdi Dabirnia, Okan Demir, Behnam Ghasemiparvin, Özgecan Karabulut, Erdem Karagül, Gokçe Kuralay, Caner Odabaş, Dilara Oğuz, Amir Rahimzadeh, Damla Sarıca, Parisa Sharif, Burak Şahinbaş. Life is always tough, but sometimes it crashes you down. However, I had the chance to have these valuable friends around me to help me in every obstacle in my path to this moment. These are the precious people in Bilkent, who are like a family to me and I am going to miss them so much.

- First friends I have made in Bilkent EE department, who pave the way for me to adjust to the new environment and keep it up during first semesters, whom I will always be indebted to Elif Aydoğdu, and Alexander Suhre, Y. Hakan Habiboğlu, Kıvanç Köse.

- All my valuable friends, whom I had precious time during three years of my life in Bilkent with: Seher Acer, Volkan Açıkel, Başar Akbay, Mehmet Dedeoğlu, Aslan Etminan, Sebastian Hettenkofer, A. Nail İnal, Ramiz Kian, Onur Külçe, Samad Nadimi, Abdullah Öner, Hamed Rezanejad, Sina Rezaei, Alireza Sadegi, M. Ömer Sayin, Manoochehr Takrimi, İsmail Uyanık, Aslı Ünlügedik, and Aras Yurtman.

- Ali Ayremlou, and Hadi Shahmohammadi, who our friendship goes back to years ago and cosidering the far distance between us, their friendship and support continued up to now.

- My colleagues for helping me during two years of my M.Sc. in EE-310, C. Emre Akbaş, M. Tunç Arslan, Alican Bozkurt, Osman Günay, Oğuzhan Oğuz, R. Akın Sevimli, and Onur Yorulmaz.

In all stages of my life, I am indebted to my precious parents and sisters, Samira and Soheila, whom without their support, I wouldn't be able to reach this success. I dedicate this thesis to my family, which no words can express my gratitude to them and show their love and endless support to me.

Dedicated to my dearest family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Many signal and image restoration and reconstruction problems can be considered as inverse problems. In these problems we try to recover the original signal or image from observations which are usually corrupted by noise. Denoising, deconvolution, and compressive sensing are among the well-known inverse problems. In restoration methods, the aim is to get as much closer as possible to the original image. The distance between the estimated image and the original image is called the cost, and the function which measures this cost is the cost function. Therefore, in these methods the aim is to minimize a given cost function. Since most of the cost functions in such problems are convex functions, convex optimization algorithms can be considered to solve them. In the following sections a convex optimization method and its application to denoising, deconvolution, and compressive sensing problems are described.

1.1 Convex Optimization

Convex optimization studies the problem of minimizing the convex functions. In a convex function the objective and the constraint functions are convex, which

means they satisfy the following inequality:

$$f(\alpha \mathrm{x} + \beta \mathrm{y}) \leq \alpha f(\mathrm{x}) + \beta f(\mathrm{y}) \tag{1.1}$$

for all $\mathrm{x}, \mathrm{y} \in \mathbb{R}^N$ and all $\alpha, \beta \in \mathbb{R}$ with $\alpha + \beta = 1, \alpha \geq 0, \beta \geq 0$. Some inverse problems can be solved using convex optimization methods. Consider $Ax = b$, where A is matrix and x and b are vectors. Here the aim is to solve as inverse problem to find the solution x. The trivial solution would be $x = A^{-1}b$. However, every A will not be invertible, therefore, to solve this problem, pseudo-inverse of A would be required, then the solution would be $x = A^{+}b$.

Sometimes the inverse of the matrix A cannot be found directly, then optimization methods can be used for such inverse problems. In order to solve these problems, an objective function is defined. This function measures how close the obtained estimated solution from the optimization process, fits the observed data. This function is the cost function of the optimization problem. There are many cost functions used in inverse problems. The standard cost function $f(x)$ is usually of the following form:

$$f(\mathrm{x}) = \|b - A\mathrm{x}\|_2^2, \tag{1.2}$$

which $\|.\|_2^2$ is the ℓ_2-norm. The $f(x)$ is the ℓ_2-norm between the observed data and the predicted data.

Plenty of optimization methods are proposed according to the problem. Among them descent methods such as gradient descent method and steepest descent method, and the Newton's method are well-known [5]. All these nonlinear methods solve the optimization problems in an iterative manner, such that in each iteration the value of the cost function is measured and the aim is to obtain minimum cost (or maximum efficiency).

One of the methods used in convex optimization is Projection Onto Convex Sets (POCS). This method, similar to Descent methods, tries to find the minimum point on the cost function by iterative projections onto convex sets. A set $\mathcal{C}$ is convex if the line segment between any two points in $\mathcal{C}$ lies inside $\mathcal{C}$, i.e., if for

any $\mathrm{x}_1, \mathrm{x}_2 \in \mathcal{C}$ and any θ with $0 \leq \theta \leq 1$, we have:

$$\theta \mathrm{x}_1 + (1-\theta)\mathrm{x}_2 \in \mathcal{C}. \tag{1.3}$$

In POCS the projections are performed onto the sets. Sometimes projection onto the surface of a function is very hard. Therefore, to make it easier, projections can be performed onto the hyperplane passing through the points over the epigraph set of convex sets. A hyperplane can be defined as follows:

$$\mathcal{H} = \{\mathrm{x} | a^T \mathrm{x} = b\} \tag{1.4}$$

where $a \in \mathbb{R}^n$, $a \neq 0$, and $b \in \mathbb{R}$. Geometrically, the above hyperplanes can be interpreted as the set of the points with a constant inner product to a given vector a, as the normal vector. Considering these information, the POCS algorithm is described in the following section.

1.2 Projection Onto Convex Sets (POCS)

In this thesis, a new convex optimization algorithm based on orthogonal Projections onto the Epigraph Set of a Convex cost function (PESC) is introduced. This algorithm is based on standard POCS algorithm. In Bregman's standard POCS approach [6, 7], the algorithm converges to the intersection of convex constraint sets, as in Figure 1.1. In this section, it is shown that it is possible to use a convex cost function in a POCS based framework using the epigraph set and the new framework is used in many signal and image processing applications [8–13].

Bregman also developed iterative methods based on the so-called Bregman distance to solve convex optimization problems [11]. In Bregman's approach, it is necessary to perform a Bregman projection at each step of the algorithm, which may not be easy to compute the Bregman distance in general [10, 12].

In standard POCS approach, the goal is simply to find a vector, which is in the intersection of convex constraint sets [7, 14–35]. In each step of the iterative algorithm an orthogonal projection is performed onto one of the convex sets .

Bregman showed that successive orthogonal projections converge to a vector, which is in the intersection of all the convex sets, as in Figure 1.1. If the sets do not intersect, iterates oscillate between members of the sets [36, 37], as in Figure 1.2. Since, there is no need to compute the Bregman distance in standard POCS, it found applications in many practical problems.

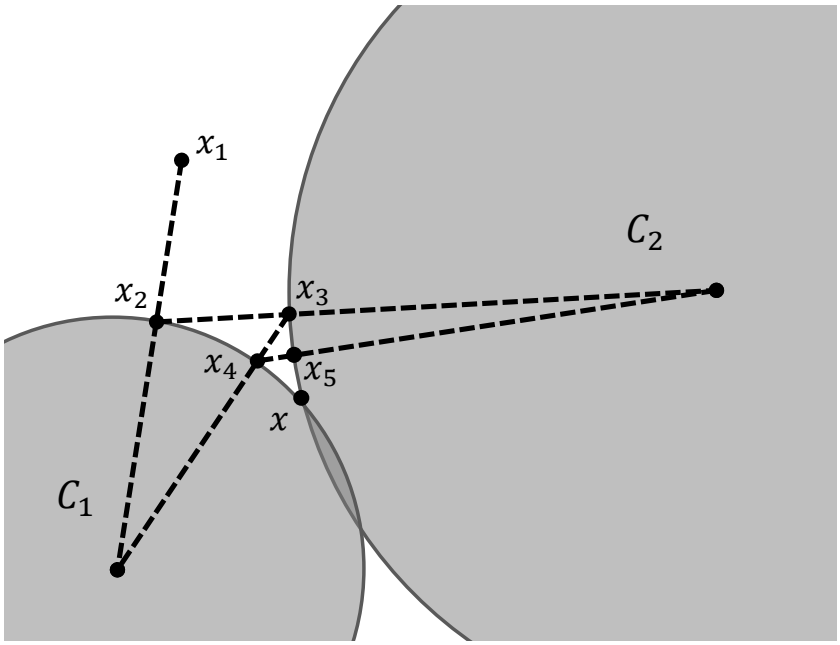


Figure 1.1: Sets $\mathcal{C}_1$ and $\mathcal{C}_2$ are two convex sets. The initial vector x_1 is sequentially projected onto the sets $\mathcal{C}_1$ and $\mathcal{C}_2$ to find the vector, x, in the intersection of these sets.

In PESC approach, the dimension of the signal reconstruction or restoration problem is lifted by one and sets corresponding to a given convex cost function are defined. This approach is graphically illustrated in Figure1.2. If the cost function is a convex function in $\mathbb{R}^N$, the corresponding epigraph set is also a convex set in $\mathbb{R}^{N+1}$. As a result, the convex minimization problem is reduced to finding the $[\mathbf{w}^*, f(\mathbf{w}^*)]$ vector over the epigraph set corresponding to the cost function as shown in Figure 1.2. As in standard POCS approach, the new iterative optimization method starts with an arbitrary initial estimate in $\mathbb{R}^{N+1}$ and an orthogonal projection is performed onto one of the constraint sets. The resulting

vector is then projected onto the epigraph set. This process is continued in a sequential manner at each step of the optimization problem. This method provides globally optimal solutions for convex cost functions, such as total-variation [38], filtered variation [9], ℓ_1-norm [39], and entropic function [16]. The iteration process is shown in Figure 1.2. Regardless of the initial value $\underline{\mathbf{w}}_0$, iterates converge to $[\mathbf{w}^*, f(\mathbf{w}^*)]$ pair as shown in Figure 1.2.

This Thesis is organized as follows. In Section 1.3, the epigraph of a convex cost function is defined and the convex minimization method based on the PESC approach is introduced. In Chapter 2, the TV based PESC algorithm is presented. In Chapter 3, the ℓ_1-norm based PESC algorithm is described. The new approach does not require a regularization parameter as in other TV based methods [15, 26, 38]. In Chapter 4, deconvolution using PESC is described. In Chapter 5, compressive sensing using PESC is introduced. At the end of each chapter, the simulation results are presented. Finally, this thesis is concluded in Chapter 6.

1.3 Projection Onto Epigraph Set of a Convex Cost Function (PESC)

Let us first consider a convex minimization problem

$$\min_{\mathbf{w}\in\mathbb{R}^N} f(\mathbf{w}), \tag{1.5}$$

where $f : \mathbb{R}^N \to \mathbb{R}$ is a convex cost function. We increase the dimension by one to define the epigraph set of f in $\mathbb{R}^{N+1}$ as follows:

$$\mathcal{C}_f = \{\underline{\mathbf{w}} = [\mathbf{w}^T \ y]^T : \ y \geq f(\mathbf{w})\}, \tag{1.6}$$

which is the set of $N+1$ dimensional vectors, whose $(N+1)^{st}$ component y is greater than $f(\mathbf{w})$. We use bold face letters for N dimensional vectors and underlined bold face letters for $N+1$ dimensional vectors, respectively. Another set that is related with the cost function $f(\mathbf{w})$ is the level set:

$$\mathcal{C}_s = \{\underline{\mathbf{w}} = [\mathbf{w}^T \ y]^T : \ y \leq 0, \ \underline{\mathbf{w}} \in \mathbb{R}^{N+1}\}, \tag{1.7}$$

where it is assumed that $f(\mathbf{w}) \geq 0$ for all $f(\mathbf{w}) \in \mathbb{R}$. Both $\mathcal{C}_f$ and $\mathcal{C}_s$ are closed and convex sets in $\mathbb{R}^{N+1}$. Other closed and convex sets describing a feature of the desired solution can be also used in this approach. Sets $\mathcal{C}_f$ and $\mathcal{C}_s$ are graphically illustrated in Figure 1.2. An important component of the PESC approach is to

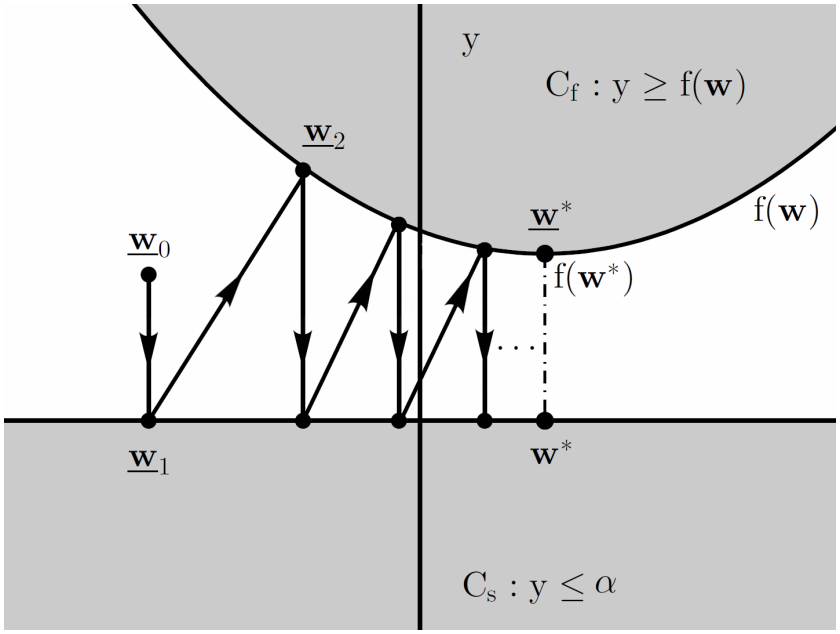


Figure 1.2: Two convex sets $\mathcal{C}_f$ and $\mathcal{C}_s$ corresponding to the convex cost function f. We sequentially project an initial vector $\underline{\mathbf{w}}_0$ onto $\mathcal{C}_s$ and $\mathcal{C}_f$ to find the global minimum, which is located at $\underline{\mathbf{w}}^* = [\mathbf{w}^* \ f(\mathbf{w}^*)]^T$.

perform an orthogonal projection onto the epigraph set. Let $\underline{\mathbf{w}}_1$ be an arbitrary vector in $\mathbb{R}^{N+1}$. The projection $\underline{\mathbf{w}}_2$ is determined by minimizing the distance between $\underline{\mathbf{w}}_1$ and $\mathcal{C}_f$, i.e.,

$$\underline{\mathbf{w}}_2 = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \|\underline{\mathbf{w}}_1 - \underline{\mathbf{w}}\|^2. \tag{1.8}$$

Equation (1.8) is the ordinary orthogonal projection operation onto the set $\mathcal{C}_f \in \mathbb{R}^{N+1}$. In order to solve the problem in Eq. (1.8), we do not need to compute the Bregman's so-called D-projection or Bregman projection. Projection onto the set $\mathcal{C}_s$ is trivial. We simply force the last component of the $N+1$ dimensional vector to zero. In the PESC algorithm, iterates eventually oscillate between the two nearest vectors of the sets $\mathcal{C}_s$ and $\mathcal{C}_f$ as shown in Figure 1.2. As a result, we

obtain

$$\lim_{n\to\infty} \underline{\mathbf{w}}_{2n} = [\mathbf{w}^* \ f(\mathbf{w}^*)]^T, \tag{1.9}$$

where $\mathbf{w}^*$ is the N dimensional vector minimizing $f(\mathbf{w})$. The proof of Eq. (1.9) follows from Bregman's POCS theorem [6]. It was generalized to non-intersection case by Gubin et. al [36]. Since the two closed and convex sets $\mathcal{C}_s$ and $\mathcal{C}_f$ are closest to each other at the optimal solution case, iterations oscillate between the vectors $[\mathbf{w}^* \ f(\mathbf{w}^*)]^T$ and $[\mathbf{w}^* \ 0]^T$ in $\mathbb{R}^{N+1}$ as n tends to infinity. It is possible to increase the speed of convergence by non-orthogonal projections [27].

If the cost function f is not convex and have more than one local minimum, then the corresponding set $\mathcal{C}_f$ is not convex in $\mathbb{R}^{N+1}$. In this case, the iterates may converge to one of the local minima.

In current TV based denoising methods [38, 40], the following cost function is used:

$$f(\mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|^2 + \lambda \mathrm{TV}(\mathbf{w}), \tag{1.10}$$

where $\mathbf{v}$ is the observed signal. The solution of this problem can be obtained using the method in an iterative manner, by performing successive orthogonal projections onto $\mathcal{C}_f$ and $\mathcal{C}_s$, as discussed above. In this case, the cost function is $f(\mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|_2^2 + \lambda \mathrm{TV}(\mathbf{w})$. Therefore,

$$\mathcal{C}_f = \{\mathbf{w} \in \mathbb{R}^{N+1} : \|\mathbf{v} - \mathbf{w}\|^2 + \lambda \mathrm{TV}(\mathbf{w}) \leq y\}. \tag{1.11}$$

The denoising solutions that we obtained are very similar to the ones found by Chambolle's in [38] as both methods use the same cost function. One problem in [38] is the estimation of the regularization parameter λ. One has to determine the λ in an ad-hoc manner or by visual inspection. In Chapter 1, new denoising methods with a different TV based cost function and ℓ_1-norm cost function are described. The new method with TV function does not require a regularization parameter. Concept of epigraph is first used in signal reconstruction problems in [41, 42]. We also independently developed epigraph based algorithms in [43].

As mentioned before, a hyperplane is in the form of $\mathcal{H} = \{\mathrm{x}|a^T\mathrm{x} = b\}$. This hyperplane can be interpreted in the following form:

$$\mathcal{H} = \{\mathrm{x}|a^T(\mathrm{x} - \mathrm{x}_0) = 0\}, \tag{1.12}$$

where x_0 is any point over the hyperplane (which satisfies $a^T\mathrm{x}_0 = b$). The projection onto a hyperplane $a^T\mathrm{x} = b$ with normal a can easily be computed using simple algebra. The projection is as follows:

$$\mathrm{x}_p = \mathrm{x} + \frac{b - a^T\mathrm{x}}{\|a\|_2^2}a, \tag{1.13}$$

where $\|.\|_2^2$ is the Euclidean norm. This operation is illustrated in Figure The convex optimization application in image reconstruction is described in the following sections.

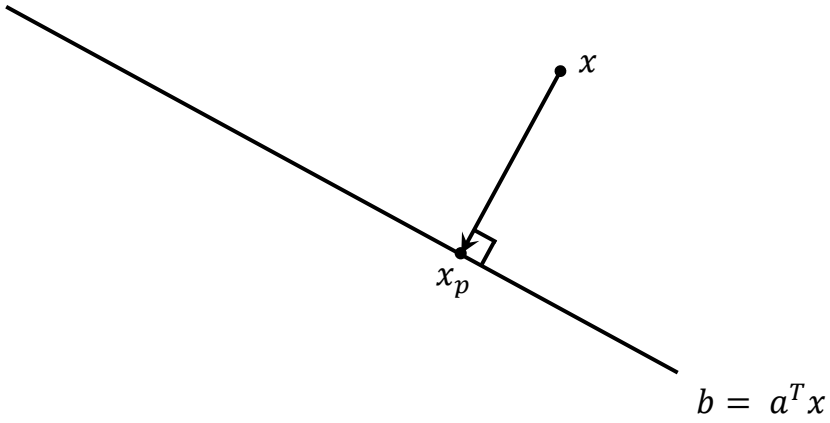


Figure 1.3: Graphical illustration of projection onto a hyperplane.

1.4 Denoising

Denoising refers to removing unwanted signal from the original signal while the important information of the original signal is preserved as much as possible. Many signal and image denoising methods are proposed in signal processing literature in the past decades. However, the study in this field is open and many researchers are focused over the issue of denoising signals and images under various conditions.

The denoising problem can typically be studied under optimization problems,

in which appropriate objective function is minimized under some certain constraints. For instance, in [8, 26, 38] a denoising algorithm based on the Total Variation (TV) function as the constraint is proposed. The idea of minimizing the TV for image denoising was first suggested in [8]. In [8], the noisy image is the addition of original image and random Gaussian noise with estimated variance equal to σ^2. Therefore, the aim is to solve the following minimization problem:

$$\min\{\mathrm{TV}(\mathbf{w}) : \|\mathbf{w} - \mathbf{w}_{\mathrm{orig}}\|^2 = N^2\sigma^2\}, \tag{1.14}$$

where N^2 is the total number of the pixels.

In [26], for image with sharp contours and block features, the following restoration problem is studied:

$$\min\{\mathrm{TV}(\mathbf{w}) + \lambda\|\mathbf{w} - \mathbf{v}\|^2 = N^2\sigma^2\} \quad \lambda \geq 0, \tag{1.15}$$

where λ is the regularization parameter. Finding the exact λ is a computationally expensive issue. Therefore, it is determined in an add-hoc manner.

In [44] and in this thesis [13,43,45], the same minimization in 1.15 is considered. The problem in 1.15 is split into two constraints as:

$$\min\{\|\mathbf{w} - \mathbf{v}\|^2\} \quad \text{such that} \quad \mathrm{TV}(\mathbf{w}) \leq \tau, \tag{1.16}$$

where τ is a positive constraint bound on TV value in [44]. However, in our method in [13, 43, 45], there is no need to define a constraint on the TV value, since the TV value of the obtained image converges to the TV value of the original image. It can be inferred that, both (1.15) and (1.16) are equivalent for some specific values of the regularization parameters. However, in [44], the adjustment of λ parameter is eliminated, and instead τ is required sufficient adjustment which is easier compared to defining λ. In the proposed method in [44], the authors use a proximal algorithm and epigraph projection to solve minimization problem.

In [46], a denoising algorithm based on 3D filtering of similar image blocks is proposed. In this algorithm, the similar blocks of the noisy image are grouped together using block matching methods and a 3D array is obtained. Then this

arrays are denoised using a 3D collaborative Wiener filter. Then the denoised blocks are combined together to reconstruct the image.

In [47], an adaptive data-driven threshold for denoising images via wavelet soft-thresholding is proposed and it claims that lossy compression can also be used for denoising. The reason for this claim is that a lossy compression such as quantization with zero-zero is similar to soft-thresholding.

In [48], a denoising algorithm based on interscale orthonormal wavelet thresholding is proposed. In this algorithm, the denoising process is parameterized directly as the summation of basic nonlinear processes in which their weights are unknown. Then to solve the denoising problem, the estimate of the mean square error between original image and noisy image is minimized. However, they do not use the original image to estimate MSE. They use an accurate, statistically unbiased, MSE estimate which is quadratic in the unknown weights. They use the Stein's Unbiased Risk Estimate (SURE) which is similar to a priori estimate of the MSE resulting from an arbitrary processing of noisy data. In this algorithm the thresholding is performed in discrete wavelet domain.

In [1], multivariate wavelet denoising is combined with Principle Component Analysis (PCA). Wavelet denoising methods are popular for 1D signal denoising. The proposed algorithm in [1] is also used for 1D signal denoising. This work deals with regression models such as $\mathbf{w} = \mathbf{w}_{\text{orig}} + \boldsymbol{\xi}$, where the observation $\mathbf{w}$ is p-dimensional, and $\boldsymbol{\xi}$ is the additive noise. In this method, PCA is used to detect the insignificant components of the signal and enhance the denoising process by eliminating those components of the wavelet coefficients.

In this thesis, we propose a convex optimization method based on Projections onto Epigraph Set of Convex Cost function (PESC) to solve inverse problems such as denoising, deconvolution, and compressive sensing. The PESC method is used to solve the denoising problem similar to (1.16). The PESC algorithm is used both for 2D signals (images) and 1D signals. Total variation cost function is used for 2D denoising (Projections onto Epigraph Set of TV (PES-TV)) and ℓ_1-norm cost function is used for 1D signals denoising (Projections onto Epigraph

Set of ℓ_1-Ball (PES-ℓ_1)). The PESC method is illustrated in detail in Chapter 2. The simulation results for comparing the PESC algorithm with other algorithms are also provided at end of Chapter 2.

1.5 Deconvolution

Deconvolution is the act of reversing the effect of the convolution. In other words, the deconvolution algorithms try to reconstruct the signals which are convolved together. In image processing applications, usually one of the signals is the original signal and the second one is the blurring signal which degrades the quality of the original signal. The deconvolution algorithms found applications in many fields of image processing, i.e., medical image processing. For instance, the images obtained from microscopes has the focusing problem and are usually blurred. The aim of deconvolution algorithms is to enhance the quality of these images as much as possible.

In [49, 50], Vonesch *et al.* proposes a deconvolution algorithm based on a Fast Thresholding Landweber (FTL) algorithm. This algorithm minimizes a quadratic data term subject to a regularization on the ℓ_1-norm of the wavelet coefficients of the solution. In this approach, it is assumed that the PSF is known.

We propose a deconvolution algorithm based on PESC algorithm. In this method, two constraint sets are defined and the projection onto these sets are performed to obtain the deblurred image. The first set is the set of hyperplanes obtained from the deconvolution problem, and the intersection of these hyperplanes is the deconvolution solution. In order to speed up the deconvolution process and enhance the quality of the output image, we impose the TV constraint using Projection onto Epigraph Set of TV function (PES-TV). This deconvolution method is presented in detail in Chapter 4. The simulation results are also presented in 4.2.

1.6 Compressive Sensing

According to Shannon/Nyquist sampling theorem, in order to avoid losing information during sampling, transformation, and reconstruction, the sampling rate should be equal or greater than two times of the signal bandwidth. However, in many applications, increasing the sampling rate is very expensive [39]. Therefore, many methods are studied during last decades to solve this problem. The Compressive Sensing (SC) theory is proposed according to sparse nature of the signal as a possible solution. Sparsity expresses the idea that a signal can be represented with much smaller amount of components than suggested by its bandwidth. In other words, CS exploits the fact that many natural signals are sparse and compressible in the sense that they have shorter representation in a proper transform domain.

In [51], Matching Pursuit (MP) algorithm is proposed. According to this algorithm any signal is decomposed into a linear expansion of waveforms that belong to a redundant function dictionary. Then in selection of the waveforms the aim is to find the best match for the signal structures. In adaptive signal representations, matching pursuits are the general procedure. Therefore, an interpretation of the signal structures are provided by matching pursuit decomposition. This algorithm is a greedy algorithm. It chooses a waveform which is best adopted to an approximate of a part of the signal, in an iterative manner. Matching pursuits are very flexible in signal representations, because they have unlimited choice of dictionaries.

In [52], the Compressive Sensing Matching Pursuit (CoSaMP) algorithm is proposed. CoSaMP is an iterative recovery algorithm for CS problems. This algorithm recovers the signal from its noisy samples using four inputs. These inputs are: observation matrix, a vector of (noisy) samples of the unknown signal, the sparsity level of the signal to be produced (s), and a stopage criterion. In the first step, it forms a proxy of the residual from the current samples and determines the largest ones. Then using these samples it updates the current approximation. The algorithm solves a least square problem to approximate the updated signal.

Then it preserves the largest entries in the least squares signal approximation. Then the samples are updated to reflect the residual. This is performed according to Algorithm 1 in [52].

In [53], the ℓ_p-norm optimization based CS reconstruction algorithm is proposed. Considering Φ as an $M \times N$ measurment matrix, and $\Phi\mathbf{w} = b$ the vector of an N-dimensional signal $\mathbf{w}$. This algorithm solves the CS problem by solving the minimization problem as $\mathbf{w}^\star = \min_{\mathbf{w}} \|\mathbf{w}\|_p^p$ subject to $\Phi\mathbf{w} = b$, which $\mathbf{w}^\star$ is the reconstructed signal.

Considering that the CS problem is a convex inverse problem, we can apply PESC algorithm to such problems. In this approach, two sets are defined, and the combination of these two sets leads to the CS problem's solution. The first set is the set of hyperplanes defined in CS problems, which are the observation hyperplanes. The intersection of these hyperplanes is the solution or the reconstructed signal. The second set, which imposes the TV constraint to the estimated image at each step of the iterations enhances the performance of the PES-TV algorithm. This algorithm is illustrated in detail in Chapter 5, and the simulation results are also presented in 5.3.

Chapter 2

Denoising Using Projection onto Epigraph Set of Total Variation Function (PES-TV)

Denoising refers to the process of reducing noise in a given signal, image and video. The basic idea of projection-based denoising algorithm is described in Chapter 1. As mentioned before, Projection onto Epigraph Set of Convex Cost function (PESC), can be used for denoising 2D signals. The Total Variation (TV) cost function is used for denoising 2D signals. In Section 2.1, the Projection onto Epigraph Set of TV function (PES-TV) is presented.

2.1 The PES-TV Algorithm

In this section, we present a new denoising method, based on the epigraph set of the TV function. Let the original signal or image be $\mathbf{w}_{orig}$ and its noisy version be $\mathbf{v}$. Suppose that the observation model is the additive noise model:

$$\mathbf{v} = \mathbf{w}_{\text{orig}} + \boldsymbol{\eta}, \tag{2.1}$$

where $\boldsymbol{\eta}$ is the additive noise. In this approach, we solve the following problem for denoising:

$$\underline{\mathbf{w}}^{\star} = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \|\underline{\mathbf{v}} - \underline{\mathbf{w}}\|^2, \tag{2.2}$$

where $\underline{\mathbf{v}} = [\mathbf{v}^T \ 0]$ and $\mathcal{C}_f$ is the epigraph set of TV or FV in $\mathbb{R}^{N+1}$. The TV function, which we used for an $M \times M$ discrete image $\mathbf{w} = [\mathrm{w}^{i,j}], \quad 0 \leq i, j \leq M-1 \ \in \mathbb{R}^{M \times M}$ is as follows:

$$TV(\mathbf{w}) = \sum_{i,j} (|\mathrm{w}^{i+1,j} - \mathrm{w}^{i,j}| + |\mathrm{w}^{i,j+1} - \mathrm{w}^{i,j}|). \tag{2.3}$$

The minimization problem (2.2) is essentially the orthogonal projection onto the set $\mathcal{C}_f = \{\mathbf{w} \in \mathbb{R}^{N+1} : TV(\mathbf{w}) \leq y\}$. This means that we select the nearest vector $\underline{\mathbf{w}}^{\star}$ on the set $\mathcal{C}_f$ to $\mathbf{v}$. This is graphically illustrated in Figure 2.1. Let us explain the projection onto an epigraph set of a convex cost function ϕ in detail. Equation (2.2) is equivalent to:

$$\underline{\mathbf{w}}^{\star} = \begin{bmatrix} \mathbf{w}_p \\ \phi(\mathbf{w}_p) \end{bmatrix} = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \| \begin{bmatrix} \mathbf{v} \\ 0 \end{bmatrix} - \begin{bmatrix} \mathbf{w} \\ \phi(\mathbf{w}) \end{bmatrix} \|, \tag{2.4}$$

where $\underline{\mathbf{w}}^{\star} = [\mathbf{w}_p^T, \phi(\mathbf{w}_p)]$ is the projection of $[\mathbf{v}, 0]$ onto the epigraph set. The projection $\underline{\mathbf{w}}^{\star}$ must be on the boundary of the epigraph set. Therefore, the projection must be on the form $[\mathbf{w}_p^T, \phi(\mathbf{w}_p)]$. Equation (2.4) becomes:

$$\underline{\mathbf{w}}^{\star} = \begin{bmatrix} \mathbf{w}_p \\ \phi(\mathbf{w}_p) \end{bmatrix} = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \|\mathbf{v} - \mathbf{w}\|_2^2 + \phi(\mathbf{w})^2. \tag{2.5}$$

In the case of total variation $\phi(\mathbf{w}) = TV(\mathbf{w})$. It is also possible to use $\lambda\phi(.)$ as a the convex cost function and Eq. 2.5 becomes:

$$\underline{\mathbf{w}}^{\star} = \begin{bmatrix} \mathbf{w}_p \\ \phi(\mathbf{w}_p) \end{bmatrix} = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \|\mathbf{v} - \mathbf{w}\|_2^2 + \lambda^2\phi(\mathbf{w})^2. \tag{2.6}$$

Actually, Combettes and Pesquet and other researchers including us used a similar convex set in denoising and other signal restoration applications [9,26,40, 42]. The following convex set in $\mathbb{R}^N$ describes all signals whose TV is bounded by an upper bound ϵ:

$$\mathcal{C}_f = \{\mathbf{w} : \mathrm{TV}(\mathbf{w}) \leq \epsilon\}. \tag{2.7}$$

The parameter ϵ is a fixed upper bound on the total variation of the signal and it has to be determined a priori in an ad-hoc manner. On the other hand we do not specify a prescribed number on the TV of vectors in the epigraph set based approach. The upper bound on TV is automatically determined by the orthogonal projection onto $\mathcal{C}_f$ from the location of the corrupted signal as shown in Figure 2.1.

In current TV based denoising methods [38, 40] the following cost function is used:

$$f(\mathbf{w}) = \|\mathbf{v} - \mathbf{w}\|_2^2 + \lambda \mathrm{TV}(\mathbf{w}). \tag{2.8}$$

The solution of (2.8) can be also obtained using the method that we discussed in Section 1.3. Similar to the LASSO approach [54] a major problem with this approach is the estimation of the regularization parameter λ. One has to determine the λ in an ad-hoc manner or by visual inspection. It is experimentally observed that Eq. (2.6) produces good denoising results when $\lambda = 1$. Experimental results are presented in Section 2.3.1. During this orthogonal projection operations, we

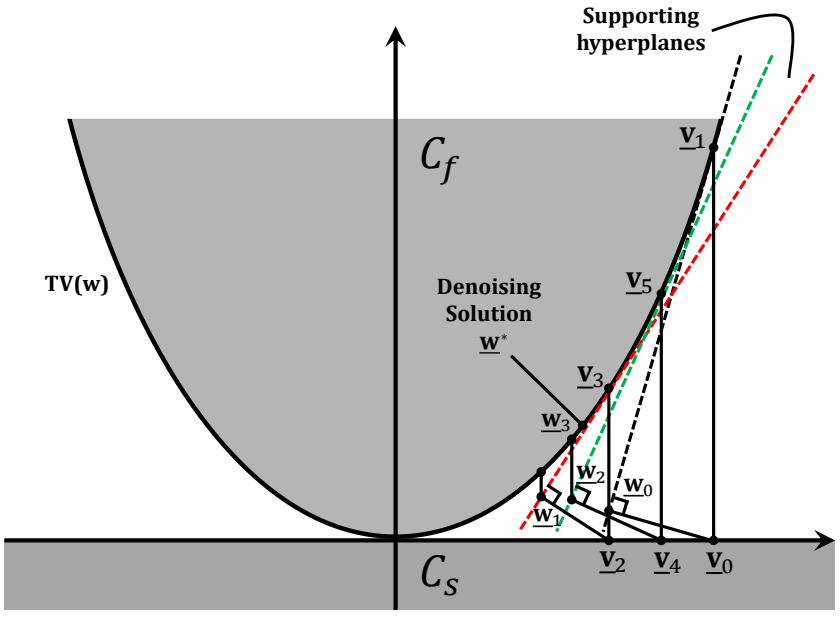


Figure 2.1: Graphical representation of the minimization of Eq. (2.2), using projections onto the supporting hyperplanes of $\mathcal{C}_f$. In this problem the sets $\mathcal{C}_s$ and $\mathcal{C}_f$ intersect because $TV(\mathbf{w}) = 0$ for $\mathbf{w} = [0, 0, ..., 0]^T$ or for a constant vector.

do not require any parameter adjustment as in [38].

2.1.1 Implementation of PES-TV

The projection operation described in Eq. (2.2) can not be obtained in one step when the cost function is TV. The solution is determined by performing successive orthogonal projections onto supporting hyperplanes of the epigraph set $\mathcal{C}_f$. In the first step, $\mathrm{TV}(\mathbf{v}_0)$ and the surface normal at $\underline{\mathbf{v}}_1 = [\mathbf{v}_0^T \quad \mathrm{TV}(\mathbf{v}_0)]$ in $\mathbb{R}^{N+1}$ are calculated. In this way, the equation of the supporting hyperplane at $\underline{\mathbf{v}}_1$ is obtained. The vector $\underline{\mathbf{v}}_0 = [\mathbf{v}_0^T \quad 0]$ is projected onto this hyperplane and $\underline{\mathbf{w}}_0$ is obtained as our first estimate as shown in Figure 2.1. In the second step, $\underline{\mathbf{w}}_0$ is projected onto the set $\mathcal{C}_s$ by simply making its last component zero. The TV of this vector and the surface normal, and the supporting hyperplane is calculated as in the previous step. We calculate the distance between $\underline{\mathbf{v}}_0$ and $\underline{\mathbf{w}}_i$ at each step of the iterative algorithm described in the previous paragraph. The distance $\|\underline{\mathbf{v}}_0 - \underline{\mathbf{w}}_i\|^2$ does not always decrease for high i values. This happens around the optimal denoising solution $\underline{\mathbf{w}}^\star$. Once we detect an increase in $\|\underline{\mathbf{v}}_0 - \underline{\mathbf{w}}_i\|^2$, we perform a refinement step to obtain the final solution of the denoising problem. In refinement step, the supporting hyperplane at $\underline{\mathbf{v}}_{2i-1} = \frac{\underline{\mathbf{v}}_{2i-5}+\underline{\mathbf{v}}_{2i-3}}{2}$ is used in the next iteration. For instance, when $\underline{\mathbf{v}}_2$ is projected, the distance is increased, therefore, in $i = 0$ in Figure 2.1, instead of $\underline{\mathbf{v}}_3$, vector $\underline{\mathbf{v}}_5$ will be used in next step. Next, $\underline{\mathbf{v}}_4$ is projected onto the new supporting hyperplane, and $\underline{\mathbf{w}}_2$ is obtained. In Figure 2.1, by projecting the $\underline{\mathbf{w}}_2$ onto $\mathcal{C}_f$, the vector $\underline{\mathbf{w}}_3$ is obtained which is very close to the denoising solution $\underline{\mathbf{w}}^\star$. In general iterations continue until $\|\underline{\mathbf{w}}_i - \underline{\mathbf{w}}_{i-1}\| \leq \epsilon$, where ϵ is a prescribed number, or iterations can be stopped after a certain number of iterations. A typical convergence graph is shown in Figure 2.2 for the "note" image. It is possible to obtain a smoother version of $\underline{\mathbf{w}}^\star$ by simply projecting $\underline{\mathbf{v}}$ inside the set $\mathcal{C}_f$ instead of the boundary of $\mathcal{C}_f$. The PES-TV algorithm is evaluated by comparison with well-known denoising algorithms. The simulation results are presented in 2.3.1.

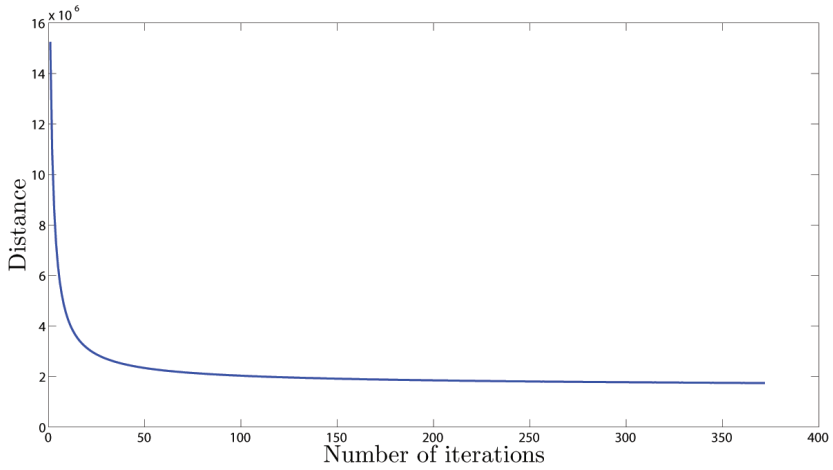


Figure 2.2: Euclidian distance from $\mathbf{v}_0$ to the epigraph of TV at each iteration, with noise standard deviation of $\sigma = 30$.

2.2 Denoising Images Corrupted by Impulsive Noise Using 3D Block Maching, 3D Wiener Filtering, and the PESC algorithm

The Block Matching 3D (BM3D) denoising algorithm [46], is introduced by Dabov *et al.* This method outperforms almost all the denoising algorithms proposed up to now in two-dimensional (2D) for Gaussian noise. However, it is unable to denoise an image corrupted by impulsive noise. We modified this algorithm using PES-TV algorithm to denoise the images corrupted both by Gaussian noise and impulsive noise. This algorithm is a two step method which is presented in the following sections.

2.2.1 Two Step Denoising Framework

A novel algorithm for denoising images that are corrupted by impulsive noise is presented. Impulsive noise generates pixels which their gray level values are not consistent with the neighboring pixels. The proposed denoising algorithm

is a two step procedure. In the first step, image denoising is formulated as a convex optimization problem, whose constraints are defined as limitations on local variations between neighboring pixels. Projections onto the epigraph set of TV function (PES-TV) are performed in the first step. Unlike similar approaches in the literature, the PES-TV method does not require any prior information about the noise variance. The first step is only capable of utilizing local relations among pixels. It does not fully take advantage of correlations between spatially distant areas of an image with similar appearance. In the second step a Wiener filtering approach is cascaded to the PES-TV based method to take advantage of global correlations in an image. In this step, the image is first divided into blocks and blocks with similar content are jointly denoised using a 3D Wiener filter. The denoising performance of the proposed two-step method was compared against three state of the art denoising methods under various impulsive noise models.

In the first step, local variations among neighboring pixel values are minimized in order to remove the impulsive components of the observed image. The first step does not fully take advantage of the correlation between distant areas of an image with similar appearance, e.g., blue sky region covering all the top portions of an image, cheek of a facial image and even textural regions of a shirt. In the second stage of the denoising method similar image blocks are determined using a block matching algorithm and they are denoised using Wiener filtering as in [46].

The first step of the proposed algorithm is based on Projections onto the Epigraph Set of the Total Variation function (PES-TV) [13, 14, 45]. In the PES-TV approach, the denoising operation is formulated as an orthogonal projection problem in which the input image is projected onto the epigraph set of the Total Variation (TV) function. This stage produces the initial (basic estimate) for second stage. In the second stage, the clock matching algorithm uses this basic estimate to group similar blocks more accurately. After block matching and obtaining the coordinates of the similar blocks, the second stage uses these coordinates to group the blocks of the noisy image. Later these 3D arrays of similar blocks are denoised using 3D Wiener filtering.

In [46], Dabov *et al.* proposed Block-Matching 3D filtering (BM3D) denoising

method that can utilize the correlation between similar areas of the image by jointly denoising them together. BM3D seems to be the best image denoising method for images corrupted by Gaussian noise [6,7,9,17,25,26,28–30,36,38,40,42, 44, 46, 48, 55–58]. BM3D is also a two-stage algorithm. However the first stage of BM3D requires an estimate of the noise variance beforehand to determine the hard thresholding level used in the first stage. Hard-thresholding based method fails to produce a good estimate of the image under impulsive noise in the first stage. As a result, the second stage of the BM3D does not produce a reliable denoised image when the noise is impulsive. On the other hand, the PES-TV denoising method does not need an estimate of the noise variance. It does not require any parameter adjustment, either. When we combine the second part of BM3D with the PES-TV approach, we get better results than ordinary BM3D approach for images corrupted by impulsive noise and very similar results for Gaussian noise. The noise information is more effective in the first step of the denoising algorithms than in the second step. An approximate estimation of the noise variance is enough for performing denoising in the second step. Furthermore, with an appropriate denoised image obtained in the first step, estimated variance of the noise for second step will also be more reliable. The First step is introduced in 2.1, and the second step is illustrated in the following section.

2.2.2 Second Step of the Denoising Framework: Block Matching And Collaborative Filtering

The second step of the proposed denoising method is the "3D" approach introduced by Dabov *et al.* [46]. The output of the PES-TV based denoising stage is fed into the "3D" Block Matching (BM) step of BM3D.

In natural images, spatially distant areas/blocks are correlated with each other. However, most denoising algorithms do not exploit this fact and only consider local pixel variations in an image. Dabov *et al.* introduced block matching and collaborative Wiener filtering concepts in a denoising framework to take advantage of similarities between spatially distant blocks in an image. They

first group similar looking regions in an image by block matching. Then, they denoise all those regions together using a three dimensional (3D) approach called collaborative Wiener filtering. We borrow this procedure from [46] and use it as the second step of our denoising scheme as shown in Figure 2.3. In this section, we briefly review the BM3D denoising method.

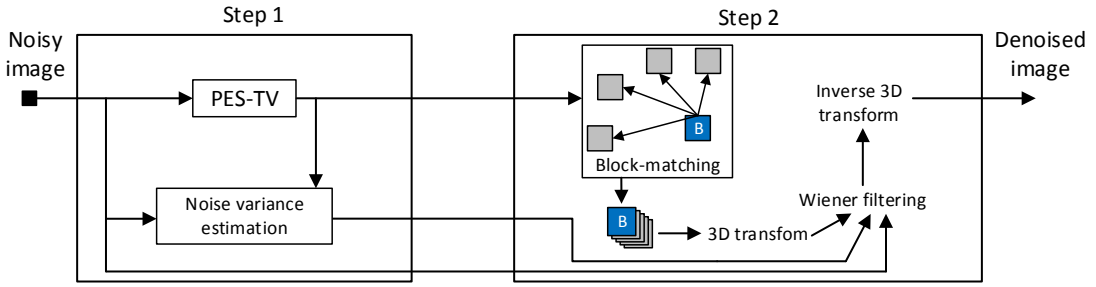


Figure 2.3: Graphical representation of the proposed two stage denoising process.

2.2.3 Block Matching

First the PES-TV denoised image is divided into non-overlapping regions of fixed size called reference blocks (B_R). Then each reference block is compared against candidate blocks of similar appearance (B_C) using the following equation:

$$d(B_R, B_C) = \frac{\|B_R - B_C\|_2^2}{N}, \tag{2.9}$$

where $N = M^2$ is the number of pixels in each block. Blocks satisfying the similarity condition are grouped together to construct 3D arrays of Similar Blocks (SB). The set of blocks satisfying the condition of block matching threshold are grouped together. This set is as follows:

$$G_{SB_R} = \{c \in \mathbf{w}_{TV-rec} : d(B_R, B_C) \leq \tau_{th}\} \tag{2.10}$$

where c represents the coordinate of blocks in the reconstructed image obtained by the PES-TV stage, $\mathbf{w}_{TV-rec}$ is the reconstructed image in the PES-TV stage, and τ_{th} is the block matching threshold. This threshold is determined according

to deterministic speculations based on the denoised image in the first step [46]. Each set G_{SB_R} is an $N \times N_{G_{SB_R}}$ 3D array of similar blocks, where $N_{G_{SB_R}}$ is the number of blocks in the set G_{SB_R}.

2.2.4 Collaborative Filtering

The 3D arrays obtained by block matching have both spatial and "temporal" similarity. Therefore, the noise can be efficiently removed by the collaborative 3D Wiener filtering. Wiener shrinkage coefficients for the set of blocks are determined from the 3D transform coefficient as follows:

$$W_{G_{SB_R}} = \frac{|T(G^{BE}_{SB_R})|^2}{|T(G^{BE}_{SB_R})|^2 + \sigma^2}, \tag{2.11}$$

where $G^{BE}_{SB_R}$ is the 3D array for similar blocks from Basic Estimate (BE), which is the output of the PES-TV step, $T(.)$ is the transformation operator, $|T(G^{BE}_{SB})|^2$ is the power spectrum of the basic estimate image, and σ^2 is the variance of the noise which is estimated from the difference image obtained by subtracting the observed image and image obtained in the first stage. After obtaining the coefficients, the collaborative filtering is realized by element wise multiplication of $W_{G_{SB_R}}$ by the 3D arrays of noisy image using the coordinates obtained in PES-TV stage, $G^n_{SB_R}$, as follows:

$$\mathbf{w}^{wie}_{rec} = T^{-1}(W_{G_{SB_R}} T(G^n_{SB_R})). \tag{2.12}$$

After filtering the 3D array, inverse transform and aggregation operation [46] is performed to get the final denoised image. The overall process is explained graphically in Figure 2.3. The simulation results for PES-TV, and BM3D with PES-TV and other algorithms are presented in Section 2.3.1, and 2.3.2, respectively.

2.3 Simulation Results

2.3.1 Denoising Using PES-TV

The PES-TV algorithm is tested with a wide range of images. Let us start with the "Note" image shown in Figure 2.6a. This is corrupted by a zero mean Gaussian noise with $\sigma = 45$ in Figure 2.6b. The image is restored using PES-TV, SURE-LET [48], and Chambolle's algorithm [38] and the denoised images are shown in Figure 2.6c, 2.6d, and 2.6e, with SNR values equal to 15.08, 13.20, and 11.02 dB, respectively. SURE-LET and Chambolle's algorithm produce some patches of gray pixels at the background. The regularization parameter λ in Eq. (1.10) is manually adjusted to get the best possible results for each image and each noise type and level in [38], and SURE-LET require the knowledge about noises standard deviation in [48]. Moreover, Structural Similarity Index (SSIM) is also calculated as in [59] for all methods. PES-TV algorithm not only produces higher SNR and SSIM values than other methods, but also provides visually better looking image. The same experiments are also done over "cancer cell" image, which the results are presented in Figure 2.7. Denoising results for other noise levels are presented in Table 2.1. We also tested the PES-TV algorithm against ϵ-contaminated Gaussian noise (salt-and-pepper noise) with the PDF of

$$f(x) = \epsilon\phi(\frac{x}{\sigma_1}) + (1-\epsilon)\phi(\frac{x}{\sigma_2}), \tag{2.13}$$

where $\phi(x)$ is the standard Gaussian distribution with mean zero and unit standard deviation. The results of the tests are presented in Table 2.3. The performance of the reconstruction is measured using the SNR criterion, which is defined as follows

$$\text{SNR} = 20 \times log_{10}(\frac{\|\mathbf{w}_{orig}\|}{\|\mathbf{w}_{orig} - \mathbf{w}_{rec}\|}), \tag{2.14}$$

where $\mathbf{w}_{orig}$ is the original signal and $\mathbf{w}_{rec}$ is the reconstructed signal. All the SNR values in Tables are in dB.

To evaluate the performance of the PES-TV algorithm, it is also possible to

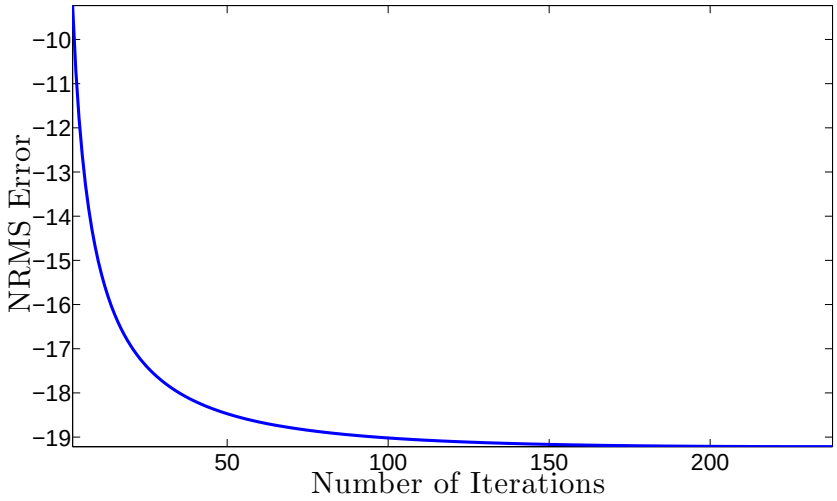


Figure 2.4: Normalized root mean square error in each iteration for "Note" image corrupted with Gaussian noise with $\sigma = 25$.

use Normalized Root Mean Square Error metric as

$$\text{NRMSE}(i) = \frac{\|\mathbf{w}_i - \mathbf{w}_{orig}\|}{\|\mathbf{w}_{orig}\|} \quad i = 1, ..., N, \tag{2.15}$$

where N is the number of the iterations, $\mathbf{w}_i$ is the denoised image in i^{th} step, and $\mathbf{w}_{\text{orig}}$ is the original image. NRMSE is used to illustrate the convergence of the PES-TV based denoising algorithm as is used in [26] . As shown in Figure 2.4, NRMSE value decreases as the iterations proceeds while denoising the "Note" image corrupted with Gaussian noise ($\sigma = 25$). For the same image another convergence metric called Normalized Total Variation (NTV), which is defined in [26] as

$$\text{NTV}(i) = \frac{\text{TV}(\mathbf{w}_i)}{\text{TV}(\mathbf{w}_{\text{orig}})} \quad i = 1, ..., N, \tag{2.16}$$

where $\mathbf{w}_i$ and $\mathbf{w}_{\text{orig}}$ are the restored image in i^{th} iteration, and the original image, respectively. As an indicator of the successful convergence of the PES-TV algorithm, the NTV curve converges to 1 in Figure 2.5, which means that the TV value of the output image converges to the TV value of the original image using PES-TV algorithm. In Figure 2.2, error value in each iteration step versus i is shown. These three curves show that iterations converge to a solution roughly

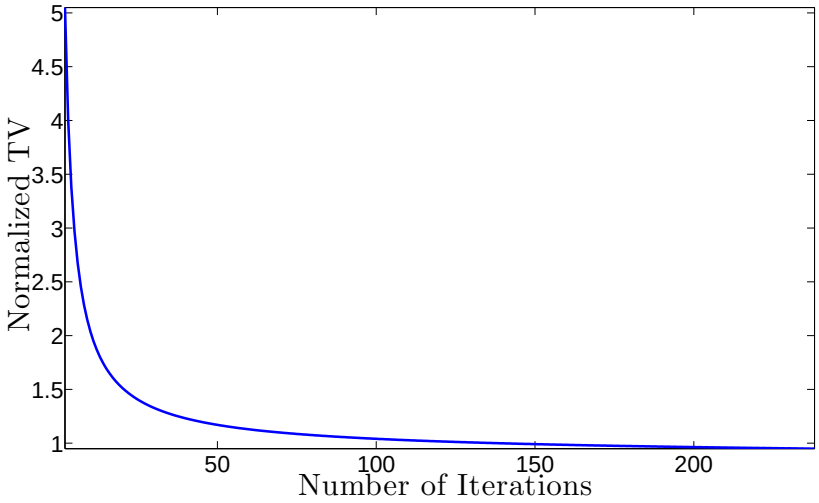


Figure 2.5: Normalized total variation in each iteration for "Note" image corrupted with Gaussian noise with $\sigma = 25$.

around 100^{th} iteration in "Note" image, which corrupted by independent Gaussian noise with $\sigma = 25$. In Table 2.2, denoising results for 34 images including 10 well-known test images from image processing literature and 24 images from Kodak Database [60], with different noise levels are presented. In almost all cases PES-TV method produces higher SNR and SSIM results than [38, 48].

2.3.2 Denoising Using BM3D and PESC

The basic estimate, which is obtained in the first step, affects the main denoising process in Wiener filtering step. In BM3D approach, first step requires the knowledge of the variance of the noise, however for impulsive noise the exact variance is unknown. Therefore this step fails to generate an appropriate estimate for impulsive noises for second step. Through the PES-TV approach [13] we bring solution to these issues.

Table 2.1: Comparison of the results for denoising algorithms with Gaussian noise for "note" image.

Noise σ	Input		**PES-TV**		Chambolle [38]		SURE-LET [48]	
	SNR	SSIM	SNR	SSIM	SNR	SSIM	SNR	SSIM
5	21.12	0.2201	30.63	0.2367	29.48	0.2326	27.42	0.2212
10	15.12	0.2037	25.93	0.2290	24.89	0.2213	22.20	0.2086
15	11.56	0.1917	22.91	0.2216	21.76	0.2141	19.13	0.1999
20	9.06	0.1825	20.93	0.2165	19.55	0.2065	16.95	0.1867
25	7.14	0.1716	19.27	0.2111	17.73	0.2006	15.34	0.1810
30	5.59	0.1636	17.89	0.2102	16.43	0.1950	13.93	0.1767
35	4.21	0.1565	16.68	0.2073	15.23	0.1903	12.87	0.1706
40	3.07	0.0.1488	15.90	0.2030	14.07	0.1855	11.77	0.1645
45	2.05	0.1407	15.08	0.1984	13.20	0.1815	11.02	0.1606
50	1.12	0.1332	14.25	0.1909	12.19	0.1766	10.17	0.1862
Average	8.00	0.1712	**19.95**	**0.2107**	18.45	0.2004	16.08	0.1862

The impulsive noise changes the pixel values in the image as follows:

$$\mathrm{v}_0^{i,j} = \begin{cases} \mathrm{v}^{i,j}, & \text{if } \mathrm{x} < l \\ i_{\min} + y(i_{\max} - i_{\min}), & \text{if } \mathrm{x} > l \end{cases} \tag{2.17}$$

where $\mathrm{v}^{i,j}$ is the $(i,j)^{th}$ pixel in the original image, $x, y \in [0,\ 1]$ are two uniformly distributed random variable, l is the parameter to determine the pixels to corrupt with noise, and $i_{\max}$ and $i_{\min}$ are the severity of the noise [61]. The salt & pepper noise and the ϵ-contaminated Gaussian noise are two types of impulsive noises. The ϵ-contaminated Gaussian noise is widely used to represent impulsive noise [55, 62]. The ϵ-contaminated Gaussian noise model is as follows:

$$\mathrm{v}_0^{i,j} = \begin{cases} \eta_1^{i,j}, & \text{with probability } 1-\epsilon \\ \eta_2^{i,j}, & \text{with probability } \epsilon \end{cases} \tag{2.18}$$

where $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ are independent Gaussian noise sources with variances σ_1^2 and σ_2^2, respectively. We assume that $\sigma_1 \ll \sigma_2$, and ϵ is a small positive number [57]. The reconstruction performance is measured using the Signal-to-Noise Ratio (SNR) as in 2.14 and Peak-SNR (PSNR) criterion, which is defined as follows:

$$PSNR = 20 \times log_{10}(\frac{max(\mathbf{w}_{orig})}{\|\mathbf{w}_{orig} - \mathbf{w}_{rec}\|_2/N}), \tag{2.19}$$

Table 2.2: Comparison of the results for denoising algorithms under Gaussian noise with standard deviations of σ.

Images	σ	Input SNR	**PES-TV**	Chambolle [38]	SURE-LET [48]
House	30	13.85	27.60	27.13	27.38
House	50	9.45	24.61	24.36	24.59
Lena	30	12.95	23.85	23.54	23.92
Lena	50	8.50	21.68	21.37	21.38
Mandrill	30	13.04	19.98	19.64	20.56
Mandrill	50	8.61	17.94	17.92	18.22
Living room	30	12.65	21.33	20.88	21.29
Living room	50	8.20	19.34	19.05	19.19
Lake	30	13.44	22.19	21.86	22.23
Lake	50	8.97	20.26	19.90	20.07
Jet plane	30	15.57	26.31	25.91	26.49
Jet plane	50	11.33	24.07	23.54	24.10
Peppers	30	12.65	24.24	23.59	23.78
Peppers	50	8.20	22.05	21.36	21.82
Pirate	30	12.13	21.43	21.30	21.27
Pirate	50	7.71	19.58	19.43	19.32
Cameraman	30	12.97	24.20	23.67	24.58
Cameraman	50	8.55	21.80	21.22	22.06
Flower	30	11.84	21.97	20.89	17.20
Flower	50	7.42	19.00	18.88	13.21
24-Kodak(ave.)	30	11.92	21.05	20.80	20.92
24-Kodak(ave.)	50	7.48	18.97	18.58	18.88
Average±std	30	12.27±1.66	**23.12**±2.35	22.66±2.34	22.70±2.91
Average±std	50	7.84±1.67	**20.85**±2.17	20.26±3.13	20.51±2.07

where $\mathbf{w}_{orig}$ is the original signal, $\mathbf{w}_{rec}$ is the reconstructed signal, and N is the total number of pixels in image.

Denoising results for "Note" image with ϵ-contaminated noise are summarized in Table 2.3. In this toy example, the PES-TV approach produces the best results. The denoising results for a set of 34 images including 10 well-known test images from image processing literature and 24 images from Kodak Database [63], which are corrupted by ϵ-contaminated noise with $\sigma_1 = 5$ and $\epsilon = 0.1$, and $\sigma_2 \in [30, 80]$ are presented in Tables 2.5 and 2.6 for PES-TV and BM3D algorithms, respectively. In this case, the noise is the combination of two Gaussian noises

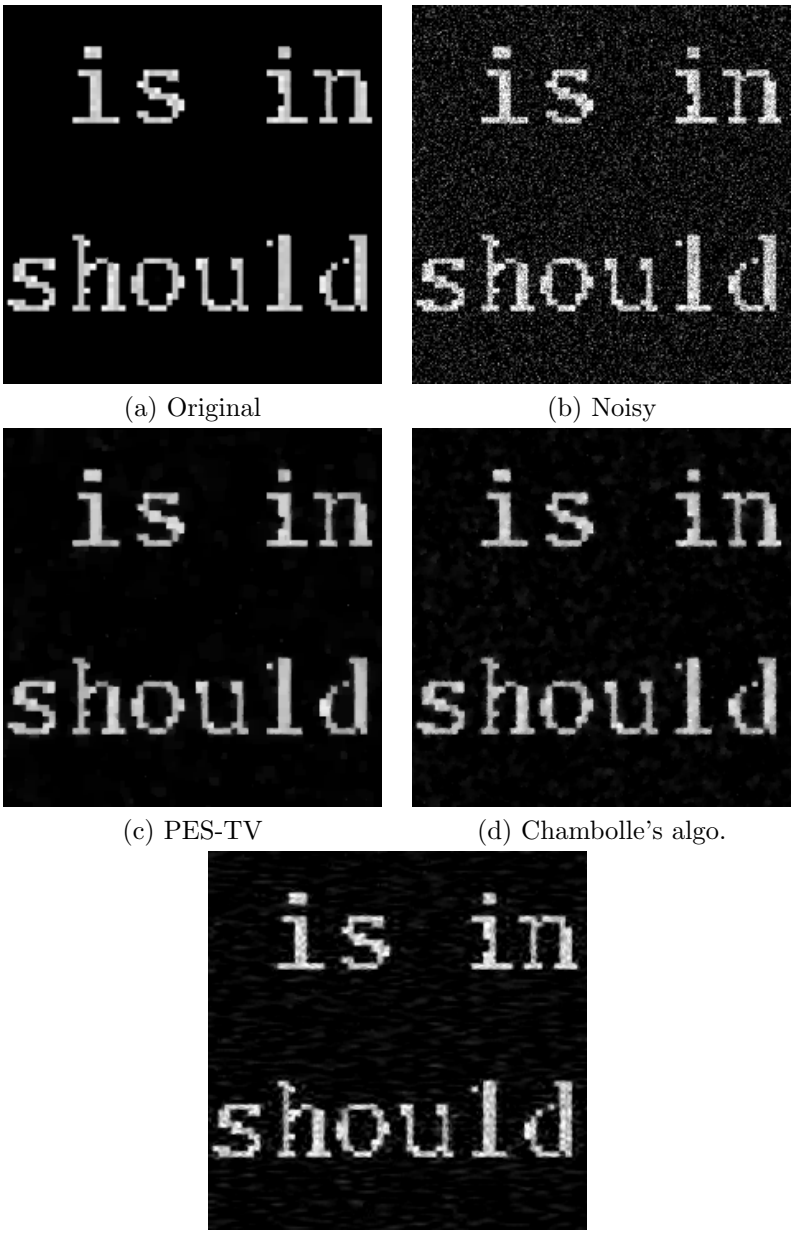
with different variances, therefore it can not be exactly modeled using a single variance parameter. The PES-TV algorithm performs better and produces higher PSNR values compared to all other denoising results obtained using [38, 46, 48], because it does not require knowledge of variance of the noise. We also present an additional illustrative example in Figure 2.9.

In another set of experiments, images that are corrupted by a mixture of salt & pepper and Gaussian noises are denoised using the proposed algorithm and also with BM3D and BM3D with median filtering for comparison purposes. The salt & pepper impulsive noise model is as follows:

$$\mathrm{v}_0^{i,j} = \begin{cases} s_{\min}, & \text{with probability } p \\ s_{\max}, & \text{with probability } q \\ \mathrm{v}^{i,j}, & \text{with probability } 1-p-q \end{cases} \tag{2.20}$$

where $\mathrm{v}^{i,j}$ is the gray level pixel value of the original image, $[s_{\min},\ s_{\max}]$ are the dynamic range of the original image, $s_{\min} \leq \mathrm{v}^{i,j} \leq s_{\max}$ for all (i,j) values, $\mathrm{v}_0^{i,j}$ is the gray level pixel value of the noisy image, $r = p+q$ defines the noise level [64].

The density of the salt & pepper noise is set to 0.02 and 0.05 and Gaussian noise is added with different variances. Results for this set of experiments are shown in Table 2.7 and 2.8, respectively. In almost all cases the PSNR values for PES-TV algorithm are higher than other algorithms. In Table 2.7 and 2.8 an α-trimmed mean filter [65] is used before processing. The third column refers to median filtering followed by second stage (3D Wiener filtering) of the BM3D algorithm ($\mathrm{BM3D}_M$).

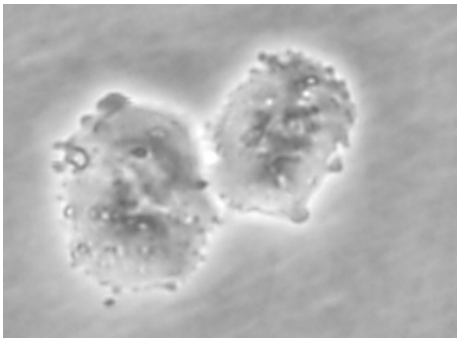


(a) Original (b) Noisy

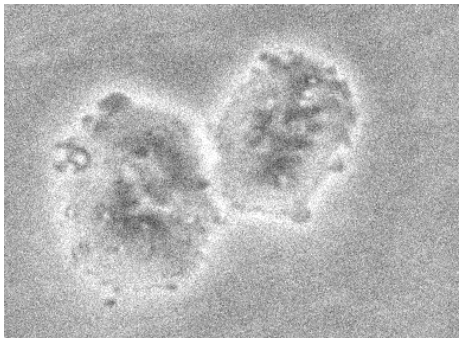
(c) PES-TV (d) Chambolle's algo.

(e) SURE-LET

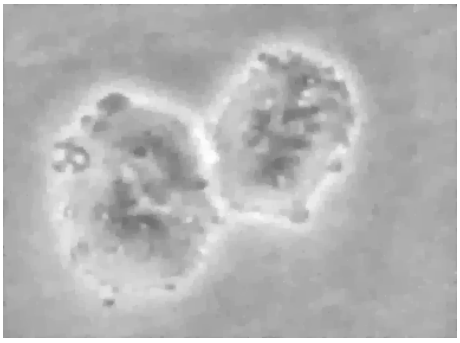
Figure 2.6: (a) A portion of original "Note" image, (b) image corrupted with Gaussian noise with $\sigma = 45$, denoised images, using: (c) PES-TV; SNR = 15.08 dB and SSIM = 0.1984, (d) Chambolle's algorithm; SNR = 13.20 dB and SSIM = 0.1815, (e) SURE-LET; SNR = 11.02 dB and SSIM = 0.1606. Chambolle's algorithm and SURE-LET produce some patches of gray pixels at the background.



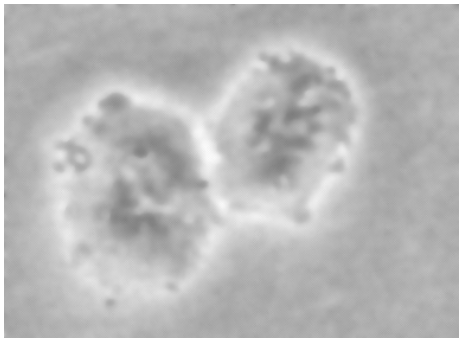
(a) Original



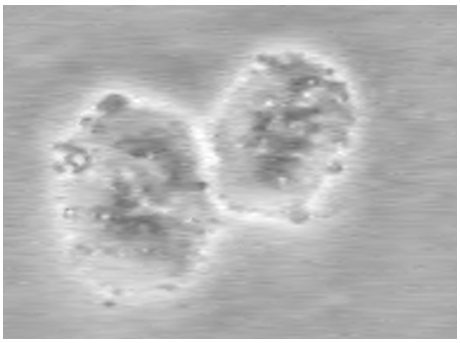
(b) Noisy



(c) PES-TV



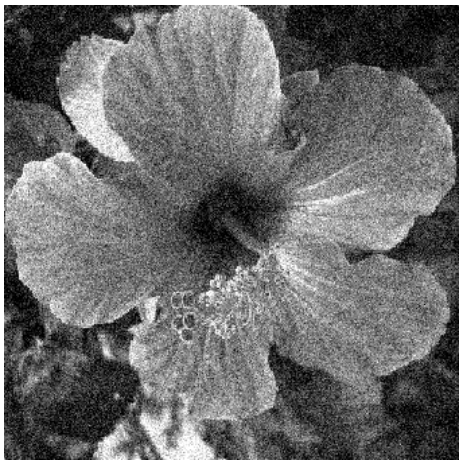
(d) Chambolle's algo.



(e) SURE-LET

Figure 2.7: (a) Original "Cancer cell" image, (b) image corrupted with Gaussian noise with $\sigma = 20$, denoised image, using: (c) PES-TV; SNR = 32.31 dB and SSIM = 0.5182, (d) Chambolle's algorithm; SNR = 31.18 dB and SSIM = 0.3978, (e) SURE-LET algorithm; SNR = 31.23 dB and SSIM = 0.4374.

(a) Original



(b) Noisy

(c) PES-TV

(d) Chambolle's algo.

Figure 2.8: "Flower" image experiments: experiments (a) Original "Flower" image, (b) "Flower" image corrupted with Gaussian noise with $\sigma = 30$, (c) Denoised "Flower" image, using PES-TV algorithm; SNR = 21.97 dB, (d) Denoised "Flower" image, using Chambolle's algorithm; SNR = 20.89 dB.

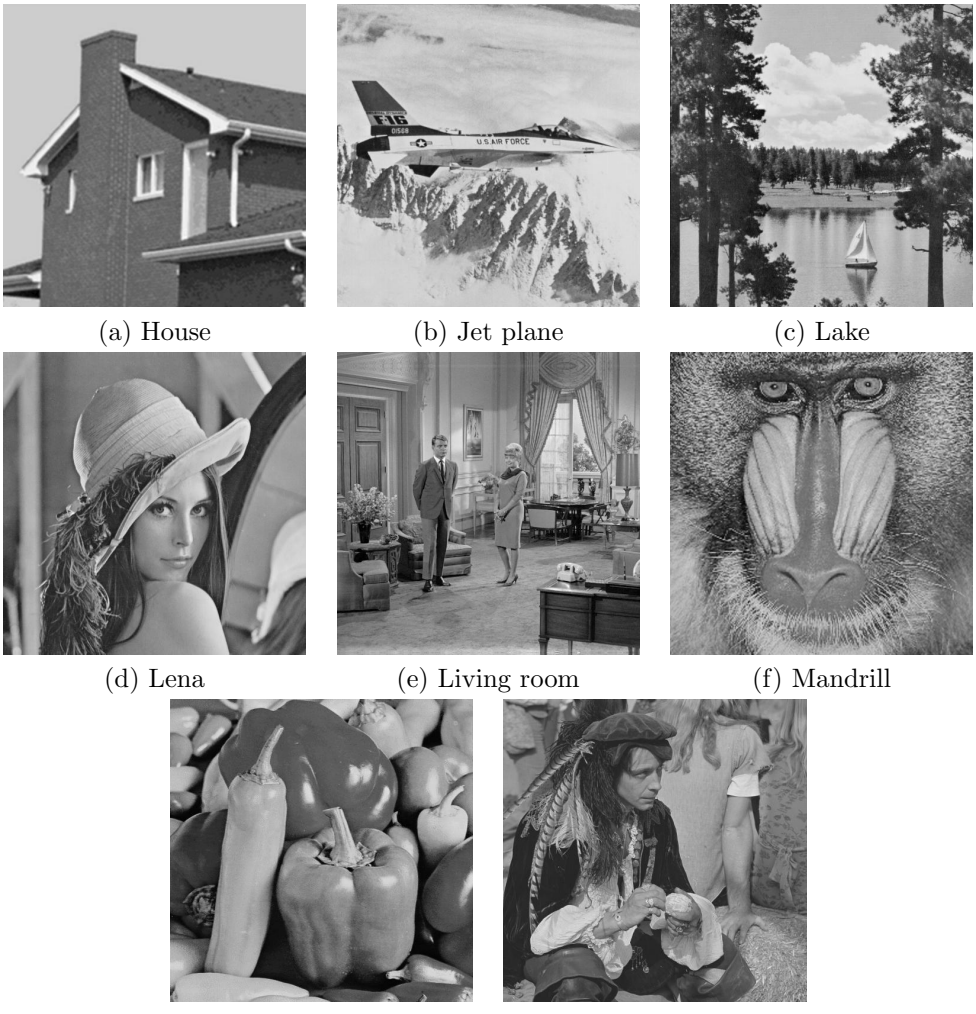
Table 2.3: Comparison of the results for denoising algorithms for ϵ-Contaminated Gaussian noise for "note" image

ϵ	σ_1	σ_2	Input SNR	**PES-TV**	Chambolle [38]	SURE-LET [48]
0.9	5	30	14.64	23.44	22.26	16.11
0.9	5	40	12.55	21.39	20.32	13.65
0.9	5	50	10.75	19.49	18.63	11.64
0.9	5	60	9.29	17.61	17.37	10.25
0.9	5	70	7.98	16.01	16.24	8.91
0.9	5	80	6.89	14.54	14.97	7.88
0.9	10	30	12.56	22.88	21.71	17.06
0.9	10	40	11.13	21.00	19.97	14.26
0.9	10	50	9.85	19.35	18.46	12.20
0.9	10	60	8.58	17.87	17.10	10.69
0.9	10	70	7.52	16.38	16.03	9.18
0.9	10	80	6.46	15.05	15.12	8.14
0.95	5	30	16.75	24.52	23.78	19.12
0.95	5	40	14.98	22.59	21.54	16.62
0.95	5	50	13.41	20.54	19.91	14.62
0.95	5	60	12.10	18.72	18.63	13.11
0.95	5	70	10.80	17.13	17.50	11.71
0.95	5	80	9.76	15.63	16.38	10.54
0.95	10	30	13.68	23.79	22.62	19.34
0.95	10	40	12.66	22.09	21.12	17.06
0.95	10	50	11.71	20.65	19.60	15.16
0.95	10	60	10.72	19.10	18.30	13.40
0.95	10	70	9.82	17.59	17.22	12.11
0.95	10	80	8.92	16.12	16.45	10.91

(a) (b)

(c) (d)

Figure 2.9: "Cameraman" image experiments: (a) Detail from the original "Cameraman" image, (b) "Cameraman" image corrupted with Gaussian noise with $\sigma = 50$, (c) Denoised "Cameraman" image, using PES-TV algorithm; SNR = 21.55 dB, (d) Denoised "Cameraman" image, using Chambolle's algorithm; SNR = 21.22 dB.



(a) House (b) Jet plane (c) Lake

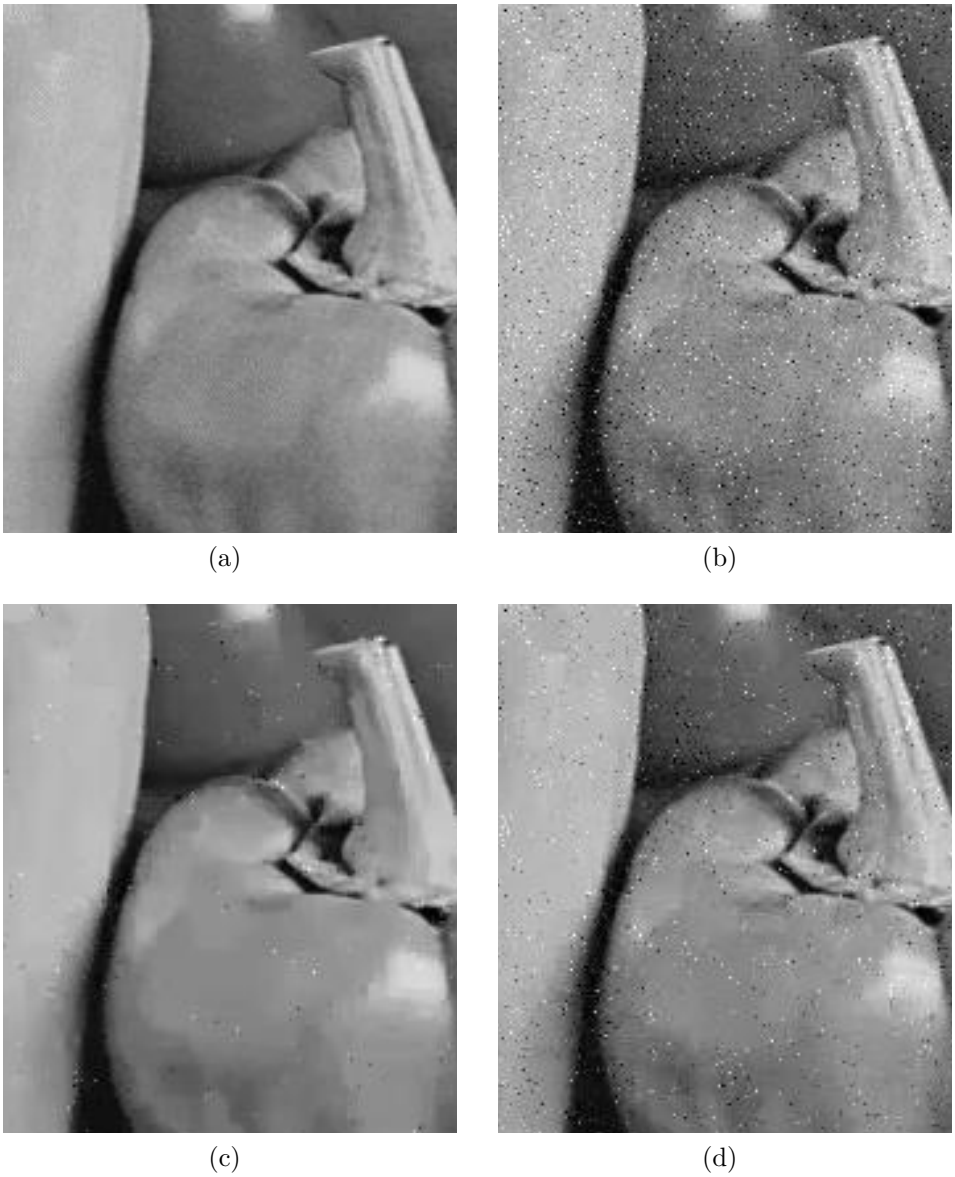
(d) Lena (e) Living room (f) Mandrill

(g) Peppers (h) Pirate

Figure 2.10: Sample images used in our experiments (a) House, (b) Jet plane, (c) Lake, (d) Lena, (e) Living room, (f) Mandrill, (g) Peppers, (h) Pirate.

Table 2.4: Comparison of the SNR results for denoising algorithms for ϵ-contaminated Gaussian noise for "Note" image

ϵ	σ_1	σ_2	$\text{SNR}_{\text{Input}}$	**PES-TV & BM3D**	Chambolle	BM3D
0.1	5	30	14.64	29.67	22.26	24.43
0.1	5	40	12.55	27.84	20.32	20.75
0.1	5	50	10.75	25.84	18.63	17.59
0.1	5	60	9.29	24.12	17.37	15.09
0.1	5	70	7.98	22.52	16.24	13.14
0.1	5	80	6.89	21.03	14.97	11.60
0.1	10	30	12.56	25.98	21.71	25.73
0.1	10	40	11.13	24.74	19.97	23.83
0.1	10	50	9.85	23.24	18.46	21.56
0.1	10	60	8.58	22.07	17.10	19.11
0.1	10	70	7.52	20.49	16.03	16.71
0.1	10	80	6.46	18.84	15.12	14.87
0.05	5	30	16.75	28.60	23.78	26.93
0.05	5	40	14.98	26.04	21.54	23.10
0.05	5	50	13.41	23.91	19.91	19.98
0.05	5	60	12.10	21.63	18.63	17.60
0.05	5	70	10.80	19.50	17.50	15.87
0.05	5	80	9.76	17.23	16.38	14.38
0.05	10	30	13.68	26.90	22.62	26.70
0.05	10	40	12.66	25.68	21.12	25.46
0.05	10	50	11.71	24.72	19.60	23.73
0.05	10	60	10.72	23.62	18.30	21.43
0.05	10	70	9.82	21.77	17.22	19.33
0.05	10	80	8.92	20.29	16.45	17.25



(a) (b)

(c) (d)

Figure 2.11: (a) A portion of original "Peppers" image, (b) image corrupted by ϵ-contaminated noise with $\epsilon = 0.1$, $\sigma_1 = 5$, and $\sigma_2 = 50$, (c) denoised image, using PES-TV algorithm; PSNR = 32.02 dB and, (d) denoised image, using BM3D; PSNR = 27.62 dB. Standard BM3D algorithm fails to clear impulsive noise.

(a)

(b)

(c)

(d)

(e)

Figure 2.12: (a) A portion of original "Lena" image, (b) image corrupted by salt & pepper noise with density 0.05, and additive white Gaussian noise with standard deviation $\sigma = 20$, (c) denoised image, using PES-TV algorithm; PSNR = 32.57 dB, (d) denoised image, using BM3D; PSNR = 28.95 dB, and (e) denoised image, using BM3D-Median; PSNR = 30.10 dB.

Table 2.5: PSNR Results for denoising using **PES-TV** algorithms under ϵ-contaminated noise with $\epsilon = 0.1$, $\sigma_1 = 5$, with different σ_2's.

Images	$\sigma_2 = 30$	$\sigma_2 = 40$	$\sigma_2 = 50$	$\sigma_2 = 60$	$\sigma_2 = 70$	$\sigma_2 = 80$
House	36.87	34.39	31.87	29.74	28.00	26.53
Lena	34.55	32.85	31.40	29.97	28.55	27.27
Mandrill	28.31	27.86	27.36	26.76	25.33	24.59
Living room	31.61	30.94	29.57	28.41	27.46	26.38
Lake	32.03	31.29	29.71	28.64	27.57	26.58
Jet plane	34.56	32.75	31.20	29.85	28.32	27.05
Peppers	34.64	33.39	32.02	30.56	29.22	27.87
Pirate	31.46	30.80	29.60	28.50	27.49	26.53
Cameraman	35.29	33.45	31.45	29.74	28.14	26.65
Flower	31.17	31.03	29.85	28.78	27.69	26.88
Kodak(ave.)	32.85	31.19	29.88	28.65	27.51	26.48
Average	33.08	31.53	30.14	28.86	27.64	27.30

Table 2.6: PSNR Results for denoising using BM3D algorithms under ϵ-contaminated noise with $\epsilon = 0.1$, $\sigma_1 = 5$, with different σ_2's.

Images	$\sigma_2 = 30$	$\sigma_2 = 40$	$\sigma_2 = 50$	$\sigma_2 = 60$	$\sigma_2 = 70$	$\sigma_2 = 80$
House	34.65	30.40	27.59	25.34	23.69	22.40
Lena	33.53	30.13	27.28	25.13	23.55	22.29
Mandrill	31.48	28.88	26.66	24.89	23.36	22.27
Living room	33.06	30.14	27.64	25.56	23.90	22.49
Lake	33.70	30.36	27.63	25.42	23.75	22.46
Jet plane	33.50	30.28	27.67	25.47	24.02	22.68
Peppers	33.66	30.50	27.62	25.48	23.86	22.46
Pirate	32.58	29.74	27.67	25.20	23.69	22.45
Cameraman	33.99	30.32	27.39	25.29	23.69	22.40
Flower	32.72	30.27	27.91	25.76	24.07	22.76
Kodak(ave.)	33.11	30.53	28.10	26.05	24.37	23.04
Average	33.28	30.50	28.00	25.92	24.26	22.93

Table 2.7: Denoising PSNR results for various algorithms for images corrupted by "salt & pepper" noise with density $d = 0.02$ plus Gaussian noise with variance σ.

	$\sigma = 20$			$\sigma = 30$			$\sigma = 40$			$\sigma = 50$		
Image	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$
Cameraman	**33.33**	27.88	31.62	**31.23**	29.40	29.15	29.75	**30.18**	27.18	28.93	**29.33**	25.63
Note	**35.39**	23.31	29.76	**32.73**	26.18	27.98	**30.72**	28.46	26.05	**28.44**	27.52	24.83
House	**35.64**	29.54	33.14	**33.74**	32.24	30.19	31.93	**33.02**	27.87	30.73	**32.07**	26.08
Flower	**30.20**	27.14	28.40	**28.33**	27.08	26.86	**27.03**	26.44	25.34	**26.12**	25.79	24.12
Lena	**32.67**	30.14	31.08	**30.66**	30.25	28.75	**29.65**	29.28	26.84	28.38	**28.78**	25.24
Jetplane	**32.34**	27.48	30.55	**30.28**	28.76	28.30	**28.84**	28.65	26.49	**27.93**	27.76	24.98
Lake	**30.31**	26.97	28.87	**28.40**	27.20	27.16	**27.08**	26.65	25.53	**26.11**	25.80	24.25
Peppers	**32.91**	29.83	31.14	**31.26**	30.24	28.87	**29.97**	29.68	26.95	**29.19**	28.90	25.37
Livingroom	**30.34**	28.49	28.82	**28.32**	27.81	27.02	**26.91**	26.87	25.48	**26.09**	25.96	24.20
mandrill	**28.63**	26.63	27.51	**26.49**	25.63	25.87	**25.03**	24.72	24.51	**24.11**	23.96	23.36
pirate	**30.32**	27.91	29.11	**28.54**	27.75	27.32	**27.32**	27.14	25.73	**26.61**	26.47	24.43

Table 2.8: Denoising PSNR results for various algorithms for images corrupted by "salt & pepper" noise with density $d = 0.05$ plus Gaussian noise with variance σ.

	$\sigma = 20$			$\sigma = 30$			$\sigma = 40$			$\sigma = 50$		
Image	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$	PES-TV	BM3D	BM3D$_M$
Cameraman	**33.21**	26.71	30.18	**31.19**	29.55	28.26	**29.73**	29.18	26.60	**28.73**	28.68	25.07
Note	**34.67**	27.10	27.24	**32.36**	27.02	25.85	**30.19**	26.49	24.72	**28.11**	25.78	23.62
House	**35.66**	28.91	31.91	**33.66**	32.67	29.35	**31.8**	32.05	27.35	**30.65**	31.27	25.71
Flower	**30.02**	25.80	27.58	**28.16**	26.26	26.12	**26.95**	25.76	24.86	**25.92**	25.28	23.75
Lena	**32.57**	28.95	30.10	**30.59**	29.55	28.09	**29.19**	28.84	26.37	**28.27**	28.17	24.87
Jetplane	**32.21**	26.40	29.2	**30.14**	28.36	27.36	**28.74**	27.70	25.83	**27.69**	27.10	24.50
Lake	**30.25**	25.81	27.71	**28.35**	26.44	26.27	**27.02**	25.79	24.92	**26.02**	25.19	23.77
Peppers	**32.76**	28.96	30.05	**31.16**	29.67	28.11	**29.83**	29.08	26.37	**28.91**	28.41	24.97
Livingroom	**30.25**	27.01	27.86	**28.19**	26.81	26.36	**26.87**	26.12	25.00	**25.89**	25.53	23.83
Mandrill	**28.50**	25.18	26.50	**26.39**	24.60	25.26	**24.92**	24.00	24.11	**23.98**	23.46	23.02
Pirate	**30.22**	26.81	28.19	**28.44**	26.99	26.66	**27.27**	26.42	25.32	**26.46**	25.95	24.05

An illustrative comparison of PES-TV vs. BM3D and BM3D$_M$ is presented in Figure 2.11 for "peppers" image. In the example images the success of the PES-TV algorithm can be easily observed. For example in Figure 2.11, the white and black dots of ϵ-contaminated noise still remains in the image denoised using BM3D algorithm. On the other hand, this issue is solved by the PES-TV method.

In [56], the proximity operator based denoising results for the Cameraman and Lena images are reported for various regularization parameter λ values for Gaussian noise with $\sigma = 15$ and 25 standard deviation levels. Best PSNR values for Lena image for $\sigma = 15$ ($\sigma = 25$) is 32.33 dB (30.13 dB), when the regularization parameter $\lambda = 0.09$ ($\lambda = 0.05$). We obtain PSNR values equal to 32.43 dB and 30.12 dB, respectively, without any regularization parameter adjustment. For Cameraman our results are much better with PSNR = 33.10 dB and 30.60 dB compared to 30.39 dB and 27.77 dB with $\lambda = 0.1$ and $\lambda = 0.07$ for $\sigma = 15$ and 25, respectively.

The first step of the BM3D approach relies on hard-thresholding, which cannot remove isolated large amplitude impulsive noise components. On the other hand, the PES-TV approach successfully reduces the impulsive noise and produces better estimates for the Wiener filtering based second stage of the BM3D denoising method. It is experimentally observed that the proposed scheme on images corrupted by impulsive noise results in much better denoising performance compared to both Chambolle's method and standard BM3D denoising.

Chapter 3

Denoising Using Projection onto Epigraph Set of ℓ_1-Norm Functions (PES-ℓ_1)

As mentioned before, Projection onto Epigraph Set of Convex Cost function (PESC), can be also used for denoising 1D signals. The ℓ_1-norm cost function is used for denoising 1D signals. Dobov *et al.*'s template based approach connot be used in most signals because there are no repetitive windows in most practical 1D signals. In the following sections, the Projection onto Epigraph Set of ℓ_1-norm function (PES-ℓ_1) is introduced.

3.1 The (PES-ℓ_1) Algorithm

In standard wavelet denoising, a signal corrupted by additive noise is wavelet transformed and resulting wavelet signals are soft- or hard-thresholded. After this step the denoised signal is reconstructed from the thresholded wavelet signals [2, 51]. Thresholding wavelet coefficients intuitively makes sense because wavelet signals obtained from an orthogonal or biorthogonal wavelet filter-bank

exhibit large amplitude coefficients only around edges or change instances of the original signal. The assumption is that other wavelet coefficients with small amplitude should be due to noise. A wide range of wavelet denoising methods which take advantage of the sparse nature of practical signals in wavelet domain are developed based on the Donoho and Johnstone's original denoising idea, see e.g. [1–3, 13, 47, 51, 66].

3.1.1 Problem Statement

Orthogonal projection of a vector onto a hyperplane is the key mathematical operation used in this thesis. Let $\mathbf{w}_o$ be a vector in $\mathbb{R}^K$. The orthogonal projection $\mathbf{w}_{\mathrm{p}o}$ of $\mathbf{w}_o$ onto the hyperplane $h = \mathbf{a}^T\mathbf{w}_o = \sum_{n=1}^{K} \mathbf{a}[n]\mathbf{w}_o[n]$ is given by

$$\mathbf{w}_{\mathrm{p}o}[n] = \mathbf{w}_o[n] + \frac{h - \sum_{n=1}^{K} \mathbf{a}[n]\mathbf{w}_o[n]}{\|\mathbf{a}\|_2^2}\mathbf{a}[n] \quad n = 1, 2, \ldots, K, \tag{3.1}$$

where $\mathbf{w}_o[n], \mathbf{w}_{\mathrm{p}o}[n]$, and $\mathbf{a}[n]$ are the n-th entries of the vectors $\mathbf{w}_o, \mathbf{w}_{\mathrm{p}o}$, and $\mathbf{a}$, respectively, and $\|\mathbf{a}\|_2$ is the Euclidean length (norm) of the vector $\mathbf{a}$. Orthogonal projection onto a hyperplane is also the key step of the well-known normalized LMS adaptive filtering algorithm and many online learning algorithms [67].

Consider the following basic denoising framework. Let $v[n]$ be a discrete-time signal and $x[n]$ be a noisy version of $v[n]$:

$$x[n] = v[n] + \xi[n], \quad n = 1, 2, \ldots, N, \tag{3.2}$$

where $\xi[n]$ is the additive, i.i.d, zero-mean, white Gaussian noise with variance σ^2. An L-level discrete wavelet transform of $x[n]$ is computed and the lowband signal $\mathbf{x}_L$ and wavelet signals $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_L$ are obtained. After this step, wavelet signals are soft-thresholded as shown in Figure 3.1. The soft-threshold, θ, can be selected in many ways [1, 2, 4, 47] using statistical methods. One possible choice is

$$\theta = \gamma.\sigma.\sqrt{2\log(N)/N}, \tag{3.3}$$

where γ is a constant [2]. In Eq. (3.3) the noise variance σ^2 has to be known or properly estimated from the observations, $x[n]$.

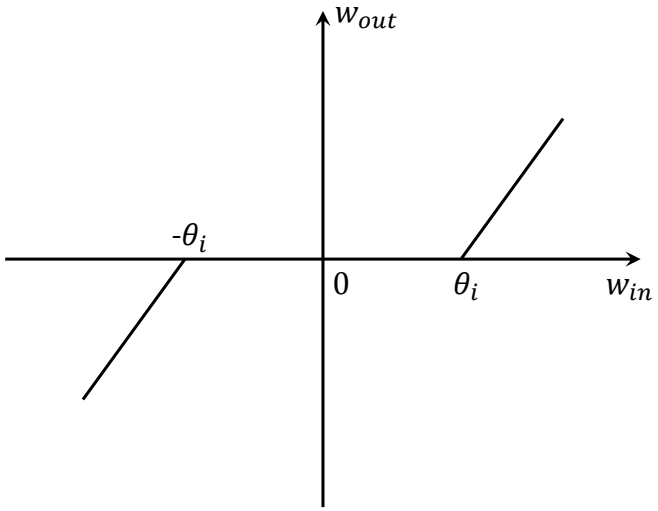


Figure 3.1: Soft-thresholding operation: $w_{out,n} = \text{sign}(w_{in,n})\{\max(|w_{in,n} - \theta_i, 0|)\}$

It is possible to define a soft-threshold θ_i for each wavelet signal $\mathbf{w}_i$. Here, a method of estimating soft-threshold values θ_i using a deterministic approach based on linear algebra and orthogonal projections is presented.

3.1.2 Wavelet Signals Denoising with Projections onto ℓ_1-balls

Let us first study the projection of wavelet signals $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_L$ onto ℓ_1-balls, which we will use to describe the projection onto the epigraph set of ℓ_1-norm cost function. We will use the term vector and signal in an interchangeable manner from now on. An ℓ_1-ball $\mathcal{C}_i$, with size d_i is defined as follows:

$$\mathcal{C}_i = \{\mathbf{w} \in \mathbb{R}^N \ : \ \sum_n |\mathbf{w}[n]| \leq d_i\}, \tag{3.4}$$

where $\mathbf{w}[n]$ is the n-th component of the vector $\mathbf{w}$, and d_i is the size of the ℓ_1-ball. In other words, an ℓ_1-ball is the set of vectors characterized by the fact that the sum of the magnitude of its components is lower than some specified value. Geometrically, such an ℓ_1-ball is a diamond shaped region bounded by a collection of hyperplanes as depicted in Figure 3.2. The orthogonal projection of

a wavelet vector $\mathbf{w}_i$ onto an ℓ_1-ball is mathematically defined as follows:

$$\begin{aligned} \mathbf{w}_{\mathrm{p}i} &= \arg\min \|\mathbf{w}_i - \mathbf{w}\|_2^2 \\ &\text{such that } \|\mathbf{w}_i\| = \sum_n |\mathbf{w}_i[n]| \leq d_i, \end{aligned} \tag{3.5}$$

where $\mathbf{w}_i$ is the i-th wavelet signal, $\|.\|_2$ is the Euclidean norm, and $\|.\|_1$ is the ℓ_1 norm. The orthogonal projection operation onto an ℓ_1-ball is graphically shown in Figure 3.2. When $\|\mathbf{w}_i\|_1 \leq d_i$ is satisfied, the wavelet signal is inside the ball, the projection has no effect and $\mathbf{w}_{\mathrm{p}l} = \mathbf{w}_l$. In general, it can be shown that the orthogonal projection operation soft-thresholds each wavelet coefficient $\mathbf{w}_i[n]$ as follows:

$$\mathbf{w}_{\mathrm{p}i}[n] = \mathrm{sign}(\mathbf{w}_i[n]) \max\big\{(|\mathbf{w}_i[n]| - \theta_i), 0\big\}, \tag{3.6}$$

where $\mathrm{sign}(\mathbf{w}_i[n])$ is the sign of $\mathbf{w}_i[n]$, and θ_i is a soft-thresholding constant whose value is determined according to the size of the ℓ_1-ball, d_i [68]. Algorithm 1 is an example of a method to solve the minimization problem (3.5) and thereby provide the constant θ_i for a given d_i value [68].

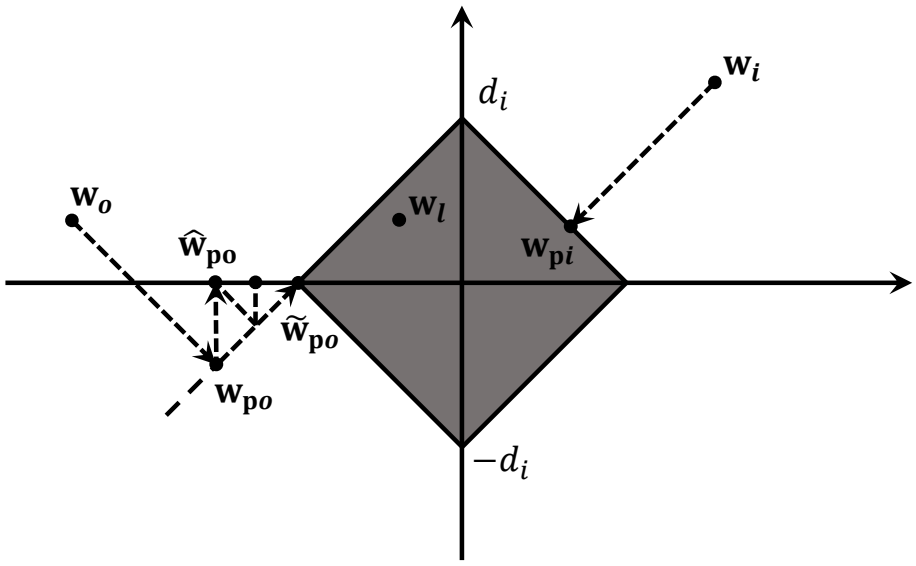


Figure 3.2: Graphical illustration of projection onto an ℓ_1-ball with size d_i: Vectors $\mathbf{w}_{\mathrm{p}i}$ and $\widetilde{\mathbf{w}}_{\mathrm{p}o}$ are orthogonal projections of $\mathbf{w}_i$ and $\mathbf{w}_o$ onto an ℓ_1-ball with size d_i, respectively. The vector $\mathbf{w}_l$ is inside the ball, $\|\mathbf{w}_l\|_1 \leq d_i$, and projection has no effect: $\mathbf{w}_{\mathrm{p}l} = \mathbf{w}_l$

Projection of a wavelet signal onto an ℓ_1-ball reduces amplitudes of wavelet

coefficients of the input vector and eliminates the small valued wavelet coefficients, which are less than the threshold θ_i. As a result, wavelet coefficients which are probably due to noise, are removed by the projection operation. Projection operation onto an ℓ_1-ball retains the edges and sharp variation regions of the original signal because wavelet signals have large amplitude valued coefficients corresponding to edges [51] and they are not significantly affected by soft-thresholding. In standard wavelet denoising methods, the low-band signal $\mathbf{x}_L$ is not processed because $\mathbf{x}_L$ is a low resolution version of the original signal containing large amplitude coefficients almost for all n for most practical signals and images.

Algorithm 1 Order $(Klog(K))$ algorithm implementing projection onto the ℓ_1-ball with size d_i.

1: **Inputs:**

A vector $\mathbf{w}_i = [w_i[1], \ldots, w_i[K]]$ and a scalar $d_i > 0$

2: **Initialize:**

Sort $|\mathbf{w}_i[n]|$ for $n = 1, \ldots, K$ and obtain the rank ordered sequence $\mu_1 \geq \mu_2 \geq, \ldots, \geq \mu_K$. The soft-threshold value, θ_i, is given by

$$\theta_i = \frac{1}{\rho}\Big(\sum_{n=1}^{\rho} \mu_n - d_i\Big) \quad \text{such that} \quad \rho = \max\Big\{j \in \{1, 2, \ldots, K\} : \mu_j - \frac{1}{j}\Big(\sum_{r=1}^{j} \mu_r - d_i\Big) > 0\Big\}$$

3: **Output:**

$$\mathbf{w}_{\mathrm{p}i}[n] = \mathrm{sign}(\mathbf{w}_i[n]) \max\big\{|\mathbf{w}_i[n]| - \theta_i, 0\big\},\ n = 1, 2, \ldots, K$$

Let us consider Figure 3.2 once again and consider the vector $\mathbf{w}_o$. The vector $\mathbf{w}_{\mathrm{p}o}$ is the orthogonal projection of $\mathbf{w}_o$ onto one of the boundary hyperplanes of the ℓ_1-ball:

$$\sum_{n=1}^{K} \mathrm{sign}(\mathbf{w}_o[n])\mathbf{w}[n] - d_i = 0. \tag{3.7}$$

The vector $\widehat{\mathbf{w}}_{\mathrm{p}o}$ in Figure 3.2 is the projected version of $\mathbf{w}_{\mathrm{p}o}$ back to the quadrant of $\mathbf{w}_o$. It is also possible to use the vector $\widehat{\mathbf{w}}_{\mathrm{p}o}$ for denoising purposes. The vertical entry of $\widehat{\mathbf{w}}_{\mathrm{p}o}$ is the same as $\widetilde{\mathbf{w}}_{\mathrm{p}o}$, which is zero, but the horizontal entry of $\widehat{\mathbf{w}}_{\mathrm{p}o}$ is larger in amplitude than the first entry of $\widetilde{\mathbf{w}}_{\mathrm{p}o}$, which is the projection vector in ℓ_1-ball. In general, zero valued entries of the $\widetilde{\mathbf{w}}_{\mathrm{p}o}$ and $\widehat{\mathbf{w}}_{\mathrm{p}o}$ are the same. Therefore, significant coefficients of wavelet signals can be determined after two orthogonal projections, and the rest of coefficients are zeroed out.

In standard wavelet denoising, the noise variance now has to be estimated to determine the soft-threshold value. Equivalently, the size of the ℓ_1-ball, d_i, has to be estimated. Moreover, both standard wavelet denoising and the ℓ_1-ball based denoising need to determine the number of wavelet decomposition levels.

The next step is the estimation of the size of the ℓ_1-ball, d_i. We estimate the size of the ℓ_1-ball, d_i, by projecting $\mathbf{w}_i$ onto the epigraph set of ℓ_1-norm cost function which is an upside down pyramid in $\mathbb{R}^{N+1}$ as shown in Figure 3.3. An upside down pyramid is constructed by a family of ℓ_1-balls or diamond shaped regions with different sizes ranging from 0 to $d_{\max,i} = \sum_n |\mathbf{w}_i[n]|$, whose value is the ℓ_1-norm of $\mathbf{w}_i$. When we orthogonally project $\mathbf{w}_i$ onto the upside down pyramid, we not only estimate the size of the ℓ_1-ball, but also soft-threshold the wavelet signal $\mathbf{w}_i$ as discussed in the Section 3.1.3.

3.1.3 Estimation of Denoising Thresholds

The epigraph set of ℓ_1-norm cost function is an upside down pyramid shaped region as shown in Figure 3.3. Each horizontal slice of the upside down pyramid is an ℓ_1-ball. The smallest value of the ℓ_1-ball is 0, which is at the bottom of the pyramid. The largest value of the ℓ_1-ball in the upside down pyramid is $d_{\max,i} = \|\mathbf{w}_i\|_1$, which is determined by the boundary of the ℓ_1-ball touching the wavelet signal $\mathbf{w}_i$, i.e., the wavelet signal $\mathbf{w}_i$ is on one of the boundary hyperplanes of the ℓ_1-ball.

Orthogonal projection of $\mathbf{w}_i$ onto an ℓ_1-ball with $d = 0$ produces an all-zero result. Projection of $\mathbf{w}_i$ onto an ℓ_1-ball with size $d_{\max,i}$, does not change $\mathbf{w}_i$ because $\mathbf{w}_i$ is on the boundary of the ℓ_1-ball. Therefore, for meaningful results, the size of the ℓ_1-ball, $d_i = z_{\mathrm{p}i}$, must satisfy the inequality $0 < z_{\mathrm{p}i} < d_{\max,i}$, for denoising. This condition can be expressed as follows:

$$\|\mathbf{w}_i\|_1 = \sum_{k=1}^{K} |\mathbf{w}_i[k]| \leq z_{\mathrm{p}i}, \tag{3.8}$$

where K is the length of the wavelet vector $\mathbf{w} = [\mathbf{w}[1], \mathbf{w}[2], \ldots, \mathbf{w}[K]]^T \in \mathbb{R}^K$.

The condition (3.8) corresponds to the epigraph set $\mathcal{C}$ of the ℓ_1-norm cost function in $\mathbb{R}^{K+1}$, which is graphically illustrated in Figure 3.3 for $\mathbf{w}_i \in \mathbb{R}^2$ [13, 66]. The epigraph set $\mathcal{C}$ is defined in $\mathbb{R}^{N+1}$ as follows:

$$\mathcal{C} = \{\underline{\mathbf{w}}_i = [\mathbf{w}_i^T, z_{\mathrm{p}i}]^T \in \mathbb{R}^{K+1} \ : \ \|\mathbf{w}_i\|_1 \leq z_{\mathrm{p}i}, \ z_{\mathrm{p}i} \leq d_{\max,i}\}, \tag{3.9}$$

which represents a family of ℓ_1-balls for $0 < z_{\mathrm{p}i} \leq d_{\max,i}$ in $\mathbb{R}^{K+1}$. In (3.9) there are $K+1$ variables: $\mathbf{w}_i[1], \ldots, \mathbf{w}_i[K]$, and $z_{\mathrm{p}i}$. Since the space is now $K+1$ dimensional, we increase the size of wavelet signals by one:

$$\underline{\mathbf{w}}_i = [\mathbf{w}_i^T, 0]^T = [w_i[1], w_i[2], \ldots, w_i[K], 0]^T, \tag{3.10}$$

where $\underline{\mathbf{w}}_i \in \mathbb{R}^{K+1}$. The signal $\underline{\mathbf{w}}_i$ is the $K+1$ dimensional version of vector $\mathbf{w}_i \in \mathbb{R}^K$. From now on, we underline vectors in $\mathbb{R}^{K+1}$ to distinguish them from K dimensional vectors.

The extended wavelet vector $\underline{\mathbf{w}}_i$ can be projected onto the epigraph set $\mathcal{C}$ to determine the vector $\underline{\mathbf{w}}_{\mathrm{p}i} = [\mathbf{w}_{\mathrm{p}i}[1], \ldots, \mathbf{w}_{\mathrm{p}i}[K], z_{\mathrm{p}i}]^T$ as graphically illustrated in Figure 3.3. This projection is unique and is the closest vector on the epigraph set to $\underline{\mathbf{w}}_i = [\mathbf{w}_i^T, 0]^T$. The baseline mathematical operation is an orthogonal projection onto a hyperplane which is the face (boundary) of the epigraph set $\mathcal{C}$ in the quadrant of the $\underline{\mathbf{w}}_i$. The orthogonal projection $\underline{\mathbf{w}}_{\mathrm{p}i}$ of $\underline{\mathbf{w}}_i$ is a denoised version of $\underline{\mathbf{w}}_i$ because it is equivalent to the orthogonal projection of $\mathbf{w}_i$ onto the ℓ_1-ball with size $z_{\mathrm{p}i}$ in $\mathbb{R}^K$, as graphically illustrated in Figure 3.3.

Orthogonal projection onto the epigraph set $\mathcal{C}$ can be computed in two steps. In the first step, $[\mathbf{w}_i^T, 0]^T$ is projected onto the boundary hyperplane of the epigraph set which is defined as:

$$\sum_{n=1}^{K} \mathrm{sign}(\mathbf{w}_i[n]).\mathbf{w}_i[n] - z_{\mathrm{p}i} = 0, \tag{3.11}$$

where the coefficients of the above hyperplane are determined according to the signs of $\mathbf{w}_i[n]$. This hyperplane determines the boundary of the epigraph set $\mathcal{C}$ facing the vector $\underline{\mathbf{w}}_i$ as shown in Figure 3.3. The projection vector $\underline{\mathbf{w}}_{\mathrm{p}i}$ onto the hyperplane (3.11) in $\mathbb{R}^{K+1}$ is determined using Equation (3.1), which is:

$$\mathbf{w}_{\mathrm{p}i}[n] = \mathbf{w}_i[n] - \frac{\sum_{n=1}^{K} |\mathbf{w}_i[n]|}{K+1} \mathrm{sign}(\mathbf{w}_i[n]) \quad n = 1, 2, \ldots, K, \tag{3.12}$$

where $K+1 = \|[\text{sign}(\mathbf{w}_i[1]), \ldots, \text{sign}(\mathbf{w}_i[k]), -1]\|^2$ and the last component $z_{\text{p}i}$ of $\underline{\mathbf{w}}_{\text{p}i}$ is given by

$$z_{\text{p}i} = \frac{\sum_{n=1}^{K} \text{sign}(\mathbf{w}_i[n])\mathbf{w}_i[n]}{K+1} = \frac{\sum_{n=1}^{K} |\mathbf{w}_i[n]|}{K+1}. \tag{3.13}$$

As mentioned earlier above, this orthogonal projection operation also determines the size of the ℓ_1-ball, $d_i = z_{\text{p}i}$, which can be verified using geometry.

In general, the projection vector $\underline{\mathbf{w}}_{\text{p}i}$ may or may not be the projection of $\underline{\mathbf{w}}_i$ onto the epigraph set $\mathcal{C}$. In Figures 3.2 and 3.3, it is. The ℓ_1-ball in Figure 3.2 can be interpreted as the projection of 3-D ℓ_1-ball onto 2-D plane (view from the top). The issue comes from the fact that projecting onto the ℓ_1-ball has been simplified to projecting onto a single hyperplane, which may not yield the desired result in some specific geometrical configurations. For instance, in Figure 3.2, the vector $\mathbf{w}_{\text{p}o}$ is neither the orthogonal projection of $\mathbf{w}_o$ onto the ℓ_1-ball, nor to the epigraph set of the ℓ_1-ball, because $\mathbf{w}_{\text{p}o}$ is not on the ℓ_1-ball. Such cases can easily be spotted by checking the signs of the entries of the projection vectors. If the signs of the entries $\mathbf{w}_{\text{p}i}[n]$ of projection vector $\mathbf{w}_{\text{p}i}$ are the same as $\mathbf{w}_i[n]$ for all n then the $\mathbf{w}_{\text{p}i}$ is on the epigraph set $\mathcal{C}$, otherwise $\mathbf{w}_{\text{p}i}$ is not on the ℓ_1-ball. If $\mathbf{w}_{\text{p}i}$ is not on the ℓ_1-ball we can still project $\mathbf{w}_i$ onto the ℓ_1-ball using Algorithm 1 or Duchi et al's ℓ_1-ball projection algorithm [68] using the value of $d_i = z_{\text{p}i}$ determined in Equation (3.13).

In summary, we have the following two steps: (i) project $\underline{\mathbf{w}}_i = [\mathbf{w}_i^T, 0]^T$ onto the boundary hyperplane of the epigraph set $\mathcal{C}$ and determine d_i using Equation (3.13); (ii) if $\text{sign}(\mathbf{w}_i[n]) = \text{sign}(\mathbf{w}_{\text{p}i}[n])$ for all n, $\mathbf{w}_{\text{p}i}$ is the projection vector; otherwise, use $d_i = z_{\text{p}i}$ in Algorithm 1 to determine the final projection vector. Since there are $i = 1, 2, \ldots, L$ wavelet signals, each wavelet signal $\mathbf{w}_i$ should be projected onto possibly distinct ℓ_1-balls with sizes d_i. Notice that d_i is not the value of the soft-threshold, it is the size of the ℓ_1-ball. The value of the soft-threshold is determined using Algorithm 1.

In practice, we may further simplify step (ii) in denoising applications. Our goal is to zero out insignificant wavelet coefficients. Therefore, we compare signs of entries of $\mathbf{w}_{\text{p}o}$ and $\mathbf{w}_o$. We can zero out those entries whose signs change after

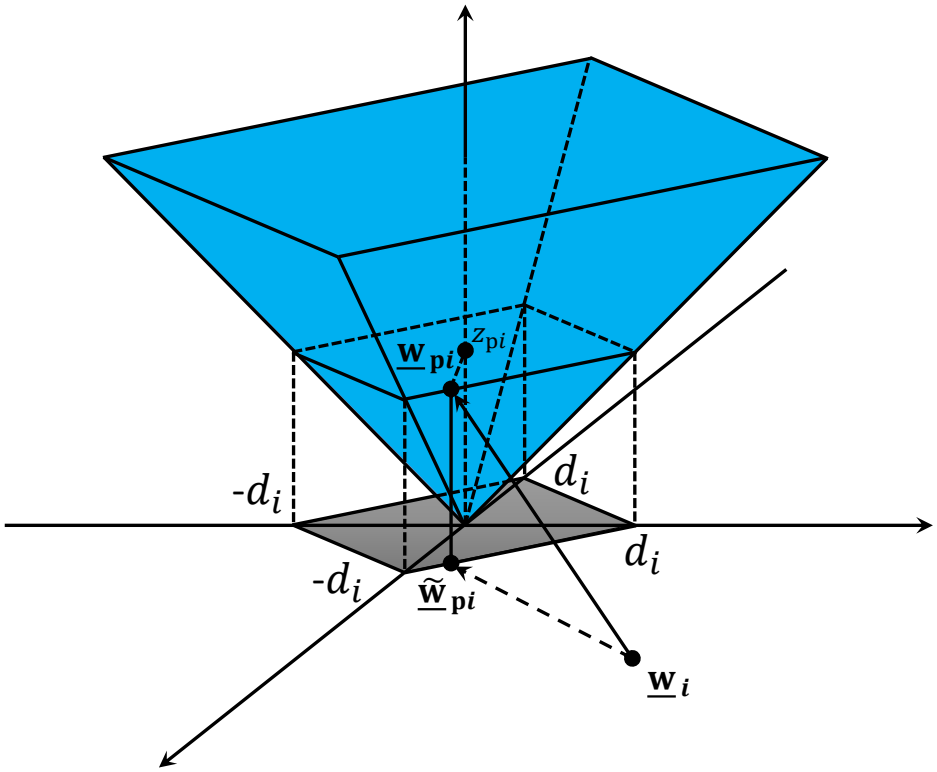


Figure 3.3: Projection of $\mathbf{w}_i[n]$ onto the epigraph set of ℓ_1-norm cost function: $\mathcal{C} = \{\mathbf{w} : \sum_{n=0}^{K-1} |\mathbf{w}[k]| \leq z_{\mathrm{p}i}\}$, gray shaded region

the orthogonal projection. Therefore, step (ii) becomes

$$\widehat{\mathbf{w}}_{\mathrm{p}o}[n] = \begin{cases} \mathbf{w}_{\mathrm{p}o}[n], & \text{if } \mathrm{sign}(\mathbf{w}_{\mathrm{p}o}[n]) = \mathrm{sign}(\mathbf{w}_o[n]) \\ 0, & \text{otherwise.} \end{cases} \tag{3.14}$$

This operation is also graphically illustrated in Figure 3.2. The vector $\mathbf{w}_o$ is projected onto the boundary hyperplane facing $\mathbf{w}_o$ to obtain $\mathbf{w}_{\mathrm{p}o}$, which then projected back to the quadrant of $\mathbf{w}_o$ to obtain the denoised version $\widehat{\mathbf{w}}_{\mathrm{p}o}$. This process can be iterated a couple of times to approach the orthogonal projection vector $\widetilde{\mathbf{w}}_{\mathrm{p}o}$ as shown in Figure 3.2.

Stronger denoising of the input vector is simply a matter of selecting a z_{p} value smaller than $z_{\mathrm{p}i}$ in Equation (3.13). A z_{p} value closer to zero leads to a higher threshold and forces more wavelet coefficients to be zero after the projection operation.

3.1.4 How to Determine the Number of Wavelet Decomposition Levels

There are many ways to estimate the number of wavelet decomposition levels [1]. It is also possible to use the Fourier transform of the noisy signal to approximately estimate the bandwidth of the signal. Once the bandwidth ω_0 of the original signal is approximately determined, it can be used to estimate the number of wavelet transform levels and the bandwidth of the low-band signal $\mathbf{x}_L$. In an L-level wavelet decomposition, the low-band signal $\mathbf{x}_L$ approximately comes from the $[0, \frac{\pi}{2^L}]$ frequency band of the signal $\mathbf{x}$. Therefore, $\frac{\pi}{2^L}$ must be comparable to ω_0 so that the actual signal components are not soft-thresholded. Only wavelet signals $\mathbf{w}_1, \ldots, \mathbf{w}_{L-1}, \mathbf{w}_L$, whose Fourier transforms approximately occupy the bands $[\frac{\pi}{2}, \pi], \ldots, [\frac{\pi}{2^{L-1}}, \frac{\pi}{2^{L-2}}], [\frac{\pi}{2^L}, \frac{\pi}{2^{L-1}}]$, respectively, should be soft-thresholded in denoising. For example, consider the `cusp` signal defined in MATLAB. It is possible to estimate an approximate frequency value ω_0 for this signal. The `cusp` signal is corrupted by additive zero-mean white Gaussian noise with $\sigma = 20\%$ of the maximum amplitude of the original signal as shown in Figure 3.8b. The magnitude of the Fourier transform of the `cusp` signal is shown in Figure 3.4. For this signal, an $L = 5$ level wavelet decomposition is suitable because the magnitude of the Fourier transform approaches the noise floor level at high frequencies after $\omega_0 \approx \frac{\pi}{46}$ as shown in Figure 3.4. Therefore, $L = 5$ ($\frac{\pi}{2^5} > \omega_0$) is selected as the number of wavelet decomposition levels.

It is also possible to use a pyramidal structure for signal decomposition instead of the wavelet transform. The noisy signal is low-pass filtered with cut-off frequency $\frac{\pi}{8}$ for `cusp` signal and the output $x_{lp}[n]$ is subtracted from the noisy signal $x[n]$ to obtain the high-pass signal $x_{hp}[n]$ as shown in Figure 3.5. The signal is projected onto the epigraph of ℓ_1-ball and $x_{hd}[n]$ is obtained. Projection onto the Epigraph Set of ℓ_1-ball (PES-ℓ_1), described in the previous section, removes the noise by soft-thresholding. The pyramidal signal decomposition approach is similar to undecimated wavelet transform. The denoised signal $x_{den}[n]$ is reconstructed by adding $x_{hd}[n]$ and $x_{lp}[n]$ as shown in Figure 3.5. It is possible to use

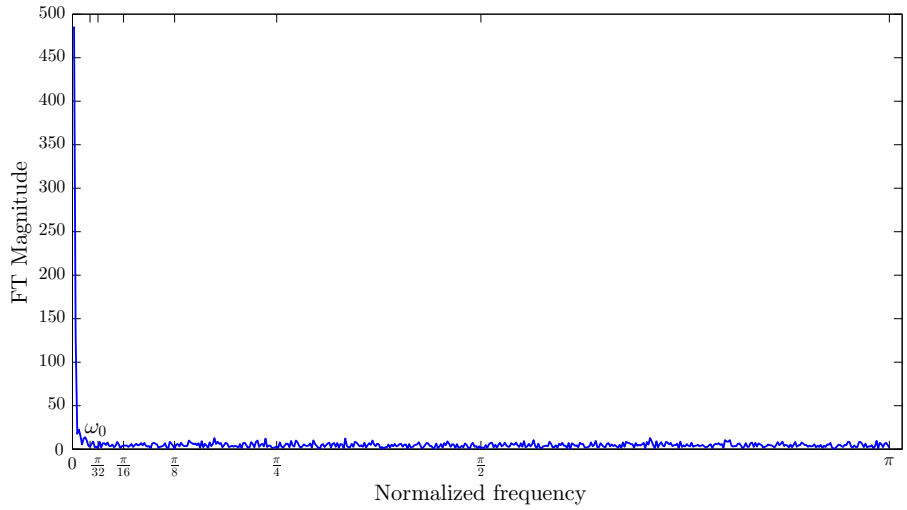


Figure 3.4: Discrete-time Fourier transform magnitude of `cusp` signal corrupted by noise. The wavelet decomposition level L is selected as 5 satisfying $\frac{\pi}{2^5} > \omega_0$, which is the approximate bandwidth of the signal.

different thresholds for different subbands as in wavelet transform, using a multistage pyramid as shown in Figure 3.5. In the first stage a low-pass filter with cut-off $\frac{\pi}{2}$ can be used and $x_{hp1}[n]$ is projected onto the epigraph set of ℓ_1-ball producing a threshold for the subband $[\frac{\pi}{2}, \pi]$. In the second stage, another low-pass filter with cut-off $\frac{\pi}{4}$ can be used and $x_{hp}[n]$ is projected onto the epigraph set producing a threshold for $[\frac{\pi}{4}, \frac{\pi}{2}]$, etc.

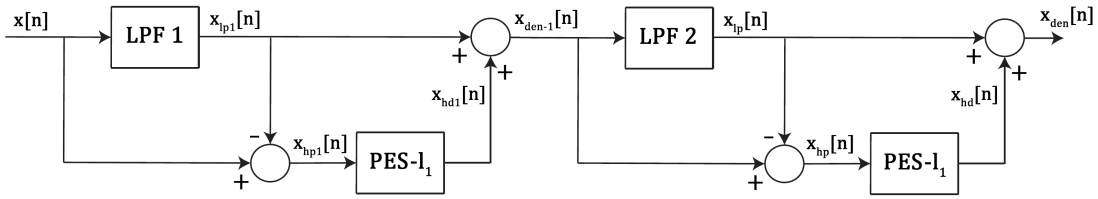


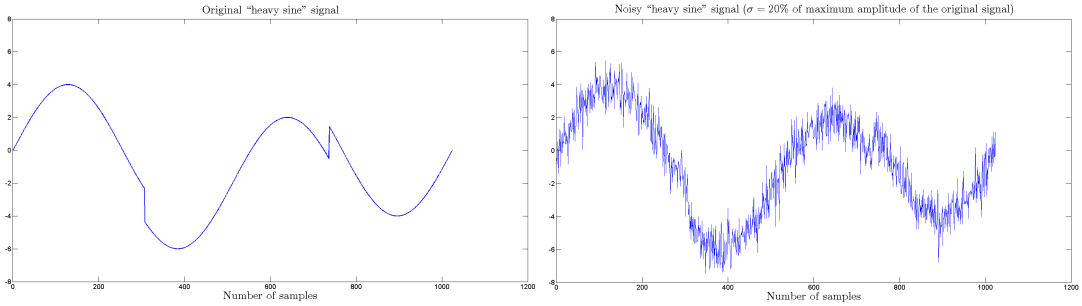
Figure 3.5: Pyramidal filtering based denoising. the high-pass filtered signal is Projected onto the Epigraph Set of ℓ_1 (PES-ℓ_1).

The simulation results for PES-ℓ_1 algorithm with both pyramidal structure and wavelet transform domain are compared to the well-known algorithms. This simulation results are presented in Section 3.2.

3.2 Simulation Results

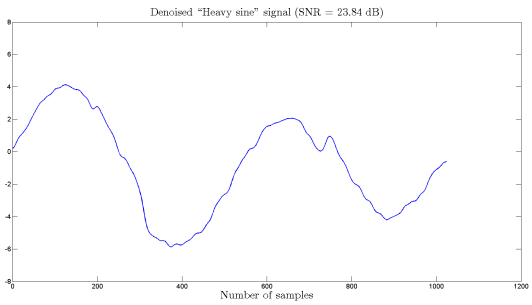
Epigraph set based threshold selection is compared with wavelet denoising methods used in MATLAB [1, 2, 47]. The `heavy sine` signal shown in Figure 3.6a is corrupted by a zero mean Gaussian noise with $\sigma = 20\%$ of the maximum amplitude of the original signal. The signal is restored using PES-ℓ_1 with pyramid structure, PES-ℓ_1 with wavelet, MATLAB's wavelet multivariate denoising algorithm [1], MATLAB's soft-thresholding denoising algorithm (for `minimaxi` and `rigrsure` thresholds), and wavelet thresholding denoising method. The denoised signals are shown in Figure 3.6c, 3.6d, 3.6e, 3.6f, 3.6g, and 3.6h with SNR values equal to 23.84, 23.79, 23.52, 23.71, 23.06 dB, and 21.38, respectively. The original, noisy and denoised signals for `piece-regular`, and `cusp` signals are also presented in Figures 3.7 and 3.8, respectively. In Figures 3.7b and 3.8b, the original signal is corrupted by a zero mean Gaussian noise with $\sigma = 10\%$ of the maximum amplitude of the original signal. On the average, the proposed PES-ℓ_1 with pyramid and PES-ℓ_1 with wavelet method produce better thresholds than the other soft-thresholding methods. In another example the `cusp` signal is corrupted by a zero mean Gaussian noise with $\sigma = 20\%$ of the maximum amplitude of the original signal as in Figure 3.10a. The denoising results for this case is presented in Figure 3.10b. MATLAB codes of the denoising algorithms and other simulation examples are available in the following web-page: http://signal.ee.bilkent.edu.tr/1DDenoisingSoftware.html.

Extensive simulation results for other test signals in MATLAB are presented in Tables 3.1, 3.2, and 3.3, for the cases with Gaussian noise with $\sigma = 10$, 20, and 30 % of maximum amplitude of original signal, respectively. These results are obtained by averaging the SNR values after repeating the simulations for 300 times. The SNR is calculated using the formula: $\text{SNR} = 20 \times log_{10}(\|\mathbf{w}_{orig}\|/\|\mathbf{w}_{orig} - \mathbf{w}_{rec}\|)$. In this lecture note, it is shown that soft-denoising threshold can be determined using basic linear algebra.

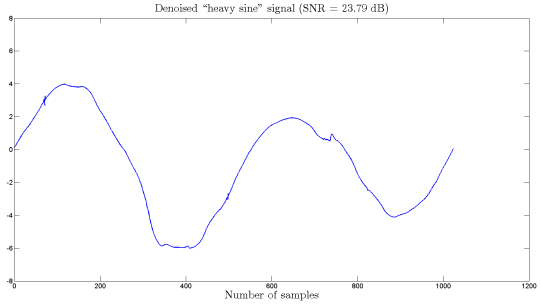


(a) Original signal

(b) Noisy signal

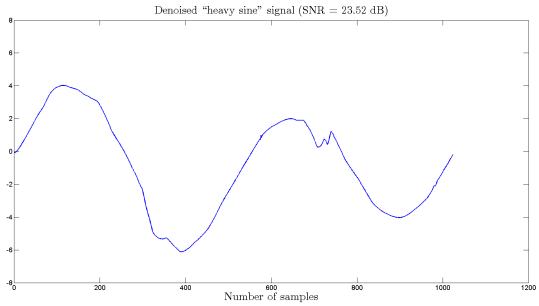


(c) PES-ℓ_1 with pyramid

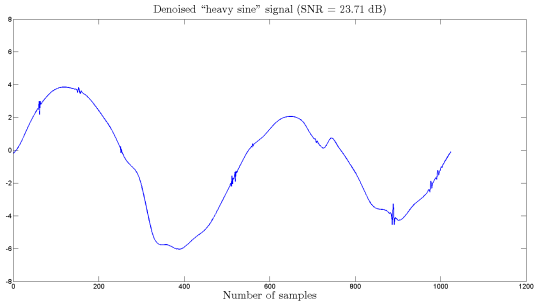


(d) PES-ℓ_1 with wavelet

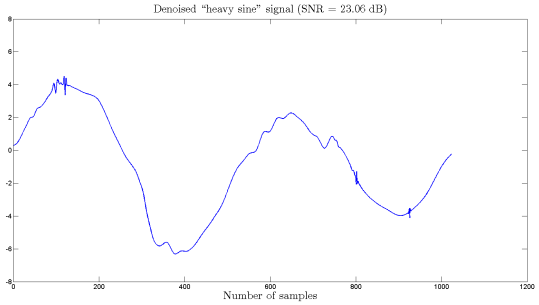
Figure 3.6: (a) Original `heavy sine` signal, (b) signal corrupted with Gaussian noise with $\sigma = 20\%$ of maximum amplitude of the original signal, and denoised signal using (c) PES-ℓ_1-ball with pyramid; SNR = 23.84 dB and, (d) PES-ℓ_1-ball with wavelet; SNR = 23.79 dB, *(cont.)*



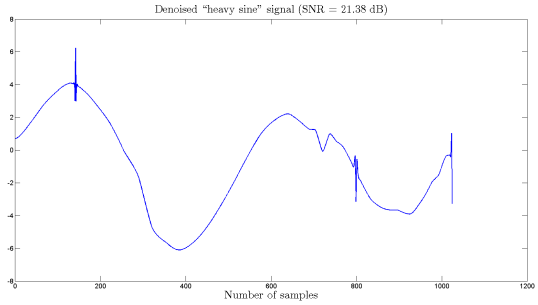
(e) Wavelet denoising in MATLAB [1]



(f) Wavelet denoising `minimaxi` algorithm [2]

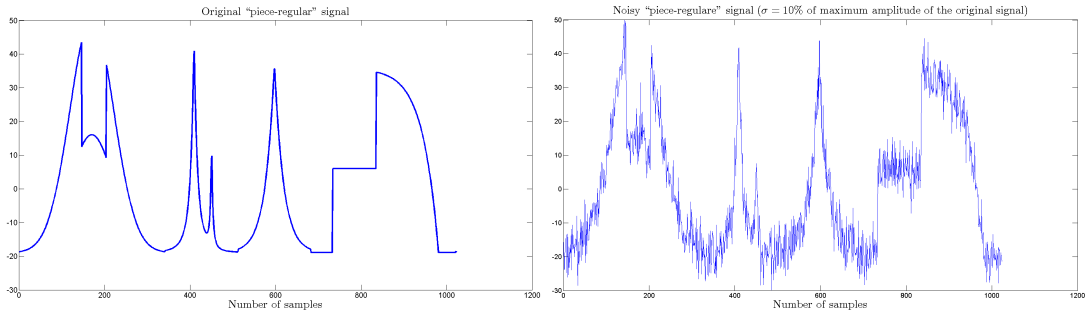


(g) Wavelet denoising `rigrsure` algorithm [3]



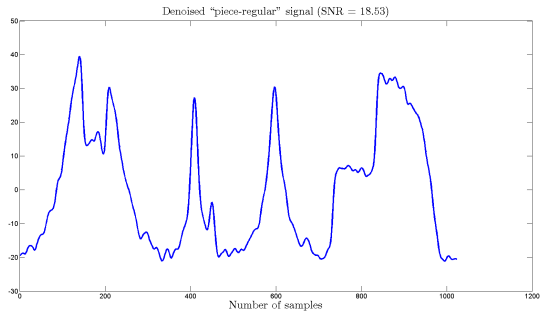
(h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]

Figure 3.6: (e) Wavelet denoising in Matlab; SNR = 23.52 dB [1], (f) Wavelet denoising `minimaxi` algorithm [2]; SNR = 23.71 dB, (g) Wavelet denoising `rigrsure` algorithm [3]; SNR = 23.06 dB, (h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]; SNR = 21.38 dB.

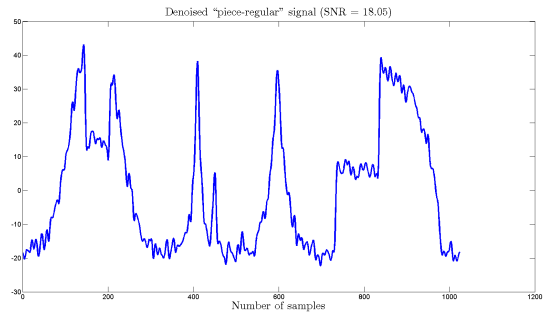


(a) Original signal

(b) Noisy signal

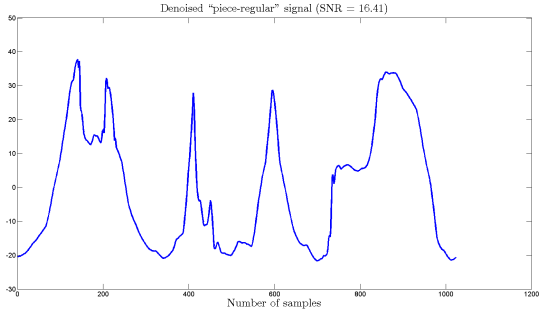


(c) PES-ℓ_1 with pyramid

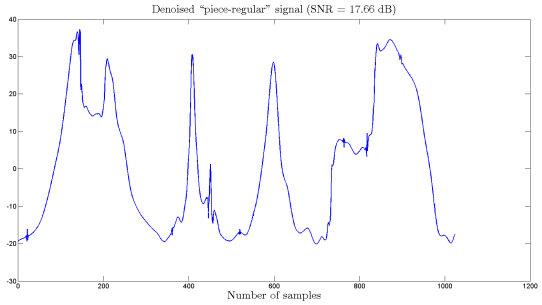


(d) PES-ℓ_1 with wavelet

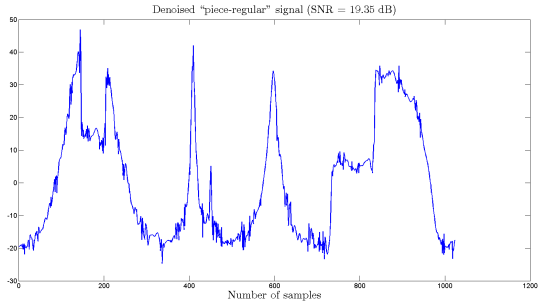
Figure 3.7: (a) Original `piece-regular` signal, (b) signal corrupted with Gaussian noise with $\sigma = 10\%$ of maximum amplitude of the original signal, and denoised signal using (c) PES-ℓ_1-ball with pyramid; SNR = 23.84 dB and, (d) PES-ℓ_1-ball with wavelet; SNR = 23.79 dB, *(cont.)*



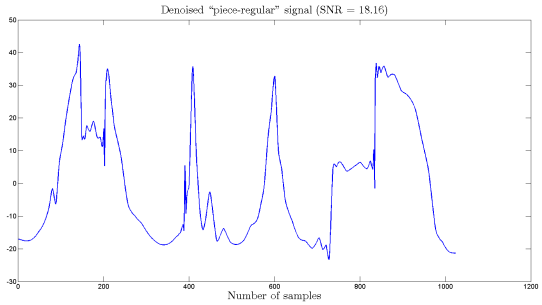
(e) Wavelet denoising in MATLAB [1]



(f) Wavelet denoising `minimaxi` algorithm [2]

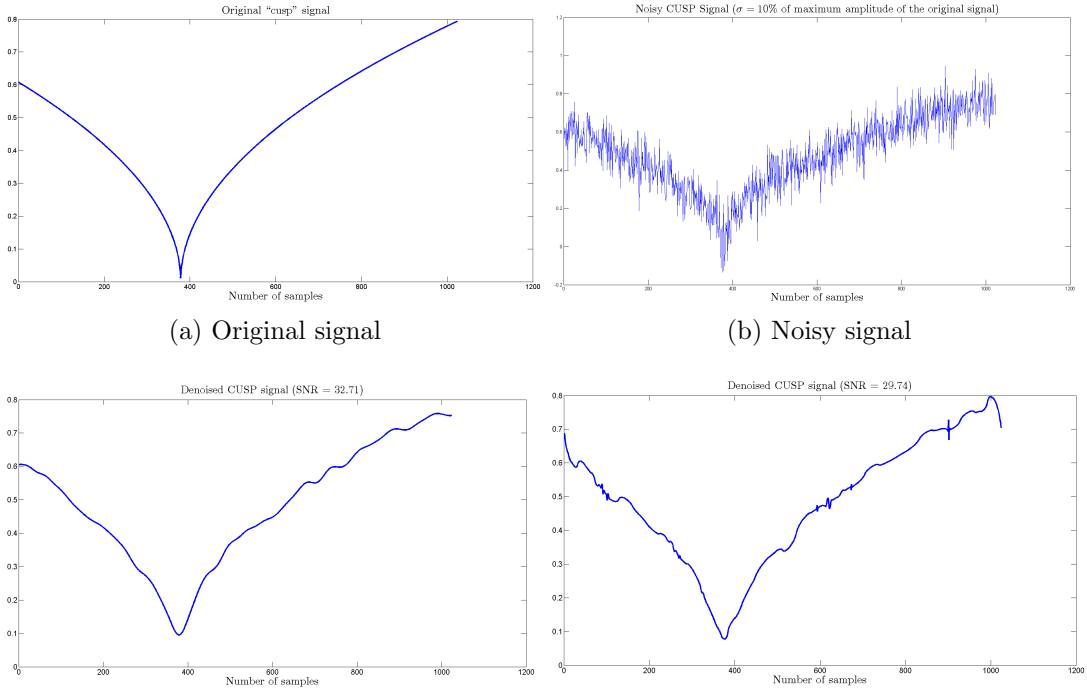


(g) Wavelet denoising `rigrsure` algorithm [3]



(h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]

Figure 3.7: (e) Wavelet denoising in Matlab; SNR = 23.52 dB [1], (f) Wavelet denoising `minimaxi` algorithm [2]; SNR = 23.71 dB, (g) Wavelet denoising `rigrsure` algorithm [3]; SNR = 23.06 dB, (h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]; SNR = 21.38 dB.



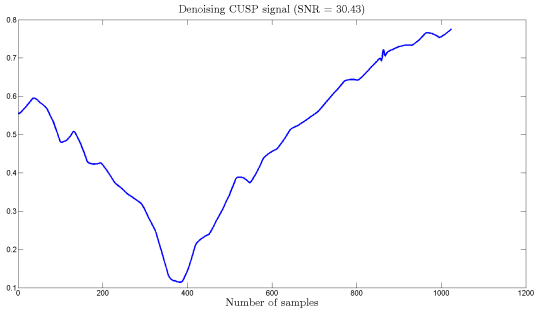
(a) Original signal

(b) Noisy signal

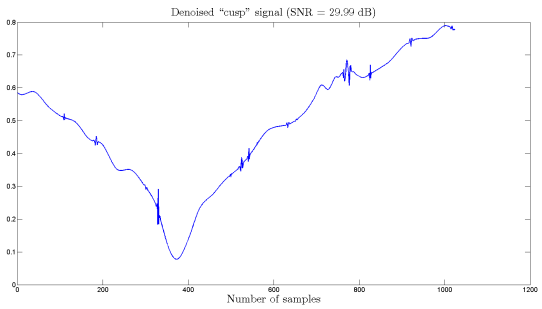
(c) PES-ℓ_1 with pyramid

(d) PES-ℓ_1 with wavelet

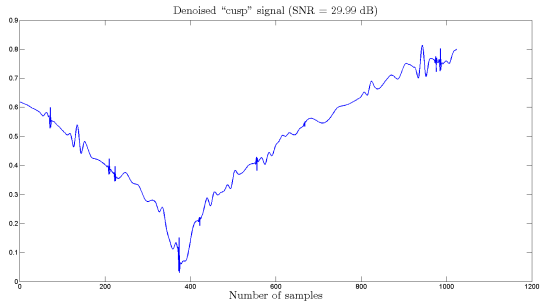
Figure 3.8: (a) Original `cusp` signal, (b) signal corrupted with Gaussian noise with $\sigma = 10\%$ of maximum amplitude of the original signal, and denoised signal using (c) PES-ℓ_1-ball with pyramid; SNR = 23.84 dB and, (d) PES-ℓ_1-ball with wavelet; SNR = 23.79 dB, *(cont.)*



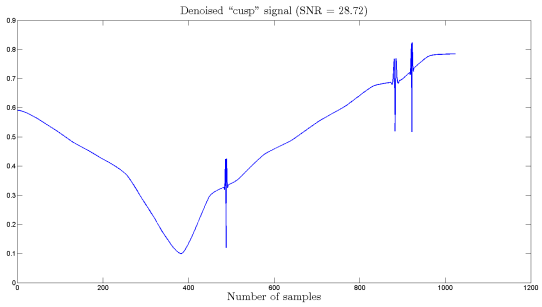
(e) Wavelet denoising in MATLAB [1]



(f) Wavelet denoising `minimaxi` algorithm [2]

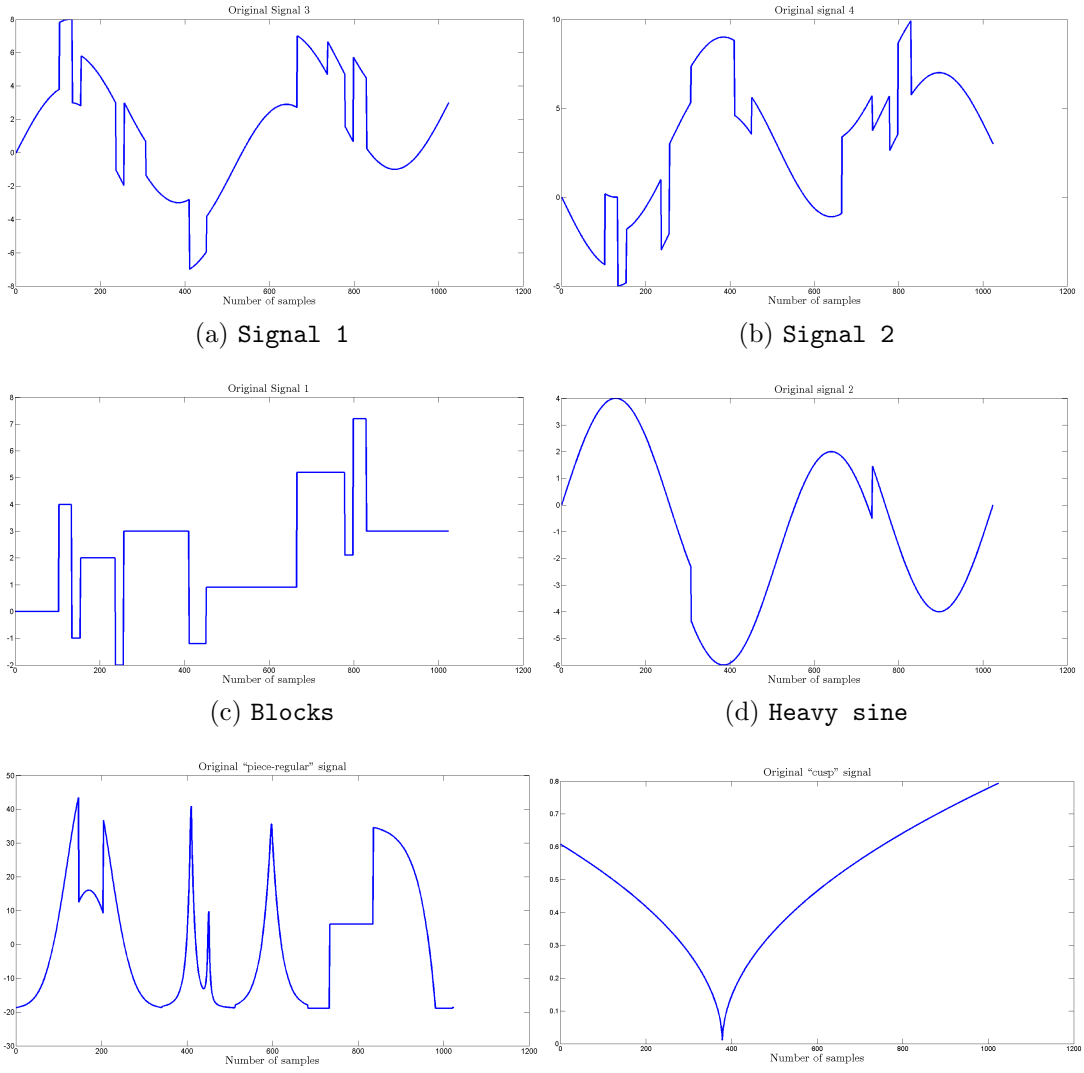


(g) Wavelet denoising `rigrsure` algorithm [3]



(h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]

Figure 3.8: (e) Wavelet denoising in Matlab; SNR = 23.52 dB [1], (f) Wavelet denoising `minimaxi` algorithm [2]; SNR = 23.71 dB, (g) Wavelet denoising `rigrsure` algorithm [3]; SNR = 23.06 dB, (h) Wavelet denoising with T = $3\hat{\sigma}$ [2, 4]; SNR = 21.38 dB.



(a) `Signal 1` (b) `Signal 2`

(c) `Blocks` (d) `Heavy sine`

(e) `Piece-regular` (f) `cusp`

Figure 3.9: Signals which are used in the simulations.

Table 3.1: Comparison of the results for denoising algorithms with Gaussian noise with $\sigma = 10$ % of maximum amplitude of original signal.

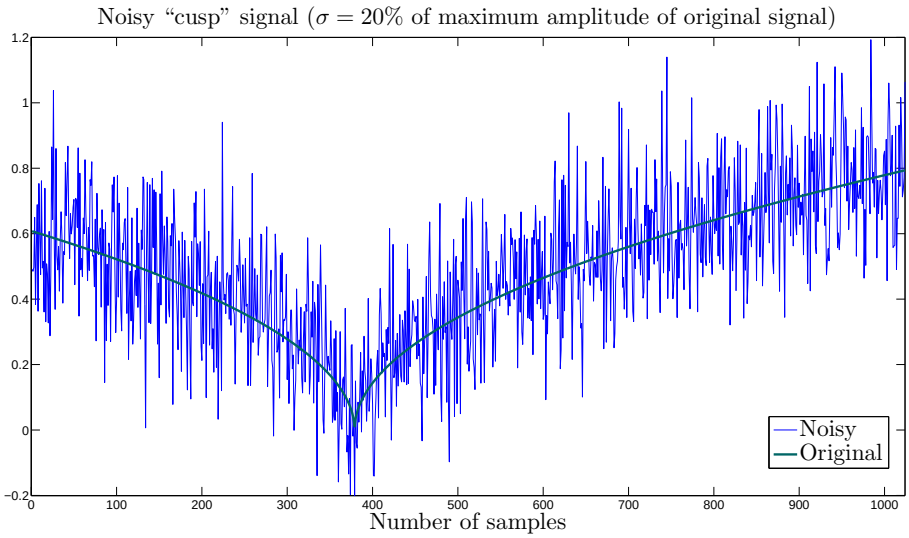
Signal	Input SNR (dB)	**PES-ℓ_1 Pyramid**	PES-ℓ_1 Wavelet	MATLAB [1]	Soft-threshold $3\hat{\sigma}$	MATLAB `rigrsure` [3]	MATLAB `minimaxi` [2]
`Blocks`	12.30	17.27	17.08	15.73	17.64	18.32	16.59
`Heavy sine`	17.77	26.17	26.62	26.87	26.22	26.82	26.75
`Signal 1`	13.09	18.43	18.10	16.63	17.80	19.18	17.41
`Signal 2`	13.83	20.37	19.94	18.39	18.92	20.53	19.08
`Piece-Regular`	12.32	18.53	18.05	16.41	18.16	19.35	17.66
`cusp`	16.29	32.58	29.40	30.43	28.72	29.10	29.99
Average	14.27	**22.23**	21.53	20.74	21.24	22.22	21.25

Table 3.2: Comparison of the results for denoising algorithms with Gaussian noise with $\sigma = 20$ % of maximum amplitude of original signal.

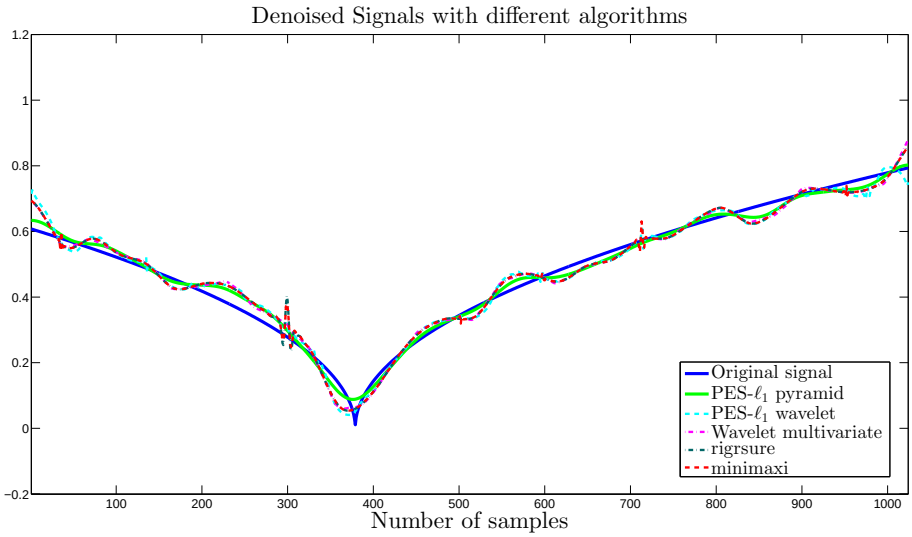
Signal	Input SNR (dB)	**PES-ℓ_1 Pyramid**	PES-ℓ_1 Wavelet	MATLAB [1]	Soft-threshold $3\hat{\sigma}$	MATLAB `rigrsure` [3]	MATLAB `minimaxi` [2]
`Blocks`	6.28	14.34	13.98	12.92	12.87	14.18	13.43
`Heavy sine`	11.75	23.84	23.79	23.52	21.38	23.06	23.71
`Signal 1`	7.07	15.70	15.28	14.00	13.30	15.15	14.42
`Signal 2`	7.80	17.13	17.07	15.84	14.56	16.65	16.20
`Piece-Regular`	6.27	15.24	14.47	13.11	13.14	14.94	13.99
`cusp`	10.25	28.24	24.89	25.04	23.27	23.48	24.44
Average	8.24	**19.08**	18.25	17.41	16.42	17.91	17.70

Table 3.3: Comparison of the results for denoising algorithms with Gaussian noise with $\sigma = 30$ % of maximum amplitude of original signal.

Signal	Input SNR (dB)	**PES-ℓ_1 Pyramid**	PES-ℓ_1 Wavelet	MATLAB [1]	Soft-threshold $3\hat{\sigma}$	MATLAB `rigrsure` [3]	MATLAB `minimaxi` [2]
`Blocks`	2.76	12.52	12.55	11.37	10.13	12.05	11.64
`Heavy sine`	9.20	20.89	21.78	21.32	18.79	20.17	21.05
`Signal 1`	3.56	13.50	13.65	12.37	10.44	13.06	12.72
`Signal 2`	4.26	15.14	14.25	14.06	12.05	14.37	14.30
`Piece-Regular`	2.77	13.21	12.70	11.37	9.94	12.43	12.05
`cusp`	6.73	25.10	23.47	21.73	19.67	19.69	21.02
Average	4.88	**16.73**	16.40	15.37	13.50	15.30	15.46



(a) Original and Noisy Signals



(b) Original and Denoised Signals

Figure 3.10: (a) The `cusp` signal and its corrupted version with Gaussian noise with $\sigma = 20\%$ of maximum amplitude of the original signal, (b) Original signal (blue), denoised signal (green) using PES-ℓ_1-ball with pyramid; SNR = 28.26 dB and, denoised signal (cyan) using PES-ℓ_1-ball with wavelet; SNR = 25.30 dB, denoised signal (magenta) using MATLAB wavelet multivariate method; SNR = 25.08 dB [1], denoised signal (petroleum blue) using wavelet denoising `rigrsure` algorithm [2]; SNR = 23.28 dB, denoised signal (red) using wavelet denoising `minimaxi` algorithm [3]; SNR = 24.52 dB.

Chapter 4

Deconvolution Using PESC and its Applications on Medical Image Processing

In image processing, the deconvolution refers to act of removing the effect of blurring filter or point spread function (PSF) and enhance the quality of the image. In this chapter a deconvolution algorithm based on PESC algorithm is presented. The TV constraint is imposed to the estimated image at each step of the iterative deconvolution algorithm in order to regularized the image and remove noise from it. The PES-TV based deconvolution algorithm is described in detail in the following sections. The simulation results are also presented at the end of this chapter.

4.1 Deconvolution Using PESC

In this section, a new deconvolution method, based on the epigraph set of the convex cost function is presented. It is possible to use TV, FV and ℓ_1-norm as the convex cost function. Let the original signal or image be $\mathbf{w}_{\text{orig}}$ and its blurred

and noisy version be $\mathbf{z}$:

$$\mathbf{z} = \mathbf{w}_{\text{orig}} * \mathbf{h} + \boldsymbol{\eta}, \tag{4.1}$$

where $\mathbf{h}$ is the blurring matrix and $\boldsymbol{\eta}$ is the additive Gaussian noise. In this approach we solve the following problems:

$$\underline{\mathbf{w}}^{\star} = \arg\min_{\underline{\mathbf{w}} \in \mathcal{C}_f} \|\underline{\mathbf{v}} - \underline{\mathbf{w}}\|^2, \tag{4.2}$$

where, $\underline{\mathbf{v}} = [\mathbf{v}^T \ 0]$ and $\mathcal{C}_f$ is the epigraph set of TV or FV in $\mathbb{R}^{N+1}$. Here we repeat Eq. 2.3, the TV function, which we used for an $M \times M$ discrete image $\mathbf{w} = [\mathrm{w}^{i,j}], \ \ 0 \leq i, j \leq M-1 \ \in \mathbb{R}^{M \times M}$:

$$TV(\mathbf{w}) = \sum_{i,j} (|\mathrm{w}^{i+1,j} - \mathrm{w}^{i,j}| + |\mathrm{w}^{i,j+1} - \mathrm{w}^{i,j}|). \tag{4.3}$$

To estimate this problem we use POCS framework using the following sets:

$$\mathcal{C}_i = \{\mathbf{w} \in \mathbb{R}^{N+1} | z_i = (\mathbf{w} * \mathbf{h})[i]\} \qquad i = 1, 2, ..., L, \tag{4.4}$$

where L is the number of pixels and z_i is the i^{th} observation; and the epigraph set:

$$\mathcal{C}_f = \{\underline{\mathbf{w}} \in \mathbb{R}^{N+1} | \underline{\mathbf{w}} = [\mathbf{w}^T \ y]^T : \ y \geq TV(\mathbf{w})\}. \tag{4.5}$$

Notice that the sets $\mathcal{C}_i$ are in $\mathbb{R}^N$ and $\mathcal{C}_f$ is in $\mathbb{R}^{N+1}$. However, it is straightforward to extend $\mathcal{C}_i$'s to $\mathbb{R}^{N+1}$ and they are still closed and convex sets in $\mathbb{R}^{N+1}$. Let us describe the projection operation onto the set $\mathcal{C}_f = \{TV(\mathbf{w}) \leq y\}$, briefly. Notice that, this $\mathcal{C}_f$ is different from the set $\{\|\mathbf{v} - \mathbf{w}\|^2 + \lambda \mathrm{TV}(\mathbf{w}) \leq y\}$. This means that we select the nearest vector $\underline{\mathbf{w}}^{\star}$ on the set $\mathcal{C}_f$ to $\mathbf{v}$. This is graphically illustrated in Figure 4.1 (repeated from Figure 2.1). During this orthogonal projection operations, we do not require any parameter adjustment as in [38]. The proposed deconvolution algorithm consists of cyclical projections onto the sets $\mathcal{C}_i$ and $\mathcal{C}_f$.

Projection onto the sets are very easy to compute because they are hyperplanes:

$$\mathbf{v}_{r+1} = \mathbf{v}_r + \frac{z_i - (\mathbf{v}_r * \mathbf{h})[i]}{\|\mathbf{h}\|^2} \mathbf{h}^T, \tag{4.6}$$

where v_r is the r^{th} iterate, v_{r+1} is the projection vector onto the hyperplane $\mathcal{C}_i$. This operation is illustrated graphically in Figure 4.2. In this figure, starting from

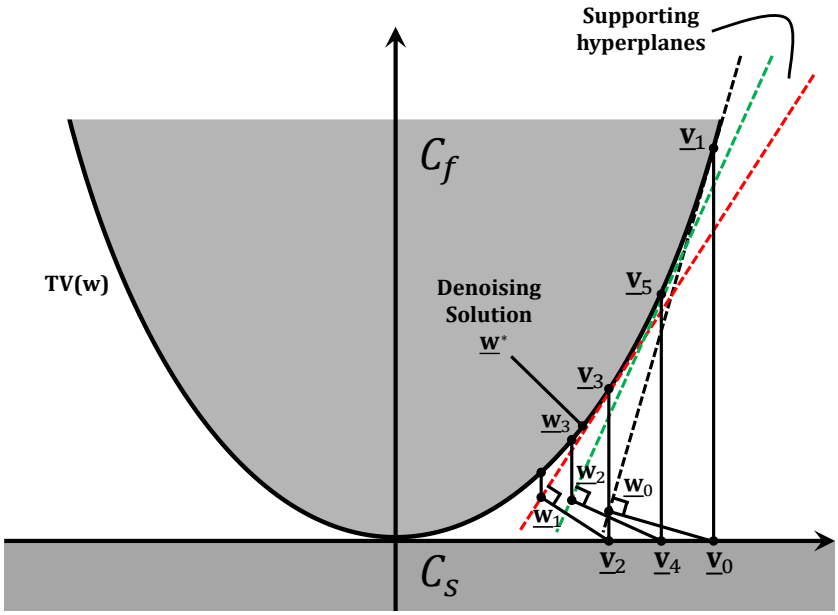


Figure 4.1: Graphical representation of the minimization of Eq. (4.2), using projections onto the supporting hyperplanes of $\mathcal{C}_f$. In this problem the sets $\mathcal{C}_s$ and $\mathcal{C}_f$ intersect because $TV(\mathbf{w}) = 0$ for $\mathbf{w} = [0, 0, ..., 0]^T$ or for a constant vector.

initial estimate $\mathbf{v}_1$, the projections converge to the intersection of the hyperplanes after some iterations. The pseudo-code of the algorithm is described in Algorithm 2.

The sets $\mathcal{C}_i$ and $\mathcal{C}_f$ may or may not intersect in $\mathbb{R}^{N+1}$. If they intersect, iterates converge to a solution in the intersection set. It is also possible to use hyperslabs, $\mathcal{C}_{i,h} = \{\mathbf{w} | z_i - \epsilon_i \leq (\mathbf{w} * \mathbf{h})[i] \leq z_i + \epsilon_i\}$, instead of hyperplanes $\mathcal{C}_i$ in this algorithm. In this case it is more likely that the closed and convex sets of the proposed framework intersect.

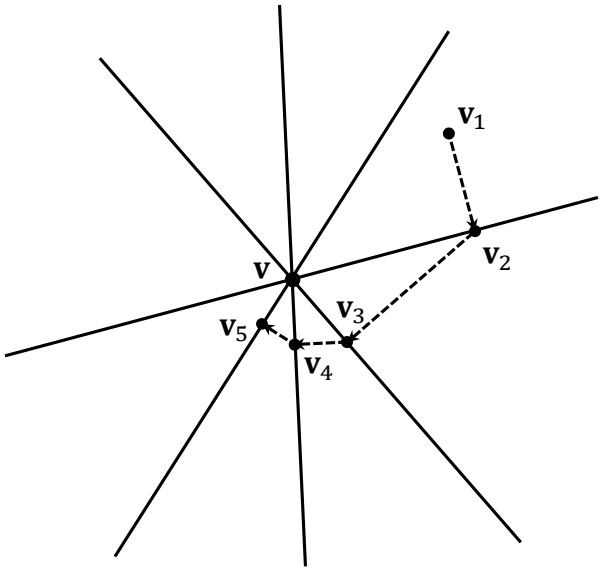


Figure 4.2: Graphical representation of the projections onto hyperplanes described in (4.6).

Algorithm 2 The pseudo-code for the deconvolution with PESC based algorithm

Begin

$\mathbf{z} \in \mathbb{R}^{N \times N}$, $\mathbf{h} \in \mathbb{R}^{N_h \times N_h}$, $K \in \mathbb{Z}^+$

v $\leftarrow$ z

for $k = 1$ to K

for $x = 1$ to N

for $y = 1$ to N

$\mathbf{v}(x - \lfloor N_h/2 \rfloor \ to \ x + \lfloor N_h/2 \rfloor, y - \lfloor N_h/2 \rfloor \ to \ y + \lfloor N_h/2 \rfloor) \leftarrow \mathbf{v}(x - \lfloor N_h/2 \rfloor \ to \ x + \lfloor N_h/2 \rfloor, y - \lfloor N_h/2 \rfloor \ to \ y + \lfloor N_h/2 \rfloor) + \frac{z(x,y) - v * h|_{x,y}}{\|h\|^2} h$

endfor

endfor

while $||\mathbf{w} - \mathbf{v}|| > \epsilon$

$\mathbf{w} \leftarrow$ Project $\mathbf{v}$ onto $\mathcal{C}_f$

$\mathbf{w} \leftarrow$ Project $\mathbf{w}$ onto $\mathcal{C}_s$

endwhile

endfor

Implementation: The sub-gradient projections of $\mathbf{v}_n$ are performed as in Eq. 4.6. Then after a loop of these projections are terminated, the PESC algorithm will be applied to the output $\mathbf{v}_n$. The minimization operation described in Eq. (4.2) can not be obtained in one step when the cost function is TV. The solution is determined by performing successive orthogonal projections onto supporting hyperplanes of the epigraph set $\mathcal{C}_f$. In the first step, $\mathrm{TV}(\mathbf{v}_0)$ and the surface normal at $\underline{\mathbf{v}}_1 = [\mathbf{v}_0^T \quad \mathrm{TV}(\mathbf{v}_0)]$ in $\mathbb{R}^{N+1}$ are calculated. In this way, the equation of the supporting hyperplane at $\underline{\mathbf{v}}_1$ is obtained. The vector $\underline{\mathbf{v}}_0 = [\mathbf{v}_0^T \quad 0]$ is projected onto this hyperplane and $\underline{\mathbf{w}}_1$ is obtained as our first estimate as shown in Figure 2.1. In the second step, $\underline{\mathbf{w}}_1$ is projected onto the set $\mathcal{C}_s$ by simply making its last component zero. The TV of this vector and the surface normal, and the supporting hyperplane are calculated as in the previous step. Next, $\underline{\mathbf{v}}_0$ is projected onto the new supporting hyperplane, and $\underline{\mathbf{w}}_2$ is obtained. In Figure 2.1, $\underline{\mathbf{w}}_2$ is very close to the denoising solution $\underline{\mathbf{w}}^\star$. In general iterations continue until $\|\underline{\mathbf{w}}_i - \underline{\mathbf{w}}_{i-1}\| \leq \epsilon$, where ϵ is a prescribed number, or iterations can be stopped after a certain number of iterations.

We calculate the distance between $\underline{\mathbf{v}}_0$ and $\underline{\mathbf{w}}_i$ at each step of the iterative algorithm described in the previous paragraph. The distance $\|\underline{\mathbf{v}}_0 - \underline{\mathbf{w}}_i\|^2$ does not always decrease for high i values. This happens around the optimal denoising solution $\underline{\mathbf{w}}^\star$. Once we detect an increase in $\|\underline{\mathbf{v}}_0 - \underline{\mathbf{w}}_i\|^2$, we perform a refinement step to obtain the final solution of the denoising problem. In refinement step, the supporting hyperplane at $\mathbf{v}_{2i-1} = \frac{\mathbf{v}_{2i-5} + \mathbf{v}_{2i-3}}{2}$ is used in the next iteration. A typical convergence graph is shown in Figure 2.2 for the "note" image.

It is possible to obtain a smoother version of $\underline{\mathbf{w}}^\star$ by simply projecting $\underline{\mathbf{v}}$ inside the set $\mathcal{C}_f$ instead of the boundary of $\mathcal{C}_f$.

4.2 Simulation Results

The PESC algorithm is tested with standard images. The noise standard deviation σ is chosen so that the averaged blurred signal to noise ratio $BSNR$ reaches

a target value:

$$\text{BSNR} = 10 \times log_{10}(\frac{\|\widetilde{\mathbf{z}} - E[\widetilde{\mathbf{z}}]\|^2}{N\sigma_\eta^2}), \tag{4.7}$$

where $\widetilde{\mathbf{z}}$ is the blurred image without noise $\widetilde{\mathbf{z}} = \mathbf{w}_{orig} * \mathbf{h}$, N is the whole number of pixels, and σ is the additive noise's standard deviation. In addition to the visual results, the deblurring algorithm is compared in term of Improved Signal to Noise Ratio (ISNR) as follows:

$$\text{ISNR} = 10 \times log_{10}(\frac{\|\mathbf{z} - \mathbf{w}_{orig}\|^2}{\|\mathbf{w}_{rec} - \mathbf{w}_{orig}\|^2}), \tag{4.8}$$

which $\mathbf{w}_{rec}$ is the reconstructed and deblurred image. The ISNR as a function of iteration number for the experiment done over MRI image is given in Figure 4.3.

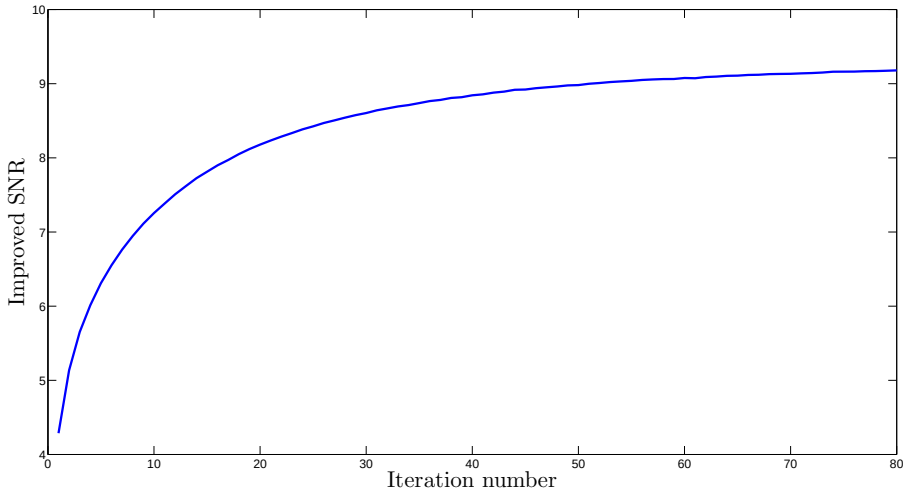
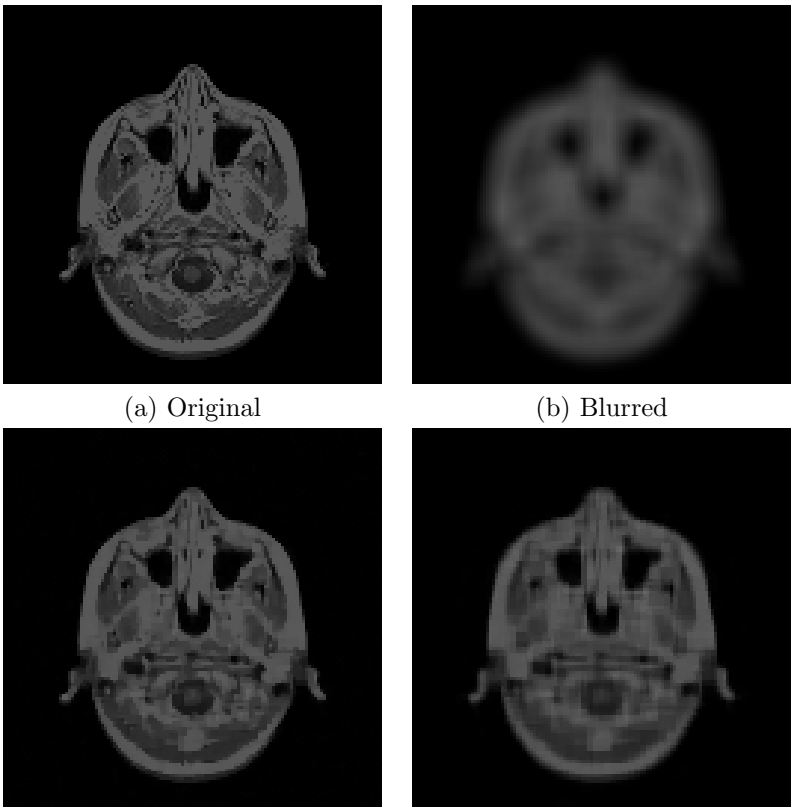


Figure 4.3: ISNR as a function of the iteration number for MRI image.

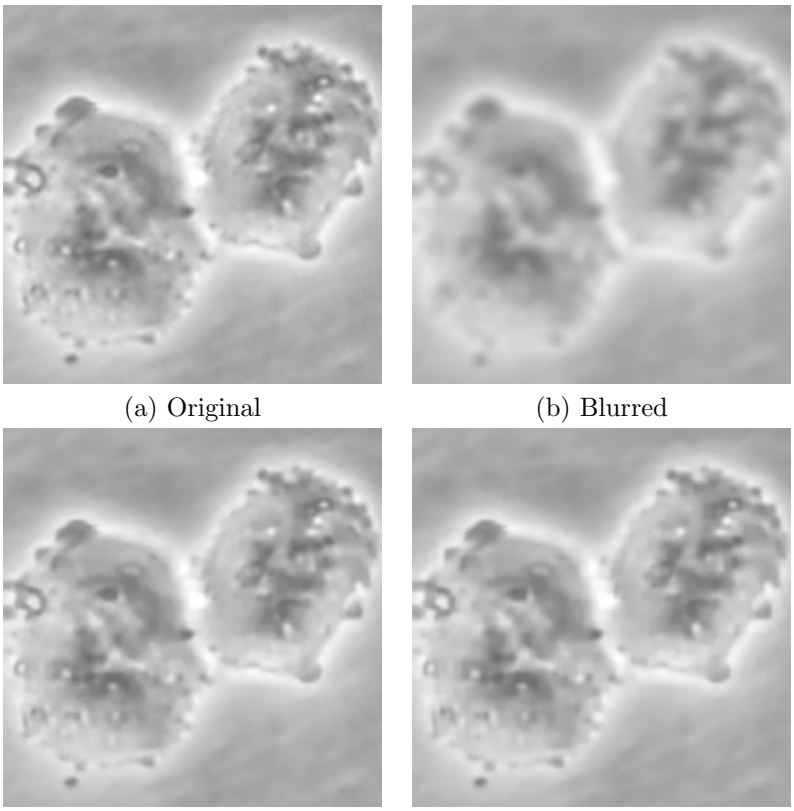
Table 4.1 and 4.2 represent the ISNR and SNR values for five BSNR values for PESC algorithm and FTL algorithm proposed by Vonesch *etal* [50]. Table 4.3 represents SNR and ISNR values for five different microscopic cancer cell images for PESC and FTL algorithms for BSNR = 45. According to the these tables, in almost all cases PESC based deconvolution algorithm performs better than FTL [50] in sense of ISNR and SNR.



(a) Original (b) Blurred

(c) PESC (d) FTL

Figure 4.4: Sample image used in our experiments (a) Original, (b) Blurred (BSNR = 50), (c) Deblurred by PESC (SNR = 18.53 dB), (d) Deblurred by FTL (SNR = 14.92 dB).



(a) Original (b) Blurred

(c) PESC (d) FTL

Figure 4.5: Cancer cell image (a) Original, (b) Blurred (BSNR = 50), (c) Deblurred by PESC (SNR = 40.58 dB), (d) Deblurred by FTL (SNR = 39.35 dB).

Table 4.1: ISNR and SNR results for PESC based deconvolution algorithm.

BSNR	Cameraman		Lena		Peppers		Pirate		Mandrill		MRI		Cancer cell	
30	**5.59**	**20.83**	**4.48**	**24.94**	**5.35**	**26.13**	**4.57**	**22.77**	**4.56**	**20.77**	**4.64**	**13.71**	**6.26**	**35.94**
35	**7.01**	**22.28**	**5.77**	**26.26**	**5.88**	**26.72**	5.55	23.28	**5.61**	**21.83**	**5.76**	**14.90**	**7.76**	**37.69**
40	**8.49**	**23.77**	**6.95**	**27.46**	7.45	28.32	**6.75**	**24.99**	**6.41**	**22.65**	**7.07**	**16.28**	**9.05**	**38.79**
45	**9.75**	**25.04**	**8.03**	**28.55**	**8.52**	**29.39**	**7.87**	**26.12**	**6.72**	**22.95**	**8.40**	**22.95**	**9.76**	**39.51**
50	**10.76**	**26.10**	**8.49**	**29.00**	**9.50**	**30.37**	**8.41**	**26.66**	**6.84**	**23.07**	**9.31**	**18.53**	**10.83**	**40.58**

Table 4.2: ISNR and SNR results for FTL based deconvolution algorithm.

BSNR	Cameraman		Lena		Peppers		Pirate		Mandrill		MRI		Cancer cell	
30	-0.4	14.79	-0.74	19.7	-3.26	17.2	0.71	18.74	4.32	20.52	4.08	13.02	4.66	34.34
35	6.16	21.35	5.46	25.97	5.66	26.11	**5.61**	**23.68**	5.45	21.65	5.03	14.36	7.65	37.58
40	7.54	22.73	6.60	27.13	**8.00**	**28.45**	6.44	24.50	5.74	21.94	5.45	14.73	8.98	38.72
45	7.89	23.08	6.93	27.46	8.14	29.02	6.67	24.75	5.89	22.08	5.56	14.86	9.44	39.19
50	8.04	23.23	7.07	27.59	8.74	29.20	6.77	24.84	5.98	22.18	5.61	14.92	9.59	39.35

In an example, in Figure 4.4 the original, blurred, and deblurred images for a MRI image, for both algorithms are presented. In this image the original MRI image is blurred using 9×9 Gaussian blurring matrix and is corrupted with additive white Gaussian noise with such a variance to obtain BSNR = 50 value. In another example in Figure 4.5 the results for cancer cell image is presented. The original image is blurred with 9×9 uniform blurring matrix and is corrupted with additive white Gaussian noise with such a variance to obtain BSNR = 50 value. The blurred image, and the deblurred images for both algorithms are presented in Figure 4.5. According to this figure, PESC algorithm performs better than FTL not only in sense of SNR, but also the results for PESC are visually better than FTL.

Table 4.3: ISNR and SNR results for PESC and FTL based deconvolution algorithms for BSNR = 45.

Image	PESC		FTL	
Cancer cell-1	**9.71**	**42.40**	8.23	40.91
Cancer cell-2	**10.47**	**41.87**	8.79	40.16
Cancer cell-3	**10.55**	**40.86**	8.93	39.22
Cancer cell-4	**9.02**	**42.09**	7.63	40.73
Cancer cell-5	**9.23**	**42.81**	7.86	41.43

Chapter 5

Compressive Sensing Using PESC

The aim in compressive sensing problems is to recover the image from a limited number of samples taken from the original image such that the output image includes as much information as possible. In this chapter, a compressive sensing algorithm based on PESC algorithm is presented. This algorithm is illustrated in detail in the following sections, including the extensive simulation results at the end of the chapter.

5.1 Compressive Sensing Problem

In transform based signal and image coding, a given signal $\mathbf{x}$ is transformed into another domain defined by the orthogonal transformation matrix $\boldsymbol{\psi}$. The transformation procedure is simply finding the inner product of the signal $\mathbf{x}$ with the rows $\boldsymbol{\psi_l}$ of the transformation matrix $\boldsymbol{\psi}$ represented as follows:

$$s_l = \langle \mathbf{x}, \boldsymbol{\psi_l} \rangle \quad l = 1, 2, ..., N, \tag{5.1}$$

where $\mathbf{x}$ is a column vector of size N.

The discrete-time signal can be reconstructed from its transform coefficients s_l, as follows:

$$\mathbf{x} = \sum_{l=1}^{N} s_l.\psi_l \quad \text{or} \quad \mathbf{x} = \boldsymbol{\psi}^T.\mathbf{s}, \tag{5.2}$$

where $\mathbf{s}$ is a vector containing the transform domain coefficients s_l and $\boldsymbol{\psi}^T$ is the inverse transform matrix.

The basic idea in digital waveform coding is that the signal should be approximately reconstructed from only a few of its non-zero transform coefficients. In most cases, including the JPEG image coding standard, the transform matrix $\boldsymbol{\psi}$ is chosen in such a way that the new signal $\mathbf{s}$ is efficiently represented in the transform domain with a small number of coefficients.

The CS theory introduced in [39,69–72] provides answers to the question of reconstructing a signal from its compressed measurement vector $\mathbf{v}$, which is defined as follows:

$$\mathbf{v} = \boldsymbol{\phi}\mathbf{x} = \boldsymbol{\phi}.\boldsymbol{\psi}^T.\mathbf{s} = \boldsymbol{\theta}.\mathbf{s}, \tag{5.3}$$

where $\boldsymbol{\phi}$ is the $M \times N$ measurement matrix and $M \ll N$. Reconstruction of the original signal $\mathbf{x}$ from its compressed measurements $\mathbf{v}$ cannot be achieved by simple matrix inversion or inverse transformation techniques. To recover $\mathbf{x}$, an iterative method is used. This iterative method is illustrated in the following section.

5.2 PESC Based Compressive Sensing Algorithm

Instead of solving the CS problem using ℓ_0-norm and ℓ_1-norm minimization, other methods were developed in the literature [73–75]. For example, in [73], a Bayesian solution to the CS problem is obtained. One popular approach is to replace ℓ_0-norm with ℓ_p-norm, where p $\in$ (0, 1] or as a combination of two different norms [8,53,74,75]. We use the epigraph of ℓ_1-norm cost function, the TV or any

other convex cost function together with the measurement hyperplanes to solve this problem. Let the epigraph set be

$$\mathcal{C}_f = \{\underline{\mathbf{s}} = [\mathbf{s}^T y]^T : f(\mathbf{s}) \leq y\} \tag{5.4}$$

where $f(\mathbf{s})$ can be the convex cost function representing the ℓ_1-norm or the TV function, and the measurement hyperplane sets are defined as follows:

$$\mathcal{C}_i = \{\mathbf{s}^T.\boldsymbol{\theta_i} = v_i\}, \quad i = 1, 2, ..., L, \tag{5.5}$$

where $\boldsymbol{\theta_i}$ is the i^{th} row of the $\boldsymbol{\theta}$ matrix and v_i is the i^{th} entry of the observation vector $\mathbf{v}$. All of the above sets are closed and convex sets. Therefore, it is possible to device an iterative signal reconstruction algorithm in $\mathbb{R}^{N+1}$ by making successive orthogonal projections onto the sets $\mathcal{C}_f$ and $\mathcal{C}_i,\ i = 1, 2, ..., L$. The crucial step of this algorithm providing regularization is the projection onto the set $\mathcal{C}_f$.

5.2.1 Projection onto the set $\mathcal{C}_f$

Let $\mathbf{s}_o$ be a vector in $\mathbb{R}^N$ and the corresponding augmented vector be $\underline{\mathbf{s}}_o = [\mathbf{s}_o^T\ 0] \in \mathbb{R}^{N+1}$. The projection vector $\mathbf{s}_p$ is obtained as follows:

$$\underline{\mathbf{s}}_p = \begin{bmatrix} \mathbf{s}_p \\ f(\mathbf{s}_p) \end{bmatrix} = \underset{\underline{\mathbf{s}}}{\operatorname{argmin}} \|\mathbf{s} - \mathbf{s}_o\|_2^2 + f(\mathbf{s})^2. \tag{5.6}$$

The projection $\mathbf{s}_p$ for an arbitrary vector $\underline{\mathbf{s}}_o = [\mathbf{s}_{o,N}^T\ s_{o,N+1}] \in \mathbb{R}^{N+1}$ is given by

$$\underline{\mathbf{s}}_p = \begin{bmatrix} \mathbf{s}_p \\ f(\mathbf{s}_p) \end{bmatrix} = \underset{\underline{\mathbf{s}}}{\operatorname{argmin}} \|\mathbf{s} - \mathbf{s}_{o,N}\|_2^2 + (s_{o,N+1} - f(\mathbf{s}))^2, \tag{5.7}$$

where $\mathbf{s}_{o,N}$ is a vector containing the first N component of $\underline{\mathbf{s}}_o$ and $s_{o,N+1}$ is the $(N+1)^{st}$ component. To solve the minimization problem, the PESC algorithm is used. Here, the convex cost function is TV function: $f(\mathbf{s}) = \mathrm{TV}(\mathbf{s})$. In PES-TV algorithm the aim is to minimize the distance between the observation image $\mathbf{s}_0$ and the epigraph set of the TV function. In other words, the aim is to find the nearest $\mathbf{s}$ on the surface of the TV function to $\mathbf{s}_0$, as shown in Figure 5.1. This process is defined in Chapter 2.

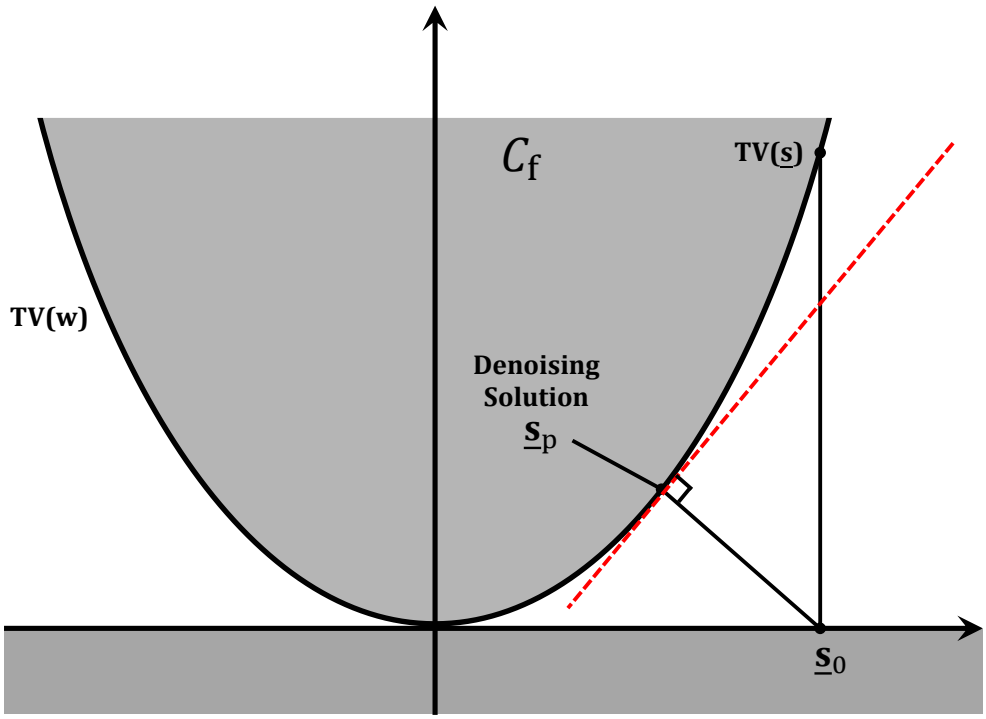


Figure 5.1: Graphical representation of PES-TV algorithm.

5.2.2 CS Algorithm

In this case, we replace the level set $\mathcal{C}_s$ in Section 2.1 with the measurement hyperplanes $\mathcal{C}_i$, and the epigraph set is as follows:

$$\mathcal{C}_f = \{\mathbf{s} \mid \mathrm{TV}(\mathbf{s}) \leq \epsilon\}. \tag{5.8}$$

The hyperplanes in general form an undetermined set of equations. As a result their intersection

$$\mathcal{C}_{int} = \bigcap_i \mathcal{C}_i \tag{5.9}$$

is highly unlikely to be an empty set. Individual $\mathcal{C}_i's$ may intersect with the epigraph set $\mathcal{C}_f$ but the intersection of hyperplanes $\mathcal{C}_{int}$ may not intersect with the epigraph set $\mathcal{C}_f$. This scenario has not been studied in POCS theory to the best of our knowledge [36, 40] but it is very similar to the scenario that we discussed in Section 2.1. As we point out in Section 1.3, two nonintersecting convex sets case studied by Gubin *et.al* [36]. Therefore, we expect that iterates

oscillate between a vector in the intersection of hyperplanes $\mathcal{C}_{int}$ and a vector in the epigraph set $\mathcal{C}_f$. We conjecture that the iterates oscillate between the nearest vectors in sets $\mathcal{C}_f$ and $\mathcal{C}_{int}$ similar to the case discussed in Section 2.1. Therefore, we essentially obtain a solution to the following problem:

$$min||\mathbf{s}_f - \mathbf{s}||_2 \quad \text{such that} \quad \mathbf{s}_f \in \mathcal{C}_f \quad \text{and} \quad \mathbf{s} \in \mathcal{C}_{int}. \tag{5.10}$$

If the sets $\mathcal{C}_f$ and $\mathcal{C}_{int}$ intersect, the iterates converge to a vector in the intersection set $\mathcal{C}_{int} \cap \mathcal{C}_f$ by Bregman's POCS theorem.

The iterative algorithm consists of performing successive orthogonal projections onto hyperplanes corresponding to measurements $v_i = \mathbf{s}^T.\boldsymbol{\theta}_i$ for $i = 1, 2, ..., L$, followed by an orthogonal projection onto the epigraph set $\mathcal{C}_f$. The algorithm is described in Algorithm 3.

If the initial condition is chosen as $\underline{\mathbf{s}}_0 = \begin{bmatrix} \mathbf{s}_0 \\ 0 \end{bmatrix}$ then Eq. (5.6) is used to implement the projection onto the epigraph set. The algorithm is essentially similar to the proximity operator based algorithms [76, 77]. However, $f^2(s)$ is used instead of $f(s)$ as the regularizing term as shown in (5.6).

In our approach it is also possible to define a smoothing parameter in both denoising and compressive sensing solutions as well. The epigraph set $\mathcal{C}_f$ can be modified as follows:

$$\mathcal{C}_{f,\alpha} = \{\underline{\mathbf{s}} : \ y \geq \alpha TV(\mathbf{s})\}. \tag{5.11}$$

The choice of the parameter $\alpha > 1$ provides smoother solution than usual and $\alpha < 1$ relaxes the smoothing constraint. In this case, Eq. (5.6) becomes

$$\underline{\mathbf{s}}_p = \begin{bmatrix} \mathbf{s}_p \\ f(\mathbf{s}_p) \end{bmatrix} = argmin\|\mathbf{s} - \mathbf{s}_o\|_2^2 + \alpha^2 f(\mathbf{s})^2. \tag{5.12}$$

It is experimentally observed that $\alpha = 1$ usually provides better denoising results than the manually selected best λ values in standard TV denoising introduced in [38].

Algorithm 3 The pseudo-code for the compressive sensing with PESC based algorithm

Input

$\mathbf{s}_0 \in \mathbb{R}^N$

for $i = 1$ to M,

for $j = 0$ to L, (L: number of measurements)

$\mathbf{s}_{j+1}$ = Project $\mathbf{s}_j$ onto $\mathcal{C}_{i,j}$

endfor

endfor

$\mathbf{w}$ = Project $\underline{\mathbf{s}}_L$ onto $\mathcal{C}_f$, $(\underline{\mathbf{s}}_L \in \mathbb{R}^{N+1})$

if $||\mathbf{w} - \mathbf{s}_L|| \leq \epsilon$

Terminate

else

Go to first **for**

endif

5.3 Simulation Results

The PESC algorithm is tested with one-dimensional (1D) signals, and 29 two-dimensional (2-D) images (24 images from Kodak database [60], and 5 standard image processing images) in compressive sensing examples. In all the experiments, the measurement matrices are chosen as zero mean Gaussian random matrices.

Consider the `cusp` and `piecewise-smooth` signals shown in Figure 5.2 and 5.3 (blue curves), respectively. These signals consist of 1024 samples. In the DCT domain, both signals can be approximated in a sparse manner. In the first set of experiments, the original signals are reconstructed with M = 204 and 717 measurements with SNR values of 45 and 58 dB for `cusp` signal and 22 and 42 dB for `piecewise-smooth` signal, respectively. The reconstructed signals using the TV cost functional based PESC algorithm are shown in Figures 5.2*a*, and 5.2*b* for `cusp` signal, and in Figures 5.3a and 5.3b for the `piecewise-smooth` signal.

PESC algorithm is compared with four well known CS reconstruction algorithms from the literature; CoSamp [52], $\ell_1$magic [69], Matching Pursuit (MP) [51], and ℓ_p optimization based CS reconstruction [53] algorithms. Three different values for p = 0.8, 1, and 1.7 are used in ℓ_p optimization based CS reconstruction algorithm.

All CS reconstruction algorithms are implemented with different number of measurements ranging from 10% to 80% of the total number of the samples of the 1D signal. The main region of interest in these experiments is $20\% - 60\%$ range. Reconstruction SNR versus the number of measurements are plotted for the `cusp` and `piecewise-smooth` signals in Figure 5.6 and Figure 5.7, respectively. To eliminate the effect of zero mean random Gaussian measurement matrices, the experiments are repeated 10 times and averaged to obtain the plots. The PESC algorithm performs better than other algorithms.

Both `cusp` and `piecewise-smooth` signals are compressible signals in DCT domain. On the other hand, the impulsive signal shown in Figure 5.4 is not compressible in DCT domain, but it is sparse in time domain. The random impulse signal with 256 samples and 25 impulses in random sample indexes is reconstructed with PESC algorithm from 60% of measurements. The reconstructed signal is also shown in Figure 5.4. Obviously, the TV function is not suitable for such signals. Smooth signal assumption of the TV cost function definitely fails in this signal. The reconstructed signal is not perfect as shown in Figure 5.5, because the random impulse signal contains isolated impulses. The SNR curve due to the TV based PESC method is well below the other algorithms in Figure 5.5.

The epigraph of ℓ_1-norm should be used for this signal. When we use the epigraph of ℓ_1-norm together with the mean value constraint set

$$\mathcal{C}_\mu = \{\underline{\mathbf{x}} : \frac{1}{N}\sum_n x[n] = \mu\}, \tag{5.13}$$

we obtain Figure 5.4. It is very easy to estimate the mean value of the signal by using a compressive measurement with a random vector containing random variables with a nonzero mean. We need $\mathcal{C}_\mu$ because we have to rescale the signal

after the projections onto the epigraph set of ℓ_1-norm. The signal is reconstructed from 153 measurements in Figure 5.4.

Table 5.1: Comparison of the results for compressive sensing with Fowler's algorithm and the PESC Algorithm (20% of measurements)

Image	Fowler B=32	Fowler B=64	**PESC B=32**	**PESC B=64**
Kodak(ave.)	20.00	19.94	20.36	21.06
Mandrill	15.72	15.79	15.40	15.85
Lena	24.81	24.74	24.21	24.94
Barbara	18.49	19.66	17.28	17.59
Peppers	23.98	23.43	25.26	26.35
Goldhill	22.05	21.93	21.75	22.62
Average	20.17	20.11	**20.43**	**21.13**

Our second set of experiments consists of CS reconstruction with 2D signals. Five well known images (Peppers, Mandrill, Lena, Barbara, Goldhill) from the image processing literature and 24 images from the "Kodak True Color images" database [63] are used in Tables 5.2, 5.1, and 5.3. These tables compare the SNR values for compressive sensing with PESC and Fowler's algorithm [78] for these images with two block-sizes of 32×32 and 64×64 for both algorithms. Images in Kodak dataset are 24 bit per pixel color images. All the color images are transformed into YUV color space and the 8 bit per pixel luminance component (Y channel) is used in our tests. Since all the images are natural images they are all compressible in DCT domain.

Fowler's algorithm is a block based compressed sensing algorithm therefore, we also divided the images into blocks and reconstructed those blocks individually. Random measurements, which are 20, 30, 40% of the total number of pixels in images, are used in Tables 5.1, 5.2, and 5.3 on both the TV based PESC algorithm and Fowler's method, respectively. On average, for 30% of measurement and for 64×64, and 32×32 blocks, we achieve approximately 1.24 dB, and 0.42 dB higher SNR respectively, compared to Fowler's algorithm. In Figure 5.8, a portion of "peppers" and "goldhill" images are presented. The CS results for

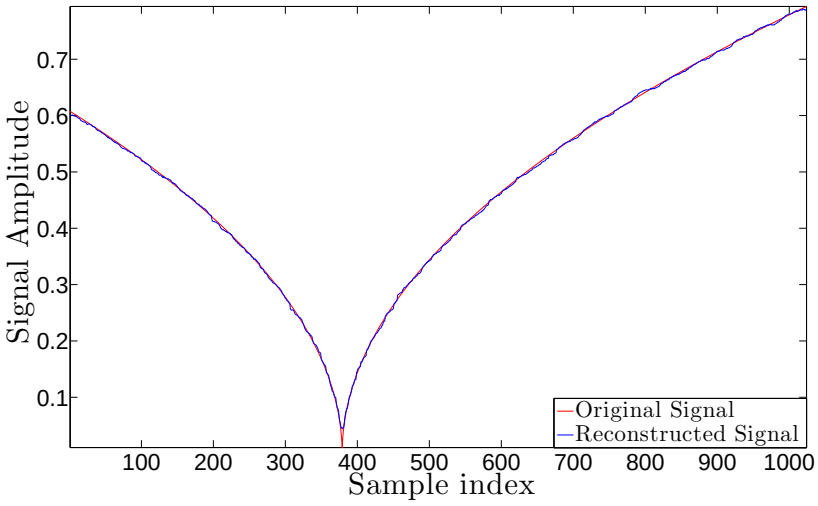
Table 5.2: Comparison of the results for compressive sensing with Fowler's algorithm and the PESC Algorithm (30% of measurements)

Image	Fowler B=32	Fowler B=64	**PESC B=32**	**PESC B=64**
Kodak(ave.)	21.40	21.36	21.96	22.74
Mandrill	16.47	16.77	16.65	16.96
Lena	26.82	26.71	26.02	26.86
Barbara	20.05	20.40	18.21	18.62
Peppers	24.66	24.46	27.06	27.93
Goldhill	22.78	23.44	23.64	24.24
Average	21.53	21.53	**22.02**	**22.77**

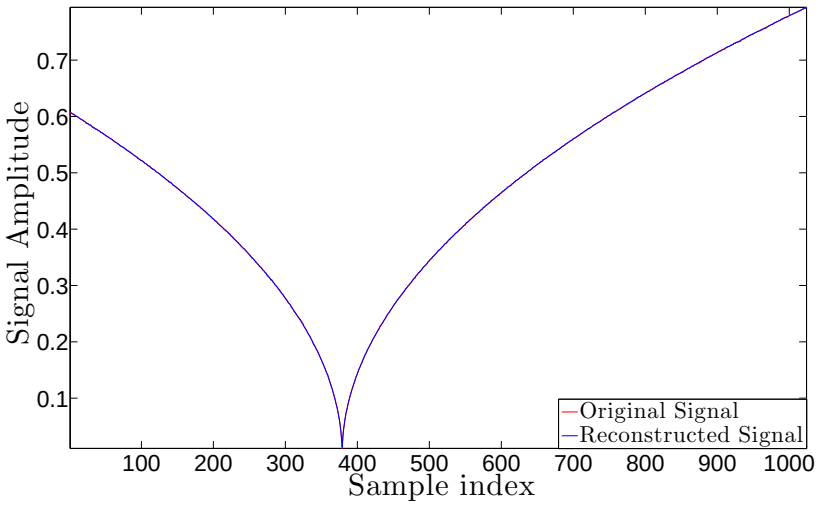
Table 5.3: Comparison of the results for compressive sensing with Fowler's algorithm and the PESC Algorithm (40% of measurements)

Image	Fowler B=32	Fowler B=64	**PESC B=32**	**PESC B=64**
Kodak(ave.)	22.99	22.54	23.07	24.11
Mandrill	17.56	17.80	17.74	18.07
Lena	28.51	28.36	27.27	28.25
Barbara	21.34	22.12	19.00	19.62
Peppers	25.96	29.02	28.00	29.02
Goldhill	24.04	24.80	24.74	25.54
Average	23.08	22.77	**23.15**	**24.11**

30% measurements for PES-TV and Fowler's algorithm, for "peppers" image using 32×32 blocks are presented in Figure 5.9. The same case for "peppers" image using 64×64 blocks are presented in Figure 5.10. Similarly, the results for "goldgill" image is shown in Figures 5.11 and 5.12. The SNR values in Tables 5.2, 5.1, and 5.3, and visual examples in Figures 5.9, 5.10, 5.11, and 5.12 indicates the efficiency of PES-TV based compressive sensing algorithm.

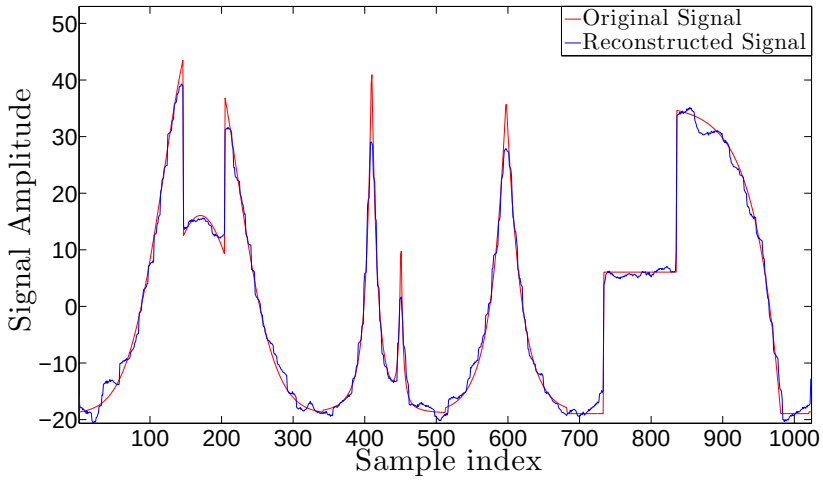


(a)

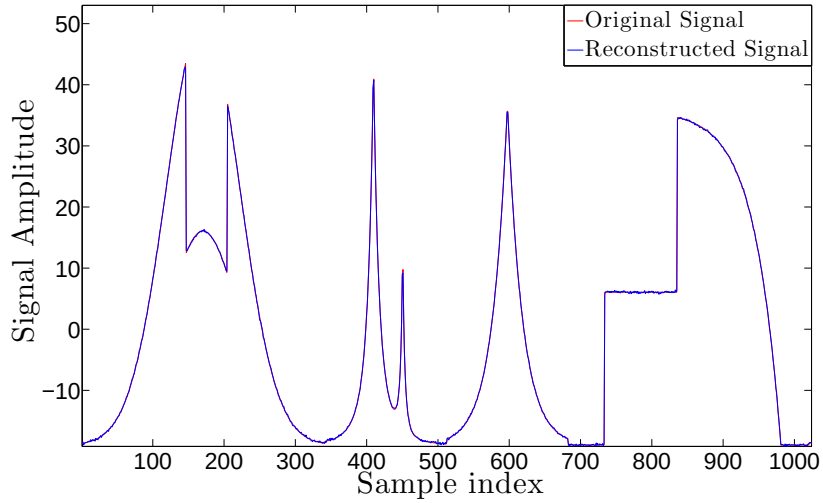


(b)

Figure 5.2: The reconstructed `cusp` signal for (a) 204 measurements (SNR = 45 dB), and (b) 717 measurements (SNR = 58 dB).



(a)



(b)

Figure 5.3: The reconstructed `piecewise-smooth` signal for (a) 204 measurements (SNR = 21.53 dB), and (b) 717 measurements (SNR = 42 dB).

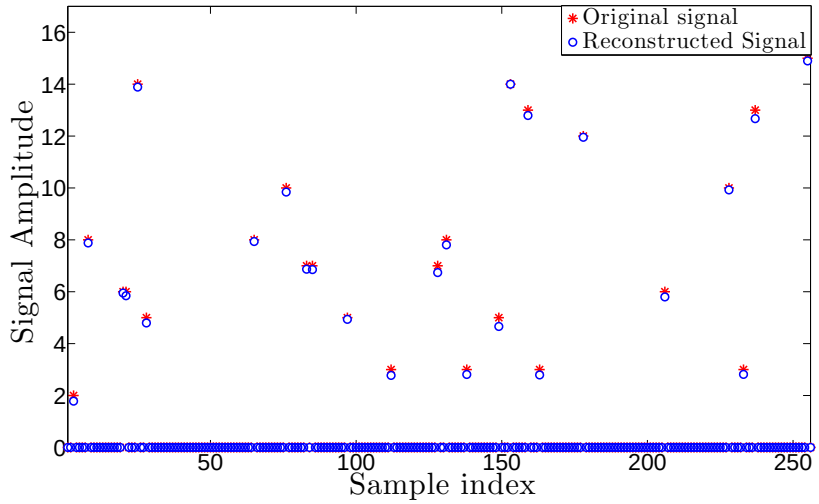


Figure 5.4: The original ($\star$) and the reconstructed random impulses signal (o) with N = 256 samples and has 25 impulses occurring in random indexes. Epigraph of ℓ_1 norm is used. The signal is reconstructed from 30% of measurements

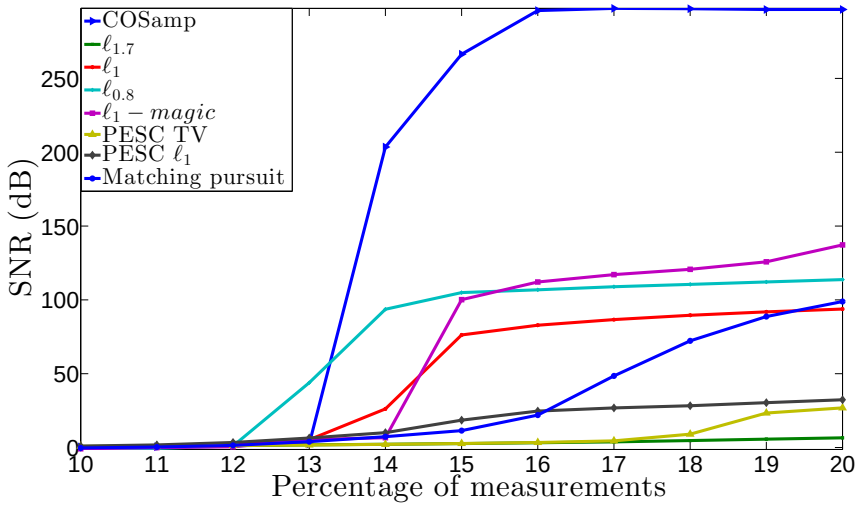


Figure 5.5: The SNR results for reconstruction of `random impulse` signal with different algorithms.

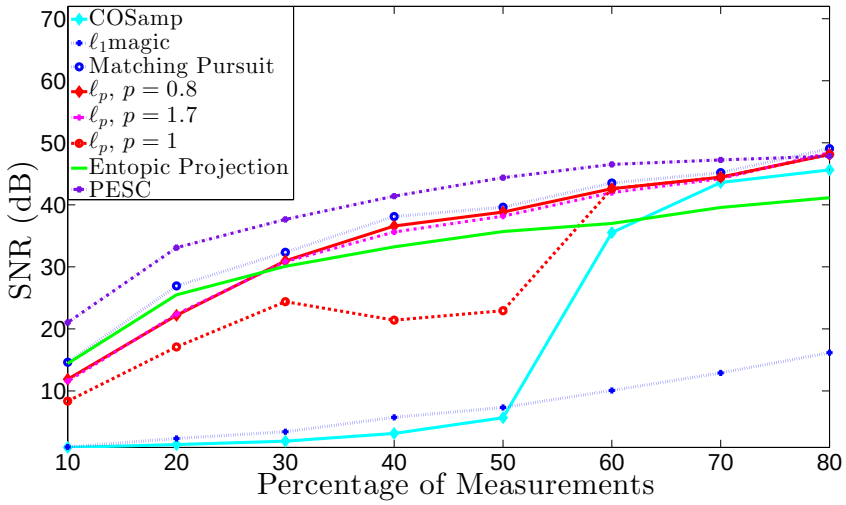


Figure 5.6: The SNR results for reconstruction of `cusp` signal with different algorithms with N = 256 samples. PESC produces the highest SNR curve.

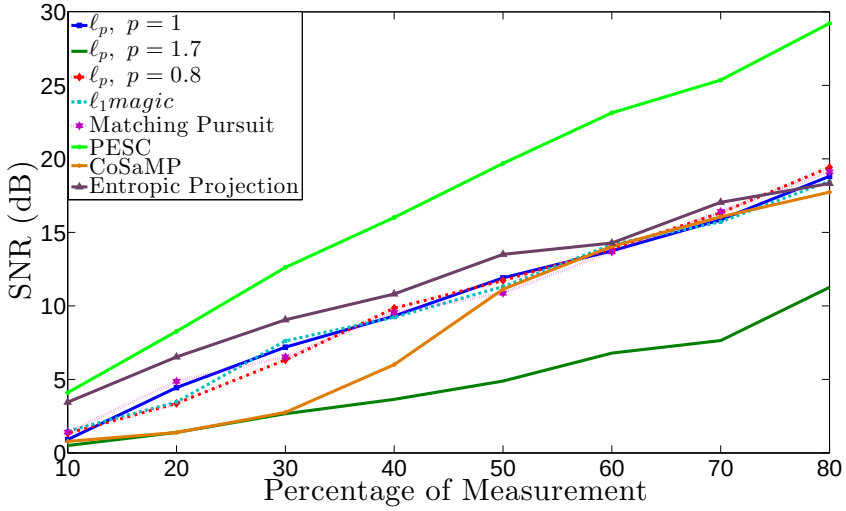
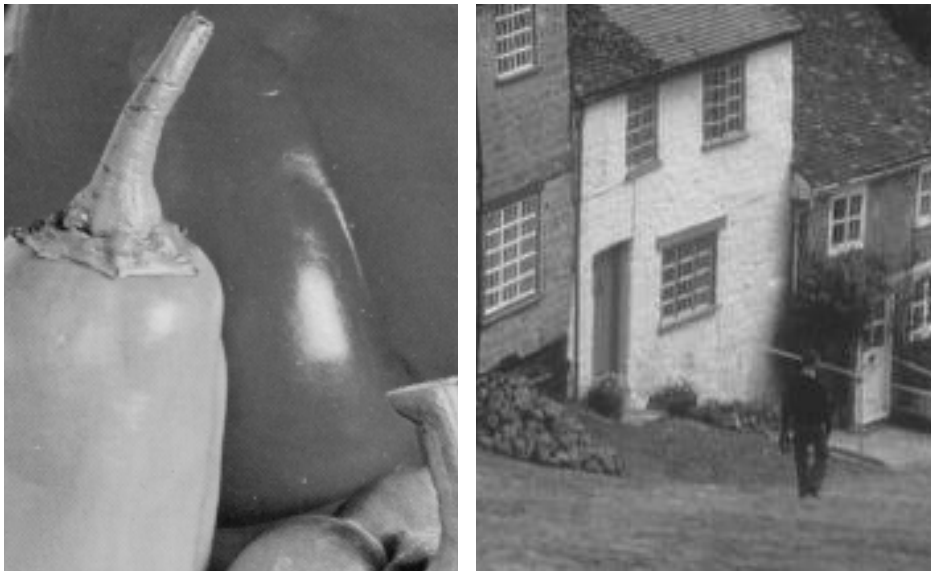


Figure 5.7: The SNR results for reconstruction of `piecewise-smooth` signal with different algorithms with N = 256 samples. PESC produces the best SNR curve among all.



(a) (b)

Figure 5.8: A portion of (a)"peppers" and (b)"goldhill" images.

(a) (b)

Figure 5.9: Results of CS experiments for "peppers" image in the case with 32×32 blocks, and using measurements as much as %30 of the samples by: (a) PESC algorithm; with SNR = 27.06 dB, and (b) Fowler's algorithm; with SNR = 24.66 dB.

(a) (b)

Figure 5.10: Results of CS experiments for "peppers" image in the case with 64×64 blocks, and using measurements as much as %30 of the samples by: (a) PESC algorithm; with SNR = 27.93 dB, and (b) Fowler's algorithm; with SNR = 24.46 dB.

(a) (b)

Figure 5.11: Results of CS experiments for "goldhill" image in the case with 32×32 blocks, and using measurements as much as %30 of the samples by: (a) PESC algorithm; with SNR = 23.64 dB, and (b) Fowler's algorithm; with SNR = 22.78 dB.

(a) (b)

Figure 5.12: Results of CS experiments for "goldhill" image in the case with 64×64 blocks, and using measurements as much as %30 of the samples by: (a) PESC algorithm; with SNR = 24.24 dB, and (b) Fowler's algorithm; with SNR = 23.44 dB.

Chapter 6

Conclusion

In this thesis, new signal reconstruction and restoration methods using the epigraph set of a convex cost function is developed. The reconstructed signal is obtained by making orthogonal projections onto the epigraph set of convex sets representing the desired signal in $\mathbb{R}^{N+1}$. The PESC algorithm based denoising, deconvolution, and compressive sensing algorithms are developed. It is shown that, in all scenarios, PESC approach may not need the optimization of the regularization parameter as in standard TV based signal reconstruction methods.

Two different versions of the PESC algorithm are developed for denoising 1D and 2D signals. For 2D signal denoising, PES-TV algorithm is developed. The PES-TV denoising method is based on the epigraph of the TV function. Epigraph sets of other convex cost functions can be also used in the new denoising approach. The new algorithm does not need the optimization of the regularization parameter as in standard TV denoising methods. Experimental results indicate that better SNR and SSIM results are obtained compared to standard TV based denoising in a large range of images. The proposed method can be incorporated into the so called 3D denoising methods [79]. In 3D denoising methods similar image blocks are grouped and shrinked according to the noise level. Since our method does not need the noise variation, it will lead to more flexible 3D methods.

Moreover, a novel algorithm for denoising images that are corrupted by impulsive noise is presented. This algorithm is a two stage algorithm, which in the first stage the PES-TV based denoising algorithm produces basic denoised estimate for the second stage. Using this basic estimate, the second stage groups the similar blocks of the noisy image and denoise these 3D arrays of the similar blocks using collaborative 3D Wiener filtering. The PES-TV algorithm does not require noise variance to denoise the image, then produces better basic estimate for second stage in comparison with standard BM3D algorithm. Experimental results indicates that higher SNR and PSNR, and better visual results are obtained using the proposed denoising method compared to other algorithms.

For 1D signals, the PES-ℓ_1 algorithm is proposed. It is shown that it is possible to determine denoising soft-threshold using a deterministic approach based on linear algebra and projection onto convex set constructed from the epigraph set of ℓ_1-norm cost function. The main assumption is that the original signal is sparse in wavelet domain or in some transform domain.

Orthogonal projection based denoising is computationally efficient because projection onto a boundary hyperplane of an ℓ_1-ball or the epigraph set can be implemented by performing only one division and $K+1$ additions and/or subtractions, and sign computations. Once the size of the ℓ_1-ball using (3.12) and (3.13) is determined, the orthogonal projection onto an ℓ_1-ball operation is an Order(K) operation. Equations (3.12) and (3.13) only involve multiplications by ± 1. However, it is not possible to incorporate any prior knowledge about the noise probability density function or any other statistical information to the orthogonal projection based denoising method. However, it produces good denoising results under additive white Gaussian noise. Most of the denoising methods available in MATLAB also assumes that the noise is additive, white Gaussian.

In this thesis, new deconvolution and compressive sensing methods based on the epigraph of the TV function are also developed. The TV constraint is imposed to the image in each step of iterative reconstruction algorithm to regularize the estimated image, reduce the convergence time, and enhance the image quality. The simulation results indicate the successful performance of the proposed

algorithms.

The PESC algorithm is also used to solve some other problems as well. In [80], a blind deconvolution algorithm is proposed. Two closed and convex sets for blind deconvolution problem are proposed. Most blurring functions in microscopy are symmetric with respect to the origin. Therefore, they do not modify the phase of the Fourier transform (FT) of the original image. As a result blurred image and the original image have the same FT phase. Therefore, the set of images with a prescribed FT phase can be used as a constraint set in blind deconvolution problems. Another convex set that can be used during the image reconstruction process is the epigraph set of TV function. This set does not need a prescribed upper bound on the total variation of the image. The upper bound is automatically adjusted according to the current image of the restoration process. Both of these two closed and convex sets can be used as a part of any blind deconvolution algorithm.

In [81], a range resolution improvement method based on PESC based deconvolution algorithm is proposed. Here instead of TV constraint, ℓ_1-norm constraint is imposed. One of the main disadvantages of using commercial broadcasts in a Passive Bistatic Radar (PBR) system is the range resolution. Using multiple broadcast channels to improve the radar performance is offered as a solution to this problem. However, it suffers from detection performance due to the side-lobes that matched filter creates for using multiple channels. In this framework, we introduced a deconvolution algorithm to suppress the side-lobes. The two-dimensional matched filter output of a PBR is further analyzed as a deconvolution problem. The deconvolution algorithm is based on making successive projections onto the hyperplanes representing the time delay of a target. Resulting iterative deconvolution algorithm is globally convergent because all constraint sets are closed and convex.

Bibliography

[1] M. Aminghafari, N. Cheze, and J.-M. Poggi, "Multivariate denoising using wavelets and principal component analysis," *Computational Statistics and Data Analysis*, vol. 50, no. 9, pp. 2381 – 2398, 2006.

[2] D. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, pp. 613–627, May 1995.

[3] D. L. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[4] J. Fowler, "The redundant discrete wavelet transform and additive noise," *IEEE Signal Processing Letters*, vol. 12, pp. 629–632, Sept 2005.

[5] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[6] L. Bregman, "Finding the common point of convex sets by the method of successive projection.(russian)," {*USSR*} *Doklady Akademii Nauk SSSR*, vol. 7, no. 3, pp. 200 – 217, 1965.

[7] D. Youla and H. Webb, "Image Restoration by the Method of Convex Projections: Part 1 Num2014;theory," *IEEE Transactions on Medical Imaging*, vol. 1, pp. 81–94, 1982.

[8] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, pp. 259 – 268, 1992.

[9] K. Kose, V. Cevher, and A. Cetin, "Filtered variation method for denoising and sparse signal processing," *IEEE ICASSP*, pp. 3329–3332, 2012.

[10] O. Gunay, K. Kose, B. U. Toreyin, and A. E. Cetin, "Entropy-functional-based online adaptive decision fusion framework with application to wildfire detection in video," *IEEE Transactions on Image Processing*, vol. 21, pp. 2853–2865, 2012.

[11] L. Bregman, "The Relaxation Method of Finding the Common Point of Convex Sets and Its Application to the Solution of Problems in Convex Programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 7, pp. 200 – 217, 1967.

[12] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing," *SIAM Journal on Imaging Sciences*, vol. 1, no. 1, pp. 143–168, 2008.

[13] M. Tofighi, K. Kose, and A. Cetin., "Denoising using projections onto the epigraph set of convex cost functions," in *IEEE International Conference on Image Processing (ICIP'14).*, Oct 2014.

[14] A. E. Cetin, A. Bozkurt, O. Gunay, Y. H. Habiboglu, K. Kose, I. Onaran, R. A. Sevimli, and M. Tofighi, "Projections onto convex sets (pocs) based optimization by lifting," *IEEE GlobalSIP, Austin, Texas, USA*, 2013.

[15] S. Ono, M. Yamagishi, and I. Yamada, "A sparse system identification by using adaptively-weighted total variation via a primal-dual splitting approach," in *IEEE ICASSP*, pp. 6029–6033, 2013.

[16] K. Kose, O. Gunay, and A. E. Cetin, "Compressive sensing using the modified entropy functional," *Digital Signal Processing*, pp. 63 – 70, 2013.

[17] Y. Censor, W. Chen, P. L. Combettes, R. Davidi, and G. T. Herman, "On the Effectiveness of Projection Methods for Convex Feasibility Problems with Linear Inequality Constraints," *Computational Optimization and Applications*, vol. 51, pp. 1065–1088, 2012.

[18] K. Slavakis, S. Theodoridis, and I. Yamada, "Online Kernel-Based Classification Using Adaptive Projection Algorithms," *IEEE Transactions on Signal Processing*, vol. 56, pp. 2781–2796, 2008.

[19] A. E. Cetin, "Reconstruction of signals from fourier transform samples," *Signal Processing*, pp. 129–148, 1989.

[20] K. Kose and A. E. Cetin, "Low-pass filtering of irregularly sampled signals using a set theoretic framework," *IEEE Signal Processing Magazine*, pp. 117–121, 2011.

[21] Y. Censor and A. Lent, "An Iterative Row-Action Method for Interval Convex Programming," *Journal of Optimization Theory and Applications*, vol. 34, pp. 321–353, 1981.

[22] S. Konstantinos, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in reproducing kernel hilbert spaces: the robust beamforming case," *IEEE Transactions on Signal Processing*, vol. 57, pp. 4744–4764, 2009.

[23] K. S. Theodoridis and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97–123, 2011.

[24] Y. Censor and A. Lent, "Optimization of "$\log x$" entropy over linear equality constraints," *SIAM Journal on Control and Optimization*, vol. 25, no. 4, pp. 921–933, 1987.

[25] H. J. Trussell and M. R. Civanlar, "The Landweber Iteration and Projection Onto Convex Set," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, no. 6, pp. 1632–1634, 1985.

[26] P. L. Combettes and J.-C. Pesquet, "Image restoration subject to a total variation constraint," *IEEE Transactions on Image Processing*, vol. 13, pp. 1213–1222, 2004.

[27] P. L. Combettes, "The foundations of set theoretic estimation," *Proceedings of the IEEE*, vol. 81, pp. 182 –208, 1993.

[28] I. Yamada, M. Yukawa, and M. Yamagishi, "Minimizing the moreau envelope of nonsmooth convex functions over the fixed point set of certain quasi-nonexpansive mappings," *Springer NY*, pp. 345–390, 2011.

[29] Y. Censor and G. T. Herman, "On some optimization techniques in image reconstruction from projections," *Applied Numerical Mathematics*, vol. 3, no. 5, pp. 365–391, 1987.

[30] I. Sezan and H. Stark, "Image restoration by the method of convex projections: Part 2-applications and numerical results," *IEEE Transactions on Medical Imaging*, vol. 1, pp. 95–101, 1982.

[31] Y. Censor and S. A. Zenios, "Proximal minimization algorithm withd-functions," *Journal of Optimization Theory and Applications*, vol. 73, pp. 451–464, 1992.

[32] A. Lent and H. Tuy, "An Iterative Method for the Extrapolation of Band-Limited Functions," *Journal of Optimization Theory and Applications*, vol. 83, pp. 554–565, 1981.

[33] Y. Censor, "Row-action methods for huge and sparse systems and their applications," *SIAM review*, vol. 23, pp. 444–466, 1981.

[34] Y. Censor, A. R. De Pierro, and A. N. Iusem, "Optimization of burg's entropy over linear constraints," *Applied Numerical Mathematics*, vol. 7, no. 2, pp. 151–165, 1991.

[35] M. Rossi, A. M. Haimovich, and Y. C. Eldar, "Conditions for Target Recovery in Spatial Compressive Sensing for MIMO Radar," *IEEE ICASSP*, 2013.

[36] L. Gubin, B. Polyak, and E. Raik, "The Method of Projections for Finding the Common Point of Convex Sets," *Computational Mathematics and Mathematical Physics*, vol. 7, pp. 1 – 24, 1967.

[37] A. E. Çetin, O. Gerek, and Y. Yardimci, "Equiripple FIR Filter Design by the FFT Algorithm," *IEEE Signal Processing Magazine*, vol. 14, no. 2, pp. 60–64, 1997.

[38] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical Imaging and Vision*, vol. 20, pp. 89–97, Jan. 2004.

[39] R. Baraniuk, "Compressive sensing [lecture notes]," *IEEE Signal Processing Magazine*, vol. 24, pp. 118–121, 2007.

[40] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," Springer Optimization and Its Applications, pp. 185–212, Springer NY, 2011.

[41] G. Chierchia, N. Pustelnik, J.-C. Pesquet, and B. Pesquet-Popescu, "Epigraphical projection and proximal tools for solving constrained convex optimization problems: Part i," *CoRR*, vol. abs/1210.5844, 2012.

[42] G. Chierchia, N. Pustelnik, J.-C. Pesquet, and B. Pesquet-Popescu, "An epigraphical convex optimization approach for multicomponent image restoration using non-local structure tensor," in *IEEE ICASSP, 2013*, pp. 1359–1363, 2013.

[43] M. Tofighi, K. Kose, and A. Enis Cetin, "Signal Reconstruction Framework Based On Projections Onto Epigraph Set Of A Convex Cost Function (PESC)," *ArXiv e-prints*, Feb. 2014.

[44] G. Chierchia, N. Pustelnik, J.-C. Pesquet, and B. Pesquet-Popescu, "Epigraphical projection and proximal tools for solving constrained convex optimization problems," *Signal, Image and Video Processing*, pp. 1–13, 2014.

[45] A. E. Cetin and M. Tofighi, "Projection-based wavelet denoising," *IEEE Signal Processing Magazine*, vol. 32, September 2015.

[46] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, pp. 2080–2095, Aug 2007.

[47] S. Chang, B. Yu, and M. Vetterli, "Adaptive wavelet thresholding for image denoising and compression," *IEEE Transactions on Image Processing*, vol. 9, pp. 1532–1546, Sep 2000.

[48] F. Luisier, T. Blu, and M. Unser, "A new sure approach to image denoising: Interscale orthonormal wavelet thresholding," *IEEE Transactions on Image Processing*, vol. 16, pp. 593–606, March 2007.

[49] C. Vonesch and M. Unser, "A fast multilevel algorithm for wavelet-regularized image restoration," *IEEE Transactions on Image Processing*, vol. 18, pp. 509–523, March 2009.

[50] C. Vonesch and M. Unser, "A fast thresholded landweber algorithm for wavelet-regularized multidimensional deconvolution," *IEEE Transactions on Image Processing*, vol. 17, pp. 539–549, April 2008.

[51] S. Mallat and W.-L. Hwang, "Singularity detection and processing with wavelets," *IEEE Transactions on Information Theory*, vol. 38, pp. 617–643, Mar 1992.

[52] D. Needell and J. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301 – 321, 2009.

[53] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3869–3872, 2008.

[54] I. J. B. Efron, T. Hastie and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[55] A. Bovik, T. Huang, and J. Munson, D.C., "A generalization of median filtering using linear combinations of order statistics," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 31, pp. 1342–1350, Dec 1983.

[56] C. A. Micchelli, L. Shen, and Y. Xu, "Proximity algorithms for image models: denoising," *Inverse Problems*, vol. 27, no. 4, p. 045009, 2011.

[57] A. E. Cetin and A. Tekalp, "Robust reduced update kalman filtering," *IEEE Transactions on Circuits and Systems*, vol. 37, pp. 155–156, Jan 1990.

[58] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Processing Magazine*, vol. 28, pp. 97–123, Jan 2011.

[59] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, April 2004.

[60] K. lossless true color image suite, "http://r0k.us/graphics/kodak/," 2013.

[61] P. Windyga, "Fast impulsive noise removal," *IEEE Transactions on Image Processing*, vol. 10, pp. 173–179, Jan 2001.

[62] Y.-H. Lee and S. Kassam, "Generalized median filtering and related nonlinear filtering techniques," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, pp. 672–683, Jun 1985.

[63] Kodak lossless true color image suite. `http://r0k.us/graphics/kodak/`.

[64] R. Chan, C.-W. Ho, and M. Nikolova, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," *IEEE Transactions on Image Processing*, vol. 14, pp. 1479–1485, Oct 2005.

[65] S. Peterson, Y. H. Lee, and S. Kassam, "Some statistical properties of alpha-trimmed mean and standard type m filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, pp. 707–713, May 1988.

[66] G. Chierchia, N. Pustelnik, J.-C. Pesquet, and B. Pesquet-Popescu, "Epigraphical projection and proximal tools for solving constrained convex optimization problems," *Signal, Image and Video Processing*, pp. 1–13, 2014.

[67] K. Slavakis, S.-J. Kim, G. Mateos, and G. Giannakis, "Stochastic approximation vis-a-vis online learning for big data analytics [lecture notes]," *IEEE Signal Processing Magazine*, vol. 31, pp. 124–129, Nov 2014.

[68] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l1-ball for learning in high dimensions," in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, (New York, NY, USA), pp. 272–279, ACM, 2008.

[69] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[70] E. J. Candès, "Compressive sampling," in *Proceedings on the International Congress of Mathematicians*, pp. 1433–1452, 2006.

[71] E. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[72] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[73] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[74] R. Chartrand, "Exact reconstruction of sparse signals via nonconvex minimization," *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707–710, 2007.

[75] T. Rezaii, M. Tinati, and S. Beheshti, "Sparsity aware consistent and high precision variable selection," *Signal, Image and Video Processing*, pp. 1–12, 2012.

[76] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP),*, pp. 3734–3737, 2010.

[77] N. Pustelnik, C. Chaux, and J. Pesquet, "Parallel proximal algorithm for image restoration using hybrid regularization," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2450–2462, 2011.

[78] J. E. Fowler, S. Mun, and E. W. Tramel, "Block-based compressed sensing of images and video," *Foundations and Trends in Signal Processing*, vol. 4, no. 4, pp. 297–416, 2012.

[79] A. Danielyan, V. Katkovnik, and K. Egiazarian, "Bm3d frames and variational image deblurring," *IEEE Transactions on Image Processing*, vol. 21, pp. 1715–1728, April 2012.

[80] M. Tofighi, O. Yorulmaz, and A. Enis Cetin, "Phase and TV Based Convex Sets for Blind Deconvolution of Microscopic Images," *ArXiv e-prints*, Mar. 2015.

[81] M. T. Arslan, M. Tofighi, R. A. Sevimli, and A. E. Çetin, "Range resolution improvement in passive bistatic radars using nested fm channels and least squares approach," in *Proc. SPIE*, vol. 9474, pp. 947414–1–947414–9, 2015.