Overcomplete Dictionary and Deep Learning Approaches to Image and Video Analysis.

Kha Gia Quach

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Computer Science) at

Concordia University

Montréal, Québec, Canada

November 2017

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Kha Gia Quach**

Entitled: **Overcomplete Dictionary and Deep Learning Approaches to Image and Video Analysis.**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Jia Yuan Yu

_____ External Examiner
Dr. Mohamed Cheriet

_____ Examiner
Dr. Brigitte Jaumard

_____ Examiner
Dr. Eusebius J. Doedel

_____ Examiner
Dr. Ben Hamza

_____ Supervisor
Dr. Tien D. Bui

_____ Co-supervisor
Dr. Khoa Luu

Approved by _____
Dr. Sudhir Mudur, Chair
Department of Computer Science and Software Engineering

November 08, 2017 _____
Dr. Amir Asif, Dean
Faculty of Engineering and Computer Science

Abstract

Overcomplete Dictionary and Deep Learning Approaches to Image and Video Analysis.

Kha Gia Quach, Ph.D.

Concordia University, 2017

Extracting useful information while ignoring others (e.g. noise, occlusion, lighting) is an essential and challenging data analyzing step for many computer vision tasks such as facial recognition, scene reconstruction, event detection, image restoration, etc. Data analyzing of those tasks can be formulated as a form of matrix decomposition or factorization to separate useful and/or fill in missing information based on sparsity and/or low-rankness of the data. There has been an increasing number of non-convex approaches including conventional matrix norm optimizing and emerging deep learning models. However, it is hard to optimize the ideal ℓ_0-norm or learn the deep models directly and efficiently. Motivated from this challenging process, this thesis proposes two sets of approaches: conventional and deep learning based.

For conventional approaches, this thesis proposes a novel online non-convex ℓ_p-norm based Robust PCA (OLP-RPCA) approach for matrix decomposition, where $0 < p < 1$. OLP-RPCA is developed from the offline version LP-RPCA. A robust face recognition framework is also developed from Robust PCA and sparse coding approaches. More importantly, OLP-RPCA method can achieve real-time performance on large-scale data without parallelizing or implementing on a graphics processing unit. We mathematically and empirically show that our OLP-RPCA algorithm is linear in both the sample dimension and the number of samples. The proposed OLP-RPCA and LP-RPCA approaches are evaluated in various applications including Gaussian/non-Gaussian image denoising, face modeling, real-time background subtraction and video inpainting and compared against numerous state-of-the-art methods to demonstrate the robustness of the algorithms.

In addition, this thesis proposes a novel Robust ℓ_p-norm Singular Value Decomposition (RP-SVD) method for analyzing two-way functional data. The proposed RP-SVD is formulated as an ℓ_p-norm based penalized loss minimization problem. The proposed RP-SVD method is evaluated in four applications, i.e. noise and outlier removal, estimation of missing values, structure from motion reconstruction and facial image reconstruction.

For deep learning based approaches, this thesis explores the idea of matrix decomposition via Robust Deep Boltzmann Machines (RDBM), an alternative form of Robust Boltzmann Machines, which aiming at dealing with noise and occlusion for face-related applications, particularly. This thesis proposes an extension to texture modeling in the Deep Appearance Models (DAMs) by using RDBM to enhance its robustness against noise and occlusion. The extended model can cope with occlusion and extreme poses when modeling human faces in 2D image reconstruction. This thesis also introduces new fitting algorithms with occlusion awareness through the mask obtained from the RDBM reconstruction. The proposed approach is evaluated in various applications by using challenging face datasets, i.e. Labeled Face Parts in the Wild (LFPW), Helen, EURECOM and AR databases, to demonstrate its robustness and capabilities.

Acknowledgments

I would like to express my gratitude to all those who have supported, influenced and helped me in the process which ultimately resulted in this thesis.

First of all, I would like to thank Prof. Tien D. Bui and Dr. Khoa Luu for their support and encouragement. This dissertation could not be completed without their trust and constructive suggestions. I am also greatful to my committee members, Dr. Brigitte Jaumard, Dr. Eusebius J. Doedel and Dr. Ben Hamza for their valuable comments and suggestions. Furthermore, I would like to thank Prof. Marios Savvides for the opportunity to visit his lab and the great collaboration. Next, I would like to thank all my colleagues at Concordia University for their fruitful comments and discussions during my research work and this dissertation. I also thank my colleagues, Chi Nhan Duong and Ngan Le for the great and valuable discussions and teamwork. I would like to express my appreciation for Halina Monkiewicz, Tina Yankovich, Brittany Frost for their administrative supports. I also want to thank the colleagues at Cylab, Biometrics Center, CMU for their valuable discussions during my stay, especially Sekhar Bhagavatula and Chenchen Zhu for the excellent collaboration. I am very grateful to all other friends who helped me out with all the challenges living in Canada and US and always supported me.

My very special thanks go to my family. My father, my mother, and my wife, Quynh, always support me during and beyond my thesis. Without them, I could not go through the difficult time during my study. Therefore, I would like to dedicate my thesis to my family.

Contributions of Authors

This section listed all the papers that I previously published or submitted in international conferences and journals (including co-author papers) since I started my Ph.D. program in Concordia University.

Journal Papers

- **Kha Gia Quach**, Chi Nhan Duong, Khoa Luu, Tien D. Bui. **Non-convex Online Robust PCA: Enhance Sparsity via ℓ_p-norm Minimization**. *Computer Vision and Image Understanding* **(CVIU)**, Vol. 158, Pages 126–140, May 2017. (*Impact factor: 2.498*)

- Chi Nhan Duong, Khoa Luu, **Kha Gia Quach**, Tien D. Bui. **Deep Appearance Models: A Deep Boltzmann Machine Approach for Face Modeling**. *International Journal of Computer Vision* **(IJCV)**, 2016. **(Under review - 2nd round)** (*Impact factor: 4.27*).

- Guangyi Chen, Tien D. Bui, **Kha Gia Quach**, Shen-En Qian. **Denoising Hyperspectral Imagery Using Principal Component Analysis and Block-Matching 4D Filtering**. *Canadian Journal of Remote Sensing* **(CJRS)**, Vol. 40, No. 1, 2014. (*Impact factor: 1.95*)

International Conferences

- Chi Nhan Duong, **Kha Gia Quach**, Khoa Luu, T. Hoang Ngan Le, Marios Savvides. **Temporal Non-Volume Preserving Approach to Facial Age-Progression and Age-Invariant Face Recognition**. *The IEEE International Conference on Computer Vision* **(ICCV)**, October 2017.

- **Kha Gia Quach**, Chi Nhan Duong, Khoa Luu, Tien D. Bui. **Depth-based 3D Hand Pose Tracking**. *The 23rd International Conference on Pattern Recognition* **(ICPR)**, December 2016, pp. 2746–2751.

- **Kha Gia Quach**, Chi Nhan Duong, Khoa Luu, Tien D. Bui. **Robust Deep Appearance Models**. *The 23rd International Conference on Pattern Recognition* **(ICPR)**, December 2016, pp. 390–395.

- Chi Nhan Duong, Khoa Luu, **Kha Gia Quach**, Tien D. Bui. **Longitudinal Face Modeling via Temporal Deep Restricted Boltzmann Machines**. *The IEEE Conference on Computer Vision and Pattern Recognition* **(CVPR)**, Las Vegas, June 2016, pp. 5272–5780.

- **Kha Gia Quach**, Khoa Luu, Chi Nhan Duong, Tien D. Bui. **Robust ℓ_p-norm Singular Value Decomposition**. *NIPS Workshop on Non-convex Optimization for Machine Learning: Theory and Practice* **(NIPSW)**, December 2015.

- Chi Nhan Duong, Khoa Luu, **Kha Gia Quach**, Tien D. Bui. **Beyond Principal Components: Deep Boltzmann Machines for Face Modeling**. *The IEEE Conference on Computer Vision and Pattern Recognition* **(CVPR)**, Boston, June 2015, pp. 1786–1794.

- **Kha Gia Quach**, Chi Nhan Duong, Tien D. Bui. **Sparse Representation and Low-rank Approximation for Robust Face Recognition**. *the 22nd International Conference on Pattern Recognition* **(ICPR)**, Sweden, August 2014, pp. 1330–1335.

- Chi Nhan Duong, **Kha Gia Quach**, Tien D. Bui. **Are Sparse Representation and Dictionary Learning Good for Handwritten Character Recognition?** *the 14th International Conference on Frontiers in Handwriting Recognition* **(ICFHR)**, Greece, September 2014, pp. 575–580.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Digital revolution opens a new era of digital data analysis that urges the need of efficient methods for analyzing and recovering information in large-scale datasets. Sparsity and low-rankness are two popular properties being exploited to analyze and recover data for numerous applications in signal processing, telecommunications, computer vision and machine learning areas. There are many algorithms exploiting sparsity and low-rank properties of data or signals to efficiently recover them from very few measurements [104]. Recently, matrix optimization problems, e.g. matrix decomposition, matrix factorization, matrix completion, etc., have been using sparsity-based optimization techniques developed for compressive sensing. More recently, the emerging deep learning techniques have been developed to extract robust features from input data containing certain noise and occlusion.

1.1 Challenges in Image and Video Analysis

This thesis considers two main tasks of Image and Video Analysis but not limited to, *image denoising* and *video background subtraction*. Extracting and recovering information of user interest from Image and Video poses numerous challenges. An image or a video may contain a variety of objects, some of which may be of interest to users, while others may not be. For *image denoising*, this thesis works on two types of images, pattern images and natural scene images. Pattern/texture usually forms a certain kind of repeating structures that help to fill in missing regions/pixels in the

images. Meanwhile, natural scene images are more complex and contain much background clutter and/or having many unrelated contents, they even have low-resolution for web-based images. This problem makes the pre-processing task become more difficult. For *video background subtraction*, handling videos with large number of frames and processing videos in real-time are the main issues. In addition, separating multiple foreground layers from the complex background layer makes this task more challenging.

1.2 Research Hypothesis and the Goal of the Thesis

In the problem of matrix decomposition (additive matrix decomposition), given a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, it can be decomposed into two components, i.e. $\mathbf{L}, \mathbf{S} \in \mathbb{R}^{m \times n}$, where $\mathbf{L}$ is the low-rank matrix and $\mathbf{S}$ is the sparse component. This problem, also known as *Robust Principal Component Analysis* (**RPCA**) [17], [21] can be mathematically formulated as in Eqn. (1).

$$\min_{\mathbf{L},\mathbf{S}} \ \text{rank}(\mathbf{L}) + \lambda \|\mathbf{S}\|_0 \ \text{ s.t. } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{1}$$

where $\|\mathbf{S}\|_0$ computes the number of nonzero entries in matrix $\mathbf{S}$ and the parameter $\lambda > 0$ controls the trade-off between the sparsity level and reconstruction fidelity.

Solving Eqn. (1) is difficult since it poses as a challenging NP-hard problem. Candès et al. [17] presented the *Principal Component Pursuit* (**PCP**) method to solve Eqn. (1) using a tractable and *convex approximation* to the objective function. In their method, the non-convex ℓ_0-norm and the rank functions are approximated by a convex relaxation ℓ_1-norm and a nuclear norm respectively as shown in Eqn. (2).

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1 \ \text{ s.t. } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{2}$$

where $\|\cdot\|_*$ denotes the nuclear norm, i.e. the sum of singular values of a matrix and $\|\cdot\|_1$ denotes the ℓ_1-norm, i.e. the sum of the absolute values of the matrix entries. In the last few years, two main aspects in the RPCA literature have drawn huge attention: efficient incremental or online algorithms (scalability) and non-convex surrogate for ℓ_0-norm (non-convexity).

Although there are numerous extensions of the ℓ_1-norm PCP approach [6, 82, 117, 142], a mathematically critical difference still exists between the ℓ_0-norm and the ℓ_1-norm problems. The ℓ_0-norm treats all nonzero coefficients in the same way while the ℓ_1-norm highly depends upon the

magnitude of matrix elements. Thus, the solutions found in the ℓ_1-norm approximation are usually not optimal with respect to the corresponding ℓ_0-norm problem. Some authors [16, 27, 108] suggest that ℓ_p-norm $(0 < p < 1)$ gives a better approximation to ℓ_0-norm than ℓ_1-norm. Although some current ℓ_p-norm methods have been successfully presented (as shown in Table 1.1), there is no non-convex algorithm deriving explicitly from the non-convex regularization: the ℓ_p-norm $(p < 1)$ and the ℓ_p-Schatten-norm $(p < 1)$ used for the sparse and the low-rank matrix, respectively.

Furthermore, incremental algorithms, e.g. ReProCS [105], [106], [56] and OR-PCA [41], are preferable to batch algorithms in some applications (e.g. video surveillance) due to the nature of data generation and processing. However, most of the online algorithms are not fast enough to analyze new coming large-scale data in real-time. Real-time implementation was made possible for those algorithms thanks to the parallel processing power of a graphics processing unit (GPU) but not due to an actual reduction of their complexities. Currently, there are only a few algorithms that can handle both incremental and real-time processing (as shown in Table 1.1). Therefore, we propose a novel real-time incremental ℓ_p-norm approach to solve Eqn. (1) efficiently. Our proposed approach can *simultaneously* compute the ℓ_p-norm sparsity and provide an efficient online framework.

In addition, matrix decomposition techniques can separate unwanted information from the input signals, particularly for facial images, we can decompose an occluded face image into occluded regions and non-occluded face. However, it may not preserve the identity of the face well, since conventional additive matrix decomposition only considers non-structural data in general. On the other hand, generative models, e.g. Active Appearance Models [31], Deep Appearance Models [100], etc., are commonly used to recover and extract features from signals, especially facial images, but it may include noise or other unwanted information, i.e. occlusions and pose in face modeling. Similar to conventional matrix decomposition introduced above, Robust Boltzmann Machines [119] handle noise and occlusion using a mixture of two Gaussians: real-value and noise models.

$$E_{RoBM} = E_{GRBM} + E_{RBM_{mask}} + E_{Noise} \tag{3}$$

With this approach, we can model structural data, especially for facial images. The second part of this thesis proposes to build a robust generative model that can separate unwanted factors as well as

Table 1.1: Comparing the properties between our proposed online ℓ_p-norm RPCA (OLP-RPCA) approach and other state-of-the-art low-rank minimization and RPCA methods, where ✗ represents unknown or not directly applicable properties.

	OLP-RPCA	RPCA [17]	NC-ADMM [24]	NRPCA [98]	IRNN [86]	IRLS [85]	pROST [57]	MOG-RPCA [92]	Re-ProCS [106]	OR-PCA [41]
Non-convexity										
Sparse matrix	✓	✗	✓	✓	✗	✗	✓	✗	✗	✗
Low-rank matrix	✓	✗	✓	✓	✓	✓	✗	✓	✗	✗
Scalability										
Online	✓	✗	✗	✗	✗	✗	✓	✗	✓	✓
Real-time	✓(CPU)	✗	✗	✗	✗	✗	✓(GPU)	✗	✗	✓

recover missing regions while preserving identity information.

1.3 Overview of the Thesis

This thesis contains two main parts: 1 – conventional approaches in matrix decomposition and factorization; 2 – deep learning approach. The first part presents a highly efficient online version of non-convex Robust Principal Component Analysis (**OLP-RPCA**) for solving the problem in Eqn. (1) approximately by using ℓ_p-norm. This online approach is developed from our derivation of the non-convex objective function of RPCA problem (**LP-RPCA**). The Alternating Direction Method of Multipliers (ADMM) is employed to find appropriate solutions to this problem. The second part introduces an extension of Deep Appearance Models (DAM) [100], a generative model based on Restricted Boltzmann Machines (RBM) [59], which incorporate Robust Deep Boltzmann Machines (RDBM) to enhance the robustness of DAM to occlusion and extreme poses.

This thesis is organized as follows. In *Chapter 2*, it briefly introduces sparse coding, dictionary learning and the ℓ_p-regularized problems. Then it provides an overview of matrix decomposition and factorization approaches such as RPCA and SVD. Lastly, it reviews some basic ideas on recent deep learning models such as Boltzmann Machines and its extended models. In *Chapter 3*, it presents some ideas about solving the ℓ_p-norm based RPCA problem approximately and the solution for ℓ_p-norm based SVD problem is formulated and analyzed. In addition, it introduces a robust face recognition framework using both RPCA and dictionary learning approaches. In *Chapter 4*, it introduces Robust Deep Appearance Models (**RDAM**) that can be used for eliminating occlusion and

recovering pose in face modeling. In *Chapter 5*, some results of the ℓ_p-norm based RPCA and SVD problems are presented and analyzed. In addition, we evaluate the performance of our proposed framework RDAM in face modeling tasks using data in the wild and demonstrate its robustness in model fitting steps. Some results of robust face recognition using low-rank and sparse representation are also presented in Chapter 5. Finally, *Chapter 6* summarizes methodologies, contributions, and results. Further possible work and challenges are discussed in Chapter 6.

Chapter 2

Background and Literature Review

This chapter presents some backgrounds on sparse coding and dictionary learning problems. Some literature review on common approaches for matrix decomposition and factorization: Robust Principal Component Analysis (RPCA) and Singular Value Decomposition (SVD) are also introduced. In addition, a section on deep learning topic focusing on Boltzmann Machines will be presented in this chapter.

2.1 From Sparse Coding to Overcomplete Dictionaries

In this section, compressive sensing is first introduced and then the ℓ_p-norm is defined and its properties and optimization methods are analyzed. Next, other penalty functions are presented as well. Finally, sparse representation and dictionary learning are briefly introduced.

2.1.1 Compressive Sensing

In the standard Compressive Sensing (CS) model, the core thing is to recover a signal $\mathbf{x}$ from its observations $\mathbf{y}$ and the measurement matrix $\mathbf{\Phi}$ defined as $\mathbf{y} = \mathbf{\Phi x} + \epsilon$, where $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$ is the observation vector, $\Phi \in \mathbb{R}^{m \times n}$ is a measurement matrix, $\epsilon \in \mathbb{R}^m$ is a random noise vector and $m \ll n$. This seemingly ill-posed problem, i.e. underdetermined linear systems with an infinite number of solutions, can be solved reliably and efficiently by adding the constraint that the initial signal $\mathbf{x}$ is sparse. The sparsity of the signal is measured in terms of the ℓ_0-norm $\|\mathbf{x}\|_0 :=$

card $\{j : x_j \neq 0\}$. Then the sparse coding problem (Section 2.1.3) solves the following non-convex optimization problem:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ subject to } \Phi\mathbf{x} = \mathbf{y} \tag{4}$$

Unfortunately, the problem in Eqn. (4) is a NP-hard problem and it is computationally infeasible to solve the problem in large-scale [97]. A common approach is to relax this non-convex problem into a convex one using ℓ_1-norm. Then the desired signal $\mathbf{x}$ is found using the convex optimization problem in Eqn. (5).

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ subject to } \Phi\mathbf{x} = \mathbf{y} \tag{5}$$

Theoretical understanding of the conditions has been well-established for the ℓ_1-relaxation to produce good and equivalent solutions to the ℓ_0-minimization in (4) with high probability [15]. However, there is a mathematically critical difference between the ℓ_0-norm and the ℓ_1-norm regularized problems. While the ℓ_0-norm treats all nonzero coefficients in the same way, the ℓ_1-norm highly depends upon the absolute magnitude of elements. The solutions found in the ℓ_1-norm approximation are usually sub-optimal with respect to the corresponding ℓ_0-norm problem [9] [37] [91]. Thus, an alternative form to bridge the gap between ℓ_0 and ℓ_1-norm, which is ℓ_p-norm $(0 < p < 1)$ has been proposed.

$$\min_{\mathbf{x}} \|\mathbf{x}\|_p \text{ subject to } \Phi\mathbf{x} = \mathbf{y} \tag{6}$$

Although the ℓ_p-norm retains the nature of the overall optimization problem as being non-convex, numerous empirical experiments and theoretical analysis have shown that one can achieve better solutions (i.e. sparser) with ℓ_p-norm. The remaining question is how this type of non-convex relaxation helps solving other sparsity-related problems with large-scale data. Therefore, this thesis aims at designing an iteratively procedure to efficiently solve ℓ_p-norm $(0 < p < 1)$ based regularization for two well-known problems: matrix decomposition and matrix factorization.

2.1.2 ℓ_p-norm $(0 < p < 1)$ Regularization

Definition and Properties

Given a vector $\mathbf{x} \in \mathbb{R}^n$, a general definition of ℓ_p-norm of $\mathbf{x}$ is given as

$$\|\mathbf{x}\|_p = (|\mathbf{x}_1|^p + |\mathbf{x}_2|^p + \cdots + |\ \mathbf{x}_n|^p)^{1/p} \tag{7}$$

We consider the unit balls in $\mathbb{R}^2$ as illustrated in Fig. 2.1 to show the properties of the ℓ_p-norm with the values $0 < p < \infty$. When $p \geq 1$, it is a norm with the properties of a "length function" (or a norm) which is a convex function and holds the triangle inequality. The particular cases of $p = 1, 2$ and ∞ are widely used in many optimization procedures. This kind of norm regularization often gives a non-sparse solution, except for $p = 1$ (ℓ_1-norm) which yields sparse results in certain conditions.

When $0 < p < 1$, it is only a quasi-norm [103], it does not satisfy the triangle inequality (the inequality is actually reversed) but it induces a metric which is "concave". The resulting optimization problem involving ℓ_p-norm will be non-convex that is intractable, since it contains many strong local minima. As $p \to 0$, the solutions become more sparse, however, larger values of p give smooth (or less sparse) solutions. Fig. 2.1 illustrates the reason for this. The curves in Fig. 2.1 approach the $\mathbf{x}_1$, $\mathbf{x}_2$ axes as $p \to 0$.

We notice that,

$$\lim_{p \to 0} |\mathbf{x}_i|^p = \begin{cases} 1, & \text{for } \mathbf{x}_i \neq 0 \\ 0, & \text{for } \mathbf{x}_i = 0 \end{cases} \tag{8}$$

Note that for $p < 1$, we consider here $\|\cdot\|_p^p$ rather than $\|\cdot\|_p$ so that the above limit exists when $p \to 0$. This suggests that, by defining $0^0 = 0$, the zero-"norm" or ℓ_0-"norm" (using the term norm here is an abuse of terminology, as $\|\cdot\|_0$ does not satisfy all of the properties of a norm) of $\mathbf{x}$ is equal to

$$|\mathbf{x}_1|^0 + |\mathbf{x}_2|^0 + \cdots + |\mathbf{x}_n|^0 \tag{9}$$

which is a special case of the generalized ℓ_p-norm. It provides a way to count the number of non-zero entries in a vector $\mathbf{x}$.

Some variations appear in the naming of the norm: L_p-norm [45], ℓ_p-norm [70], ℓ_q optimization [88] or p-norm [108], but those terms are referring to the same thing. To be consistent with the existing ℓ_0, ℓ_1 and ℓ_2-norm terms, we use the term ℓ_p-norm throughout this thesis to refer ℓ_p-norm in the case of $0 < p < 1$.

Solving ℓ_0-norm minimization is a NP hard problem [97] which means that it cannot be solved by any tractable algorithm in polynomial time (or in practice). Some works [15] showed that ℓ_0-norm can be replaced by its nearest convex lower bound, the ℓ_1-norm, to obtain sparse results. We consider ℓ_p minimization as a strategy lying between two extremes, the ℓ_0 and ℓ_1 minimization. One extreme is impractical to solve but gives the optimal sparse solution, the latter can be solved efficiently, but does not guarantee optimal solutions. Meanwhile, ℓ_p minimization has some benefits, firstly, ℓ_p-norm approximates ℓ_0-norm better and yields more sparse results. Secondly, solving ℓ_p minimization is as efficient as its convex vis-à-vis ℓ_1-norm.

ℓ_p-norm minimization has been used in different fields. It was first proposed in [77] to maximize sparseness of arrays. Leahy and Jeffs [77] used an ad hoc simplex search algorithm, but it can only converge to a local minimum. Bradley and Mangasarian [9] proposed an ℓ_p approximation method called Feature Selection ConcaVe (FSV), which is used for feature/variable selection in machine learning for the first time. Knight and Fu later presented some theoretical results in [72] supporting the use of this ℓ_p-norm (also known as bridge estimators) for variable selection. Recently, other works in compressive sensing and sparse approximation drew the attention back to this ℓ_p-norm [22] [112]. Since then, many studies have provided some theoretical background guaranteeing the use of ℓ_p-norm minimization in compressive sensing. We will discuss its theoretical development in the next section.

Theoretical Analysis

In this section, we will briefly introduce all theoretical studies on ℓ_p-norm properties and benefits when applied particularly in compressive sensing problems.

Chartrand [22] showed that using the ℓ_p-norm can give exact reconstruction with substantially fewer measurements than using the ℓ_1 -norm. Later in [23], he demonstrated that with a fixed number of measurements, the non-convex case can correct the corruption of a larger number of

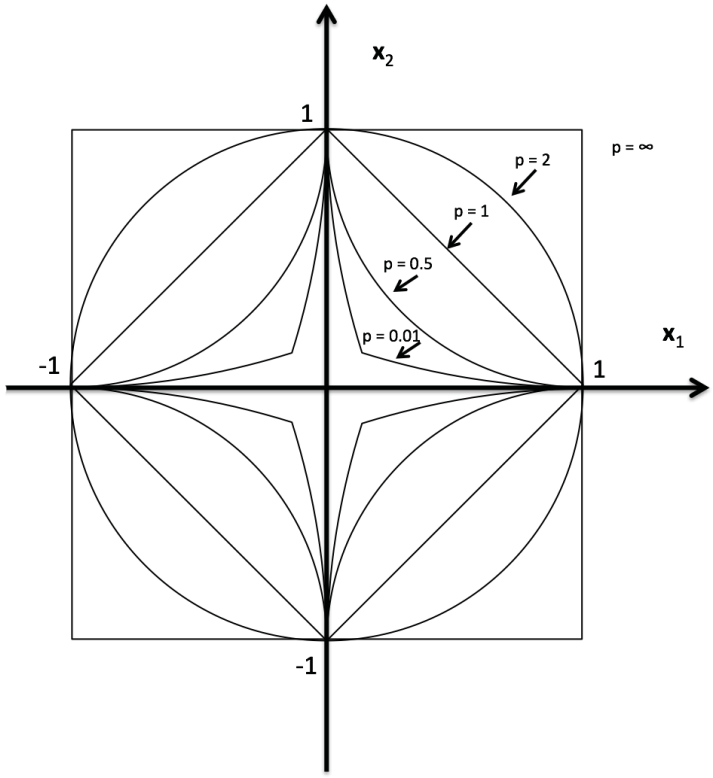

Figure 2.1: ℓ_p-norm with various values of p drawn in $\mathbb{R}^2$. When $p \to 0$, the unit ball gets closer to the $\mathbf{x}_1$ and $\mathbf{x}_2$ axes. [77]

entries. In terms of the restricted isometry property (RIP), Chartrand et al. [25] generalized the result of Candès [13] to an ℓ_p variant and determined a sufficient condition for exact recovery from perfect data via ℓ_p-minimization. An extensive study on exact recovery condition can be found in [42].

Saab et al. [112] studied ℓ_p-minimization in terms of its stability and robustness. They stated that ℓ_p-minimization (with $p < 1$) guarantees more stable and robust than ℓ_1-minimization depending on the restricted isometry constants and the noise level. Saab et al. [111] also studied the stability of ℓ_p-minimization for the sparse and compressible signals when measurements contain some additive noise and they gave the error bounds on the reconstruction error. Ince et al. [66] proposed a sparse reconstruction method based on ℓ_p-minimization knowing part of the signal support and they showed its stability and robustness.

Gribonval and Nielsen [55] considered a family of sparseness measures using ℓ_p-norm. With such a sparseness measure, they provided conditions for getting a unique sparse representation of

Table 2.1: Some popular non-convex penalty functions for sparsity regularization

Names	Formulas	Sub-gradients
ℓ_p^p norm [108]	$g_\lambda(\mathbf{x}_i) = \lambda \lvert\mathbf{x}_i\rvert^p, 0 < p < 1$	$\begin{cases} \infty & \text{if } \mathbf{x}_i{=}0, \\ \lambda p \mathbf{x}_i^{p-1} & \text{if } \mathbf{x}_i{>}0 \end{cases}$
Logarithm [128] [16]	$g_\lambda(\mathbf{x}_i) = \lambda \log(\lvert\mathbf{x}_i\rvert + \epsilon),$	$\frac{\lambda \text{sign}(\mathbf{x}_i)}{\lvert\mathbf{x}_i\rvert+\epsilon}$
SCAD [37]	$g_\lambda(\mathbf{x}_i) = \begin{cases} \lambda\lvert\mathbf{x}_i\rvert, & \lvert\mathbf{x}_i\rvert{\leq}\lambda \\ \frac{-\lvert\mathbf{x}_i\rvert^2+2\gamma\lambda\lvert\mathbf{x}_i\rvert-\lambda^2}{2(\gamma-1)}, & \lambda{<}\lvert\mathbf{x}_i\rvert{\leq}\gamma\lambda \\ \frac{(\gamma+1)\lambda^2}{2}, & \lvert\mathbf{x}_i\rvert{>}\gamma\lambda \end{cases}$	$\begin{cases} \lambda \text{sign}(\mathbf{x}_i), & \lvert\mathbf{x}_i\rvert{\leq}\lambda \\ \frac{\gamma\lambda \text{sign}(\mathbf{x}_i)-\mathbf{x}_i}{\gamma-1}, & \lambda{<}\lvert\mathbf{x}_i\rvert{\leq}\gamma\lambda \\ 0, & \lvert\mathbf{x}_i\rvert{>}\gamma\lambda \end{cases}$
MCP [136]	$g_\lambda(\mathbf{x}_i) = \begin{cases} \lambda\lvert\mathbf{x}_i\rvert - \frac{\lvert\mathbf{x}_i\rvert^2}{2\gamma}, & \lvert\mathbf{x}_i\rvert{\leq}\gamma\lambda \\ \frac{1}{2}\gamma\lambda^2, & \lvert\mathbf{x}_i\rvert{>}\gamma\lambda \end{cases}$	$\begin{cases} \lambda \text{sign}(\mathbf{x}_i) - \frac{\mathbf{x}_i}{\gamma}, & \lvert\mathbf{x}_i\rvert{\leq}\gamma\lambda \\ 0, & \lvert\mathbf{x}_i\rvert{>}\gamma\lambda \end{cases}$
ETP [43]	$g_{\lambda,\gamma}(\mathbf{x}_i) = \frac{\lambda}{1-\exp(-\gamma)} \left(1 - \exp(-\gamma\lvert\mathbf{x}_i\rvert)\right)$	$\frac{\lambda}{1-\exp(-\gamma)} \exp(-\gamma\lvert\mathbf{x}_i\rvert)$
Capped ℓ_1 [138]	$g_\lambda(\mathbf{x}_i) = \begin{cases} \lambda\lvert\mathbf{x}_i\rvert, & \lvert\mathbf{x}_i\rvert{<}\gamma \\ \lambda\gamma, & \lvert\mathbf{x}_i\rvert{\geq}\gamma \end{cases}$	$\begin{cases} \lambda \text{sign}(\mathbf{x}_i), & \lvert\mathbf{x}_i\rvert{<}\gamma \\ [0, \lambda], & \lvert\mathbf{x}_i\rvert{=}\gamma \\ 0, & \lvert\mathbf{x}_i\rvert{\geq}\gamma \end{cases}$
Geman's [46]	$g_\lambda(\mathbf{x}_i) = \frac{\lambda\lvert\mathbf{x}_i\rvert}{\lvert\mathbf{x}_i\rvert+\gamma}$	$\frac{\lambda\gamma \text{sign}(\mathbf{x}_i)}{(\lvert\mathbf{x}_i\rvert+\gamma)^2}$
Laplace [123]	$g_\lambda(\mathbf{x}_i) = \lambda \left(1 - \exp\left(-\frac{\lvert\mathbf{x}_i\rvert}{\gamma}\right)\right)$	$\frac{\lambda}{\gamma} \exp\left(-\frac{\lvert\mathbf{x}_i\rvert}{\gamma}\right)$
Gaussian [95]	$g_\lambda(\mathbf{x}) = n - \sum_{i=1}^n \left(\exp\left(-\frac{\mathbf{x}_i^2}{2\sigma^2}\right)\right)$	$\frac{1}{\sigma^2} \sum_{i=1}^n \left(\mathbf{x}_i \exp\left(-\frac{\mathbf{x}_i^2}{2\sigma^2}\right)\right)$

a signal from a dictionary and for solving all non-convex problems. In the variable selection field, Huang et al. [64], and Knight and Fu [72] studied asymptotic property (or the oracle property) of non-convex penalized estimators (ℓ_p).

Other ℓ_p-like Non-convex Penalty Functions

Beside ℓ_p-norm that can approximate the ℓ_0-norm better than the ℓ_1-norm, many other non-convex surrogate functions of ℓ_0-norm have been proposed, including ℓ_p^p norm [108], Smoothly Clipped Absolute Deviation (SCAD) [37], Logarithm [16], Minimax Concave Penalty (MCP) [136], Exponential-Type Penalty (ETP) [43], Capped L1 [138], Geman's [46], Laplace [123], and Gaussian [95]. Most of them are proposed in the context of variable/feature selection where indeed emerged the first use of non-convex penalty functions.

Rao et al. [108] proposed a slightly different version of ℓ_p-norm is called "p-norm-like diversity measures". We refer this as ℓ_p^p norm to distinct it from conventional ℓ_p-norm. In variable selection, another well-known non-convex penalty, the logaritm penalty, was also used for approximating the ℓ_0-norm by Weston et al. [128]. While Candès et al. [16] proposed an optimization method for this penalty in sparse signal approximation problems and showed its recovering capability though experiments. In this log penalty, a small shifting quantity is added to avoid infinite value when the

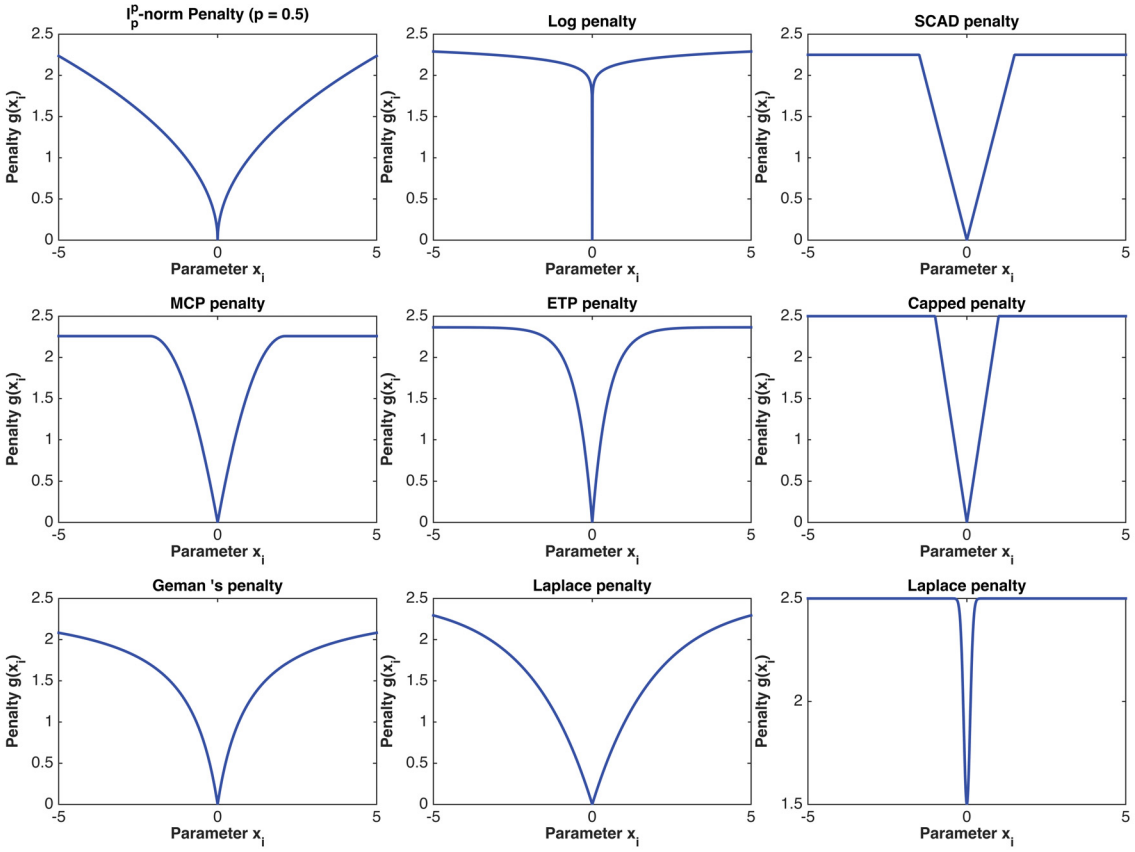

Figure 2.2: Illustration of some common convex and non-convex sparsity regularized functions [44]

parameter $\mathbf{x}_i \to 0$ as formulated in Table 2.1. This penalty has an interesting probability interpretation based on Bayesian framework with priors being a t-Student type distribution [120].

Fan and Li [37] proposed a Smoothly Clipped Absolute Deviation (SCAD) penalty function for variable selection. They highlighted three important properties of a good penalty function:

- *sparsity* – thresholding small coefficients to zero
- *continuity* – avoid instability in model selection
- *unbiasedness* – unbiased estimates for large coefficients

Based on these properties, Fan and Li [37] pointed out some drawbacks of the ℓ_1 penalty such as creating noticeably large bias on large coefficients. Then, they proved that SCAD has all necessary properties i.e. sparsity, continuity and unbiasedness.

Zhang [136] proposed a Minimax Concave Penalty (MCP) that can be considered as a variant of the SCAD penalty. The MCP has a bias controlling parameter γ, with larger values of γ, it provides smoother and less computationally complex but larger bias and less accurate variable selection. The MCP path converges to the ℓ_1 path as $\gamma \to \infty$. In the same spirit of SCAD and MCP, Gao et al. [43] proposed a non-convex penalty function which is called exponential-type penalty (ETP). The most essential point of ETP is that it bridges the ℓ_0 and ℓ_1 via a positive parameter γ. When this parameter approaches ∞ and 0, the limits of ETP are the ℓ_0 and ℓ_1 respectively. Using exponential helps to smooth the gaps between ℓ_0 and ℓ_1 (as shown in Fig. 2.2). Related to the above MCP penalty, Zhang [138] analyzed a multi-stage convex relaxation procedure with Capped-ℓ_1 regularization. This procedure solved a non-convex problem using multiple stage refining strategy.

Geman and Yang [46] applied a new sparsity regularization, which is even and non-decreasing on $[0, \infty]$, to a derivative operator of an image. Trzasko and Manduca [123] presented a homotopic approximation of the ℓ_0-minimization problem and applied it to recover undersampled magnetic resonance images (MRI). Their proposed method only guaranteed to find a local minimum, however, it allows accurate image reconstructions at higher undersampling rates than via ℓ_1-minimization. Mohimani [95] introduced a continous Gaussian-based penalty function. This function has a parameter σ (variance of Gaussian) controlling the smoothness of the ℓ_0-norm approximation. A larger value of σ gives smoother function g_σ but far away from ℓ_0-norm; and a smaller value of σ brings g_σ closer to ℓ_0-norm behavior.

These non-convex penalties can be used to approximate the rank function of a matrix. For examples, the Schatten p-norm [94], truncated nuclear norm [63] and log-det [39] [34].

ℓ_p-norm Optimization Algorithms

In this section, some optimization methods for solving ℓ_p-norm related objective functions are reviewed. There are three main groups: *iteratively reweighted* approaches, *DC programming* and *alternation approach*.

Iteratively Reweighted Approaches All methods in this category uses an iteratively update procedure which involves updating the solution and the weight vector/matrix at each iteration.

FOcal Underdetermined System Solver (FOCUSS) The FOcal Underdetermined System Solver (FOCUSS) was first named as Iterative Weighted Norm Minimization Algorithm [50] since it uses an iteratively update procedure for the weight matrix. It can be considered as a premier method in the group of iteratively reweighted approaches which we will introduce more details in this section.

Gorodnitsky et al. [52] used the FOCUSS algorithm to find a sparse solution for the Magnetoencephalography (MEG) problem, a reconstruction of the brain imaging. A detailed analysis, generalized extension and theoretical foundation of the algorithm were given in [107] and [51].

In the sense of signal processing, the MEG reconstruction problem can be modeled as a linear inverse problem with an under-determined linear system of equations,

$$\mathbf{y} \approx \mathbf{A}\mathbf{x} \tag{10}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a given matrix derived from the prior knowledge of the problem. $\mathbf{y} \in \mathbb{R}^{m \times 1}$ is the observed measurements. It is an under-determined problem ($m \ll n$) because the imaging resolution is much higher than the number of measurements. As a result, the number of solutions is infinite but sparse solutions are more suitable for the MEG problem because of the local nature of the activity in the brain. This is a similar idea to compressive sensing which having the same goal of finding sparse solutions from under-determined linear systems.

The basic FOCUSS algorithm [50] is briefly described here. In the noiseless case, observed signal $\mathbf{y}$ can be exactly represented by few columns of the given matrix $\mathbf{A}$. The minimum norm (**mn**) or the minimum energy solution is given by

$$\mathbf{x}_{\text{mn}} = \mathbf{A}^{+}\mathbf{y} \tag{11}$$

where "$+$" denotes the Moore-Penrose pseudo-inverse. In fact, the solution $\mathbf{x}_{\text{mn}}$ is the vector having the smallest ℓ_2-norm and satisfying (10). As the result of norm minimization, the energy of $\mathbf{x}_{\text{mn}}$ is spread out all elements, however, we attempt to find the solution that has few k non-zero entries. The sparse solution to (10) is needed for some problems such as sinusoid frequency estimation, power spectrum estimation, Direction of Arrival (DOA) estimation, etc. [50].

In general, the goal of the algorithm is to find the sparse solution of the following weighted ℓ_2-norm optimization problem:

$$min_{\mathbf{x}}\|\mathbf{W}^{-1}\mathbf{x}\|^2 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{A}\mathbf{x} \tag{12}$$

and it is equivalent to finding $\mathbf{x} = \mathbf{W}\mathbf{q}$, where $\mathbf{q}$ is the solution to the problem:

$$min_{\mathbf{q}}\|\mathbf{q}\|^2 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{W}\mathbf{A}\mathbf{q} \tag{13}$$

The weight matrix $\mathbf{W}$ is adaptively estimated. With an initial solution $\mathbf{x}_0$, the iterations of the basic FOCUSS algorithm are given by

$$\begin{aligned} \mathbf{W}_k &= \text{diag}(\mathbf{x}_k) \\ \mathbf{q}_{k+1} &= (\mathbf{A}\,\mathbf{W}_k)^{+}\mathbf{y} \\ \mathbf{x}_{k+1} &= \mathbf{W}_k\mathbf{q}_{k+1} \end{aligned} \tag{14}$$

The above mentioned basic FOCUSS algorithm can be applied when the matrix $\mathbf{A}$ is known or given but a FOCUSS-based dictionary learning algorithm was also proposed for the case of matrix $\mathbf{A}$ being unknown. The dictionary learning algorithm was described in details in [74]. We briefly introduce the algorithm here.

Given observed samples $\mathbf{Y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)$, we can find the solution $\mathbf{A}$ and $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_N)$ using maximum a posteriori (MAP) estimation:

$$(\hat{\mathbf{A}}_{MAP}, \hat{\mathbf{X}}_{MAP}) = \arg\max_{\mathbf{A},\mathbf{X}} P(\mathbf{A}, \mathbf{X}\,|\mathbf{Y}) \tag{15}$$

We have,

$$\begin{aligned} P(\mathbf{A},\mathbf{X}|\mathbf{Y}) &= P(\mathbf{Y}|\mathbf{A},\mathbf{X})P(\mathbf{A},\mathbf{X})/P(\mathbf{Y}) = c\mathcal{X}(\mathbf{A}\in(\mathcal{A}))P(\mathbf{Y}|\mathbf{A},\mathbf{X})P(\mathbf{X})/P(\mathbf{Y}) \\ &= \frac{c\mathcal{X}(\mathbf{A}\in(\mathcal{A}))}{P(\mathbf{Y})}\prod_{k=1}^{N}P(\mathbf{y}_k|\mathbf{A},\mathbf{x}_k)P_p(\mathbf{x}_k) \\ &= \frac{c\mathcal{X}(\mathbf{A}\in(\mathcal{A}))}{P(\mathbf{Y})}\prod_{k=1}^{N}P_q(\mathbf{y}-\mathbf{A}\mathbf{x}_k)P_p(\mathbf{x}_k) \end{aligned} \tag{16}$$

We assume that the distributions of the additive noise ν and the signal $\mathbf{x}$ are Gaussian with the following form for $P_p(\mathbf{x})$

$$P_p(\mathbf{x}) = Z_p^{-1}e^{-\gamma_p d_p(\mathbf{x})}, Z_p = \int e^{-\gamma_p d_p(\mathbf{x})}dx \tag{17}$$

where the function $d_p(\mathbf{x})$ is a ℓ_p-norm-like function and defined as:

$$d_p(\mathbf{x}) = \|\mathbf{x}\|_p^p = \sum_{i=1}^{n}|\mathbf{x}_i|^p, 0\le p\le 1 \tag{18}$$

Similarly, we have another form for $P_q(\nu)$ with $q = 2$ and $d_p(\nu) = \|\nu\|_2^2$. Given observed vector $\mathbf{y}$, we want to solve for $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{v}$ by minimizing the following:

$$(\mathbf{A},\mathbf{x}) = \arg\min_{\mathbf{A},\,\mathbf{X}}\langle d_q(y-\mathbf{A}\mathbf{x}) + \lambda d_p(\mathbf{x})\rangle \tag{19}$$

or equivalently,

$$(\mathbf{A},\mathbf{x}) = \arg\min_{\mathbf{A},\,\mathbf{X}}\left\langle \|y-\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_p^p\right\rangle \tag{20}$$

where λ is the regularization parameter.

The algorithm contains one more major step, dictionary learning using gradient descent, together with sparse vector estimation step.

Sparse vector $\mathbf{x}$ selection step at each iteration k is given by,

$$\begin{aligned} W_k &= \text{diag}(|\mathbf{x}_i|^{2-p}) \\ \mathbf{x}_k &= W_k\hat{A}^T\left(\lambda_k I + \hat{A}W_k\hat{A}^T\right)\mathbf{y}_k \end{aligned} \tag{21}$$

The matrix $\mathbf{A}$ is updated by,

$$\begin{aligned}
\Sigma_{y\hat{\mathbf{x}}} &= \frac{1}{N}\sum_k y_k \hat{\mathbf{x}}_k^T \\
\Sigma_{\hat{\mathbf{x}}\hat{\mathbf{x}}} &= \frac{1}{N}\sum_k \hat{\mathbf{x}}_k \hat{\mathbf{x}}_k^T \\
\delta\hat{A} &= \hat{A}\Sigma_{\hat{\mathbf{x}}\hat{\mathbf{x}}} - \Sigma_{y\hat{\mathbf{x}}} \\
\hat{A} &\leftarrow \hat{A} - \gamma\left(\delta\hat{A} - trace(\hat{A}^T\delta\hat{A})\hat{A}\right)
\end{aligned} \tag{22}$$

where $\gamma > 0$ is a constant controlling the learning rate.

An improved FOCUSS-based dictionary learning algorithm was proposed in [96] on three aspects: adjusting the regularization parameter, normalizing learned matrix $\hat{\mathbf{A}}$ and avoiding local optima. First, the authors suggested a heuristic method to update regularization parameter λ_k for each observed vector $\mathbf{y}_k$ to improve the quality of the solution. Second, the matrix $\hat{\mathbf{A}}$ is normalized to $\|\hat{\mathbf{A}}\|_F = 1$ avoiding problems with large magnitude elements. Last but not least, the authors proposed to reinitialize $\mathbf{x}_k$ when the sparsity is too low. Since the optimization problem (20) is concave $(p < 1)$, the FOCUSS algorithm may converge to a local minima but a good initialization can bring it to the global solution [96].

Iteratively Reweighted Least Square (IRLS) Chartrand et al. [27] considered iteratively reweighted least squares (IRLS) approach for solving the following problem for the case of $0 < p < 1$

$$\min_{\mathbf{x}} \|\mathbf{x}\|_p \text{ subject to } \Phi\mathbf{x} = \mathbf{y} \tag{23}$$

They proposed a regularized strategy for IRLS to improve sparsity of the recovery while FOCUSS algorithm, mentioned earlier, can be considered as an unregularized version of IRLS.

Similar to (20) and (21), the ℓ_p objective function in (23) is approximated by a "weighted" ℓ_2-norm. Thus, we have,

$$\min_{\mathbf{x}} \sum_{i=1}^{N} w_i \mathbf{x}_i^2, \quad s.t. \quad y = \Phi\mathbf{x}, \tag{24}$$

As (24) is a first-order approximation to the ℓ_p objective function, the weights are updated from

the previous $\mathbf{x}^{(k-1)}$ iteration as $w_i = |\mathbf{x}_i^{(n-1)}|^{p-2}$ and the new iteration $\mathbf{x}^{(k)}$ is given as:

$$\mathbf{x}^{(k)} = \mathbf{Q}_k \Phi^T (\Phi \mathbf{Q}_k \Phi^T)^{-1} \mathbf{y}, \tag{25}$$

where $\mathbf{Q}_k$ is a diagonal matrix with entries $1/w_i = |\mathbf{x}_i^{(k-1)}|^{2-p}$.

Chartrand et al. [27] suggested to regularize the weights by incorporating a small constant $\epsilon \in (0, 1)$ as

$$w_i = \left(\left(\mathbf{x}_i^{(k-1)} \right)^2 + \epsilon \right)^{p/2-1} \tag{26}$$

As shown in [27], this ϵ-regularized IRLS algorithm can converge to the global minimum of (23). For ϵ-regularized IRLS, ϵ is initialized to 1 and $\mathbf{x}^{(0)}$ initialized to the ℓ_2-norm minimizing solution of $\mathbf{y} = \Phi\mathbf{x}$. Starting with such a large ϵ would eliminate unwanted local minima and brings $\mathbf{x}$ to a nearby point where possibly contains the global solution. Then decreasing ϵ draws $\mathbf{x}$ toward the global solution and eventually converges to it as $\epsilon \to 0$.

Iteratively Reweighted ℓ_1 (IRL1) Candès et al. [16] proposed an iteratively reweighted ℓ_1 minimization algorithm. The "weighted" ℓ_1 minimization problem is defined as,

$$\min_{\mathbf{x}} \sum_{i=1}^{N} w_i |\mathbf{x}_i|, \quad s.t. \quad y = \Phi\mathbf{x}, \tag{27}$$

where w_i are positive weights.

Similar to its "weighted" ℓ_2 counterpart, this problem can be solved using an iterative algorithm to estimate $\mathbf{x}$ and then redefine the weights w_i. The algorithm is as follows:

(1) Initialize $k = 0, w_i^{(0)} = 1, i = 1, \cdots, N$

(2) Solve weighted ℓ_1-norm minimization problem in Eqn. (27) using soft-thresholding

(3) Update the weights $w_{ij}^{(k+1)}$ for each element of $\mathbf{x}$

$$w_i^{(k)} = \frac{1}{(|\mathbf{x}_i^{(k)}| + \epsilon)} \tag{28}$$

(4) Increase k and go to step (2) until convergence or reaching a specified maximum number of iterations k_{max}.

Candès et al. [16] established a connection between the log-sum penalty function and the reweighted ℓ_1 minimization in which the reweighted ℓ_1 minimization gives the solution to the log-sum problem

$$\min_{\mathbf{x}} \sum_{i=1}^{N} log(|\mathbf{x}_i| + \epsilon) \quad \text{s.t.} \quad \mathbf{y} = \Phi\mathbf{x} \tag{29}$$

Since the log-sum penalty function can encourage more sparseness than ℓ_1 norm, reweighted ℓ_1 minimization can improve the recovery of sparse signals.

Like other iteratively reweighted approaches, a good initialization for the algorithm is important. Therefore, the authors suggested to use the unweighted ℓ_1 solution as a starting point.

Iteratively Reweighted Nuclear Norm (IRNN) Lu et al. [84] proposed an Iteratively Reweighted Nuclear Norm (IRNN) algorithm to solve the general low-rank minimization problem in Eqn. (30) which is non-convex and non-smooth.

$$\min_{\mathbf{X}} \lambda \sum_{i=1}^{r} g(\sigma_i(\mathbf{X})) + f(\mathbf{X}) \tag{30}$$

where $\sigma(\mathbf{X})$ is the vector of singular values of $\mathbf{X} \in \mathbb{R}^{m\times n}$, g denotes the regularized function and f is the constrained or loss function. This problem can be considered as a general rank regularization problem.

Based on the definition and properties of the supergradient of a concave function g_λ [84], we have

$$g_\lambda(\sigma_i) \leq g_\lambda(\sigma_i^k) + w_i^k(\sigma_i - \sigma_i^k) \tag{31}$$

where

$$w_i^k \in \partial g_\lambda(\sigma_i^k) \tag{32}$$

and $\sigma_i = \sigma_i(\mathbf{X}), \sigma_i^k = \sigma_i(\mathbf{X}^k)$. Since the supergradient of g is monotonically decreasing on $[0, \infty)$, we also have

$$0 \leq w_1^k \leq w_2^k \leq \cdots \leq w_r^k \tag{33}$$

Using Eqn. (31), we minimize the following relaxed problem instead:

$$\begin{aligned}\mathbf{X}^{k+1} &= \arg\min_{\mathbf{X}} \sum_{i=1}^{r} g_\lambda(\sigma_i^k) + w_i^k(\sigma_i - \sigma_i^k) + f(\mathbf{X}) \\ &= \arg\min_{\mathbf{X}} \sum_{i=1}^{m} w_i^k \sigma_i + f(\mathbf{X})\end{aligned} \tag{34}$$

Eqn. (34) gives us the weighted nuclear norm problem which is an extension of the previously mentioned weighted ℓ_1-norm and weighted least square problems. Due to the non-convex penalty function g_λ, solving the weighted nuclear norm problem, a non-convex optimization problem, is much more difficult than the weighted ℓ_1-norm problem. However, Lu et al. [84] proposed an approach to go around this non-convex problem by linearizing $f(\mathbf{X})$ at $\mathbf{X}^k$:

$$f(\mathbf{X}) \approx f(\mathbf{X}^k) + \left\langle \nabla f(\mathbf{X}^k), \mathbf{X} - \mathbf{X}^k \right\rangle + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}^k\|_F^2 \tag{35}$$

where $\mu > L(f)$. Then, replacing this $f(\mathbf{X})$ in Eqn. (34) with the formula in Eqn. (35), we turn it into another problem having a closed form solution.

$$\begin{aligned}\mathbf{X}^{k+1} &= \arg\min_{\mathbf{X}} \sum_{i=1}^{m} w_i^k \sigma_i + f(\mathbf{X}^k) + \left\langle \nabla f(\mathbf{X}^k), \mathbf{X} - \mathbf{X}^k \right\rangle + \frac{\mu}{2}\|\mathbf{X} - \mathbf{X}^k\|_F^2 \\ &= \arg\min_{\mathbf{X}} \sum_{i=1}^{m} w_i^k \sigma_i + \frac{\mu}{2}\left\|\mathbf{X} - \left(\mathbf{X}^k - \frac{1}{\mu}\nabla f(\mathbf{X}^k)\right)\right\|_F^2\end{aligned} \tag{36}$$

The solution is then obtained by using weighted singular value thresholding.

In general, the algorithm iteratively updates $w_i^k, i = 1, \cdots, r$ using Eqn. (32) and $\mathbf{X}^{k+1}$ using Eqn. (36). The whole procedure is as follows:

(1) Initialize $k = 0, \mathbf{X}^{(0)}, w_i^{(0)} = 1, i = 1, \cdots, r$

(2) Solve weighted nuclear norm minimization problem in Eqn. (36) using weighted singular value thresholding

$$\mathbf{X}^{(k+1)} = \mathbf{U}\mathcal{S}_{\lambda w}(\Sigma)\mathbf{V}^T \tag{37}$$

where $\mathbf{Y} = \left(\mathbf{X}^k - \frac{1}{\mu}\nabla f(\mathbf{X}^k)\right)$ and $\mathbf{U}\Sigma\mathbf{V}^T$ is the SVD of $\mathbf{Y}$. The shrinkage operator $\mathcal{S}_{\lambda w}(\Sigma) = \text{diag}\left\{(\Sigma_{ii} - \lambda w_i)_+\right\}$

(3) Update the weights $w_i^{(k+1)}$ as

$$w_i^{k+1} \in \partial g_\lambda(\sigma_i(\mathbf{X}^{k+1})) \tag{38}$$

(4) Increase k and go to step (2) until convergence or reaching a specified maximum iterations k_{max}.

DC Programming/Multi-stage Convex Relaxation Gasso et al. [44] proposed to use a well-known procedure in non-convex optimization, called Difference of Convex functions (DC) programming, to solve the general problem (39).

$$\hat{\mathbf{x}} \simeq \arg\min_{\mathbf{x}} \|\mathbf{y} - \Phi\mathbf{x}\|_2^2 + \lambda \sum_{i=1}^{n} g(\mathbf{x}_i) \tag{39}$$

In practical term, they considered an equivalent variation of this problem by splitting $\mathbf{x}_i$ into two positive terms $\mathbf{x}_i^+$ and $\mathbf{x}_i^-$ so that $\mathbf{x}_i = \mathbf{x}_i^+ - \mathbf{x}_i^-$

$$\begin{aligned} \min_{\mathbf{x}^+, \mathbf{x}^- \in \mathbb{R}^n} \frac{1}{2}\|y - \Phi(\mathbf{x}^+ - \mathbf{x}^-)\|^2 + \sum_{i=1}^{n} g_\lambda(\mathbf{x}_i^+ + \mathbf{x}_i^-) \\ \text{s.t.} \quad \mathbf{x}_i^+ \geq 0,\ \mathbf{x}_i^- \geq 0,\ \ \forall j = 1, \cdots, n \end{aligned} \tag{40}$$

where the vector $\mathbf{x}^+$ and $\mathbf{x}^-$ contain elements $\mathbf{x}_i^+$ and $\mathbf{x}_i^-$ respectively.

We will briefly introduce the basic idea of DC programming. For more details about the algorithm, theory and proof, one can refer [62] for a full review on DC programming. DC algorithm considers solving the following general minimization problem:

$$\min_{\mathbf{x} \in \mathbb{R}^n} J(\mathbf{x}) \tag{41}$$

where $J(.)$ is a non-convex (may be non-smooth) objective function. This function can be split into two functions such that $J(\mathbf{x}) = J_1(\mathbf{x}) - J_2(\mathbf{x})$. Then the minimization problem becomes as,

$$\min_{\mathbf{x} \in \mathbb{R}^n} J_1(\mathbf{x}) - J_2(\mathbf{x}) \tag{42}$$

where $J_1(\cdot)$ and $J_2(\cdot)$ are convex functions.

Then we have the dual of the above minimization problem given as,

$$\min_{\mathbf{z}\in\mathbb{R}^n} J_2^*(\mathbf{z}) - J_1^*(\mathbf{z}) \tag{43}$$

where $J_1^*(\cdot)$ and $J_2^*(\cdot)$ are the conjugate function of $J_1(\cdot)$ and $J_2(\cdot)$, respectively. The conjugate function of $J_k(\cdot), k = \{1, 2\}$ is defined as,

$$J_k^*(\mathbf{z}) = \sup_{\mathbf{x}\in\mathbb{R}^n} \{\langle \mathbf{x}, \mathbf{z}\rangle - J_k(\mathbf{x})\} \tag{44}$$

The DC programming then iteratively solves the primal (Eqn. (42)) and the dual (Eqn. (43)) problems. The simple version of DC algorithm is summarized as follows [44].

(1) Initialize estimation $\mathbf{x}^0 \in dom\ J_1$ with $dom\ J_1 = \left\{\mathbf{x} \in \mathbb{R}^d : J_1(\mathbf{x}) < \infty\right\}$

(2) Solve dual problem to find $\mathbf{z}^k \in \partial J_2(\mathbf{x}^t)$

(3) Solve primal problem to find $\mathbf{x}^{k+1} \in \partial J_1^*(\mathbf{z}^t)$

(4) Increase k and go to step 2 until convergence or reaching a specified maximum iterations k_{max}.

To apply DC programming to the general non-convex optimization problem (40), Gasso et al. [44] decomposed the regularization $g_\lambda(\cdot)$ as the difference of two convex functions,

$$g_\lambda(.) = g_{vex}(.) - h(.) \tag{45}$$

In this way, the objective function of the problem (40) will split into the difference of two functions: $J_1 = \frac{1}{2}\|y - \Phi(\mathbf{x}^+ - \mathbf{x}^-)\|^2 + \sum_{i=1}^n g_{vex}(\mathbf{x}_i^+ + \mathbf{x}_i^-)$ and $J_2 = \sum_{i=1}^n h(\mathbf{x}_i^+ + \mathbf{x}_i^-)$

For non-convex penalty functions $g_\lambda(\cdot)$, the decomposition still holds if $g_{vex}(\cdot)$ and $h(\cdot)$ are convex functions. Therefore, Gasso et al. [44] defined those functions explicitly based on ℓ_1 function

as follows,

$$\begin{aligned} g_{vex}(\cdot) &= \lambda|\cdot| \\ h(\cdot) &= \lambda|\cdot| - g_\lambda(\cdot) \end{aligned} \tag{46}$$

Finally, at each iteration k, the DC algorithm minimizes the following problem,

$$\begin{aligned} \min_{\mathbf{x}^+,\mathbf{x}^-} \frac{1}{2}\|y - \Phi(\mathbf{x}^+ - \mathbf{x}^-)\|^2 + \sum_{i=1}^{n} \lambda(\mathbf{x}_i^+ + \mathbf{x}_i^-) - \sum_{i=1}^{n} \mathbf{z}_i^k(\mathbf{x}_i^+ + \mathbf{x}_i^-) \\ \text{s.t.} \quad \mathbf{x}_i^+ \geq 0, \mathbf{x}_i^- \geq 0, \ \ \forall j = 1, \cdots, d \end{aligned} \tag{47}$$

where $\mathbf{z}_i^k \in \partial h(\mathbf{x}_i^{+k} + \mathbf{x}_i^{-k})$. Moreover, we can compute the sub-gradient of h at any $\mathbf{x}_i$ value for each iteration.

Some penalty functions are differentiable such as SCAD then we can take the derivative. However, others are non-differentiable then we can apply some tricks, for examples, adding an ϵ term to avoid the zero point (for log penalty) or setting z as any element of the sub-gradient (for MCP). For convenience, we list all the sub-gradients of various penalties in Table 2.1.

Alternating Direction Method of Multipliers (ADMM) In general, the alternating direction method of multipliers (ADMM) decomposes a complex problem, having two or more variables, into smaller subproblems (usually easier) and solves them iteratively.

Chartrand and Wohlberg [26] proposed to use an efficient ADMM algorithm to solve the compressive sensing problem which encourages both sparsity and group sparsity of the signals. This sparse and group-sparse compressive sensing model is formulated as

$$\min_{\mathbf{x}} \alpha\|\mathbf{x}\|_1 + \beta \sum_{i=1}^{M} \|\mathbf{x}_i\|_2 + \frac{1}{2}\|\Phi\mathbf{x} - \mathbf{y}\|_2^2 \tag{48}$$

An auxiliary variable $\mathbf{W}$ is introduced to split the main problem into solvable subproblems. Using the method of multipliers, a dual variable (or Lagrange multiplier) Λ is also added to enforce

the equality constraint for $\mathbf{W}$ and $\mathbf{X}$:

$$\min_{\mathbf{W},\mathbf{X}} \alpha\|\mathbf{W}\|_1 + \beta\sum_{i=1}^{M}\|\mathbf{W}^i\|_2 + \frac{1}{2}\|\mathbf{W}-\mathbf{X}-\Lambda\|_F^2 + \frac{1}{2}\|\Phi\mathbf{X}-\mathbf{Y}\|_F^2 \tag{49}$$

With fixed $\mathbf{W}$, solving $\mathbf{X}$ subproblem is quadratic which we have a closed form solution:

$$(\mathbf{I}+\Phi^T\Phi)\mathbf{X} = \mathbf{W} + \Phi^T\mathbf{Y} \tag{50}$$

With fixed $\mathbf{X}$, we solve the following problem for $\mathbf{W}$ using soft thresholding.

$$\min_{\mathbf{W}} \alpha\|\mathbf{W}\|_1 + \beta\sum_{i=1}^{M}\|\mathbf{W}^i\|_2 + \frac{1}{2}\|\mathbf{W}-\mathbf{X}\|_F^2 \tag{51}$$

The solution of the above problem is given as

$$\mathbf{W}^i = \mathcal{S}_1(\mathbf{s}_1(\mathbf{X}^i,\alpha),\beta) \tag{52}$$

where $\mathbf{W}^i$ are rows of $\mathbf{W}$. $\mathbf{s}_1$ and $\mathcal{S}_1$ are shrinkage mappings computed as $\mathbf{s}_1(\mathbf{x},\alpha)_i = \frac{\mathbf{x}_i}{|\mathbf{x}_i|}\max\{0,|\mathbf{x}_i|-\alpha\}$ and $\mathcal{S}_1(\mathbf{x},\alpha) = \frac{\mathbf{x}}{\|\mathbf{x}\|_2}\max\{0,\|\mathbf{x}\|_2-\alpha\}$, respectively

In addition to the convex approach, Chartrand and and Wohlberg [26] generalized the problem to take advantage of non-convex optimization. They formulate

$$\min_{\mathbf{W},\mathbf{X}} \alpha G_{\alpha,p}(\mathbf{W}) + \beta\sum_{i=1}^{M} g_{\beta,q}(\|\mathbf{W}^i\|_2) + \frac{1}{2}\|\mathbf{W}-\mathbf{X}-\Lambda\|_F^2 + \frac{1}{2}\|\Phi\mathbf{X}-\mathbf{Y}\|_F^2 \tag{53}$$

where the generalize non-convex functions $G_{\alpha,p}$ and $g_{\beta,p}$ enforcing sparsity and group-sparsity are defined based on shrinkage operators as follows:

$$\arg\min_{\mathbf{w}\in\mathbb{R}^n} \alpha G(\mathbf{w}) + \frac{1}{2}\|\mathbf{w}-\mathbf{x}\|_2^2 = \mathcal{S}_p(\mathbf{x},\alpha) \tag{54}$$

with $G(\mathbf{w}) = \sum_{i=1}^{N} g(\mathbf{w}_i)$ for some scalar function g. The function g is defined as:

$$\arg\min_{\mathbf{w}\in\mathbb{R}^n} \alpha g(\|\mathbf{w}\|_2) + \frac{1}{2}\|\mathbf{w}-\mathbf{x}\|_2^2 = \mathcal{S}_p(\mathbf{x},\alpha) \tag{55}$$

2.1.3 Sparse Representation and Dictionary Learning

In this section, we first briefly introduce the basic idea of sparse representation and how it is applied to classification problems. Then the idea leading to dictionary learning from sparse coding and dictionary learning algorithms will be presented.

Sparse Representation/Coding

The idea of sparse coding is that a signal can be represented as a linear combination of basis elements. The basis can be either orthogonal or bi-orthogonal which is computed by taking inner products of the signals with the basis but those bases are limited in representing complex signal. Therefore, overcomplete dictionaries which has more elements (or atoms) than the dimension of the signal, were proposed as the basis.

Let's denotes the overcomplete dictionary as $\mathbf{B} = [\mathbf{b}_1, \cdots \mathbf{b}_l] \in \mathbb{R}^{n \times l}$, where $l \geq n$ and each column of $\mathbf{B}$ is the dictionary's atoms. To find the sparse coding of $\mathbf{x} \in \mathbb{R}^n$ using atoms of $\mathbf{B}$, one can solve the following optimization problem:

$$\hat{\alpha} = \arg\min_{\alpha'} \|\alpha'\|_0 \quad \text{subject to} \quad \mathbf{x} = \mathbf{B}\alpha' \tag{56}$$

Since we want to enforce sparsity of the representation, $\|\alpha\|_0$ is used to find the sparsest solution to the underdetermined linear system of equations $\mathbf{x} = \mathbf{B}\alpha$. However, similar to CS, the problem in Eqn. (56) is an NP-hard problem and it cannot be solved in polynomial time. Then, one can solve the following ℓ_1-minimization problem instead

$$\hat{\alpha} = \arg\min_{\alpha'} \|\alpha'\|_1 \quad \text{subject to} \quad \mathbf{x} = \mathbf{B}\alpha' \tag{57}$$

The problem in Eq. (57) is the closest convex optimization problem in Eq. (56). This problem (57) usually referred as Basis Pursuit. When B has incoherent columns (i.e. having uncorrelated columns or being close to orthogonal), the solutions of (57) is unique and equal to a sufficiently sparse solution of (56).

Sparse Representation-based Classification

This sub-section briefly describes how sparse representation is used for recognition/classification. Given a set of N training images for each of L classes, we can extract M-dimensional feature vectors from these images. Let's denotes $\mathbf{B}_k = [\mathbf{x}_{k1}, \cdots, \mathbf{x}_{kj}, \cdots, \mathbf{x}_{kN}]$ as an $M \times N$ matrix of feature vectors belong to the same k-th class, where $\mathbf{x}_{kj}$ denote the features from the j-th training image of the k-th class. Combining training samples from all classes to form a big matrix $\mathbf{B}$ as

$$\begin{aligned} \mathbf{B} &= [\mathbf{B}_1, \cdots, \mathbf{B}_L] \in \mathbb{R}^{M \times (N \times L)} \\ &= [\mathbf{x}_{11}, \cdots, \mathbf{x}_{1N} | \mathbf{x}_{21}, \cdots, \mathbf{x}_{2N} | \cdots\cdots | \mathbf{x}_{L1}, \cdots, \mathbf{x}_{LN}] \end{aligned} \tag{58}$$

A testing image $\mathbf{y} \in \mathbb{R}^M$ of unknown class can be represented as a linear combination of the training vector as

$$\mathbf{y} = \sum_{i=1}^{L} \sum_{j=1}^{M} \alpha_{ij} \mathbf{x}_{ij} = \mathbf{B}\alpha \tag{59}$$

where the coefficients $\alpha = [\alpha_{11}, \cdots, \alpha_{1N} | \alpha_{21}, \cdots, \alpha_{2N} | \cdots\cdots | \alpha_{L1}, \cdots, \alpha_{LN}]^T$ with $\alpha_{ij} \in \mathbb{R}$. T denotes the transpose operation.

Given enough training samples of each class, any new testing image $\mathbf{y} \in \mathbb{R}^M$ from the same class k can be approximated by the training samples from that class. This means that most of the coefficients are close to zero except the ones associated with the same class k. Thus, α is a sparse vector and it can be computed by solving the following optimization problem:

$$\hat{\alpha} = \arg\min_{\alpha'} \|\alpha'\|_1 \quad \text{subject to} \quad \mathbf{y} = \mathbf{B}\alpha' \tag{60}$$

or approximate α with Basis Pursuit DeNoising (BPDN)

$$\hat{\alpha} = \arg\min_{\alpha'} \|\alpha'\|_1 \quad \text{subject to} \quad \|\mathbf{y} - \mathbf{B}\alpha'\|_2 \leq \epsilon \tag{61}$$

when the observations are noisy as $\mathbf{y} = \mathbf{B}\alpha' + \eta$. Based on the fact that the coefficients associated with a single class k will have high values comparing to other parts of the estimated coefficients, $\hat{\alpha}$. Thus, we set all the coefficients not associated with class k to zero to compute the residual error of

class k as

$$r_k(y) = \|\mathbf{y} - \mathbf{B}\boldsymbol{\delta}_k(\hat{\alpha})\| \tag{62}$$

where $\boldsymbol{\delta}_k(\hat{\alpha}) = [0 \cdots 0 \cdots \boldsymbol{\alpha}_k \cdots 0 \cdots 0]^T$ with $\boldsymbol{\alpha}_j = 0; \forall j \neq k$ Then, class(y) $= \arg\min r_k(\mathrm{y})$. The testing image y is represented by a linear combination of all images in the dictionary $\mathbf{B}$. The purpose of computing residuals is to find the class k having the most influence in the sparse representation. The smaller the residual the more influence class k has on the outcome, so it is more likely that y has label of class k.

Dictionary Learning

The main idea is to learn a dictionary directly from the data instead of using a pre-determined dictionary $\mathbf{B}$. This usually gives better representation and provides improved results in many applications, e.g. image restoration and classification. This section will briefly present some well-known algorithms for dictionary learning.

There have been several dictionary learning algorithms, such as the Method of Optimal Directions (MOD) [36] and the K-SVD algorithm [1]. Given a set of examples $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_n]$, the K-SVD and MOD algorithms aim at finding a dictionary $\mathbf{B}$ and a sparse coefficient matrix $\boldsymbol{\Gamma}$ that minimize the following error,

$$(\hat{\mathbf{B}}, \hat{\boldsymbol{\Gamma}}) = \arg\min_{\mathbf{B}_i, \boldsymbol{\Gamma}_i} \|\mathbf{X} - \mathbf{B}\boldsymbol{\Gamma}\|_F^2 \quad \text{subject to} \quad \|\gamma_i\|_0 \leq T_0 \tag{63}$$

where γ_i is the i-th column of $\boldsymbol{\Gamma}$ and T_0 denotes level of sparsity. The main iteration of K-SVD and MOD algorithms contains two stages: sparse coding and dictionary updating. First, a column-normalized dictionary $\mathbf{B}$ is initialized, then during sparse coding step the representation vector γ_i for each sample $\mathbf{x}_i$ is computed while fixing $\mathbf{B}$,

$$\min_{\gamma_i} \|\mathbf{x}_i - \mathbf{B}\gamma_i\|_2^2 \quad \text{s.t.} \quad \|\gamma_i\|_0 \leq T_0, \forall i = 1, \cdots, n \tag{64}$$

Any sparse coding algorithms such as Orthogonal Matching Pursuit (OMP) [122] and Basis Pursuit (BP) [29] can be used to solve the above problem. During dictionary update step, MOD algorithm

updates all the atoms simultaneously by optimizing the Eqn. (65) with a closed form solution in Eqn. (66).

$$\arg\min_{\mathbf{B}} \|\mathbf{X} - \mathbf{B}\mathbf{\Gamma}\|_F^2 \tag{65}$$

with the solution as

$$\mathbf{B} = \mathbf{X}\mathbf{\Gamma}^T(\mathbf{\Gamma}\mathbf{\Gamma}^T)^{-1} \tag{66}$$

Meanwhile K-SVD performs dictionary updating atom-by-atom efficiently as in Eqn. (67).

$$\|\mathbf{X} - \mathbf{B}\mathbf{\Gamma}\|_F^2 = \|\mathbf{X} - \sum_j \mathbf{b}_j\gamma_j^T\|_2^2 = \|\left(\mathbf{X} - \sum_{j\neq j_0} \mathbf{b}_j\gamma_j^T\right) - \mathbf{b}_{j_0}\gamma_{j_0}^T\|_2^2 \tag{67}$$

where γ_j^T is the jth row of $\mathbf{\Gamma}$. To update $\mathbf{b}_{j_0}$ and $\gamma_{j_0}^T$, we can pre-compute the first term $\left(\mathbf{X} - \sum_{j\neq j_0} \mathbf{b}_j\gamma_j^T\right)$ in the above equation. The optimal solution $\mathbf{b}_{j_0}$ and $\gamma_{j_0}^T$ are found by an SVD decomposition. The convergence of the K-SVD algorithm is speedup significantly since only a subset of the columns of the first term is taken into account.

2.1.4 Robust PCA: A Review

In this section we will take a look at an extension of PCA to a problem which is closely related to low-rank approximation problem. Then, we will review some recent non-convex and online approaches as applied to Robust PCA.

Ideas of RPCA

The basic idea of **PCA** is that given data points as column vectors of a matrix $\mathbf{M} \in \mathbb{R}^{n_1\times n_2}$, since data have low intrinsic dimensionality, the matrix should have low-rank

$$\mathbf{M} = \mathbf{L}_0 + \mathbf{N}_0 \tag{68}$$

Where $\mathbf{L}_0$ is low rank and $\mathbf{N}_0$ is small noise matrix.

Meanwhile the idea of **Robust PCA** [17] is that given a data matrix $\mathbf{M}$, we know that it can be

decomposed as

$$\mathbf{M} = \mathbf{L}_0 + \mathbf{S}_0 \tag{69}$$

Where $\mathbf{L}_0$ is low rank and $\mathbf{S}_0$ is sparse. From the above definitions, we know that both PCA and robust PCA share a common idea of separation problem which decompose a data matrix $\mathbf{M}$ into a low-rank matrix plus another matrix. Mathematically, PCA is formulated as $\mathbf{M} = \mathbf{L}_0 + \mathbf{N}_0$ whereas robust PCA is defined as $\mathbf{M} = \mathbf{L}_0 + \mathbf{S}_0$. They both have the low-rank matrix but the only difference is the second term. Actually, because the nature of the data matrix $\mathbf{M}$ in PCA and robust PCA are dissimilar, one is normal matrix while the other one is highly corrupted matrix. Therefore, their main objectives are different, the goal of PCA is to find best rank-k approximation of $\mathbf{M}$ while robust PCA focus on recovering from $\mathbf{M}$ the best low-rank matrix $\mathbf{L}_0$ and the sparse component with entries having arbitrarily large magnitude.

Solution for Robust PCA: we know that we need to decompose the matrix $\mathbf{M}$ intro the low rank and the sparse component. It seems impossible to solve if we think of the number of unknowns in $\mathbf{L}_0$ and $\mathbf{S}_0$ comparing with the given measurements in $\mathbf{M}$. However, it is surprising that this decomposition problem can be solved simply by tractable convex optimization. Using the **Principal Component Pursuit** (PCP) [17] a convex optimization problem, to solve

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 \text{ s.t. } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{70}$$

With variables $\mathbf{L}, \mathbf{S} \in \mathbb{R}^{n_1 \times n_2}$ and data $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$. Let $\|\mathbf{L}\|_\star = \sum_{i=1}^{r} \sigma_i(\mathbf{L})$ is the nuclear norm and $\|\mathbf{S}\|_1 = \sum_{i,j} |S_{i,j}|$ is the ℓ_1-norm of matrix $\mathbf{S}$ seen as a vector. This optimization procedure guaranteed to work in most case. Although the solution is not beautiful as PCA since we have to use an optimization process, we have some efficient and scalable algorithms that can solve this problem with a reasonable cost compare with the standard PCA.

Separation of Low-rank and sparse component : although having a solution to the problem, there are something missing about the separation of low-rank and sparse component. The remaining question is how to identify the low-rank and the sparse components. In other words, it only makes sense when the role of each matrix is clear and the matrix $\mathbf{M}$ can perfectly separate as the low-rank and the sparse components. Therefore, the solution is only meaningful when the low-rank

component is **not** sparse, i.e. its singular vectors are reasonably spread out, and the sparse matrix $\mathbf{S}_0$ does **not** have low-rank property, i.e. having uniform distributed sparsity pattern. As shown in [17], PCP perfectly recovers the low-rank and the sparse components, if the two following condition satisfy:

(1) Rank of $\mathbf{L}_0$ not too large: $\leq O(\frac{n}{(logn)^2})$

(2) $\mathbf{S}_0$ is reasonably sparse: $\leq O(n^2)$ non-zero entries

Non-convex approaches

Non-convex regularization functions can be applied to both low-rank and sparse optimization problems. Using this idea, a general Robust PCA objective function is presented in [24]. The optimization problem is solved using the ADMM procedure. The method is called non-convex ADMM (**NCADMM**). However, the non-convex regularization function g was not explicitly defined since it was constructed from a generalization of a shrinkage operation (i.e. indirect approach). In this chapter, we formulated a solution of an explicit non-convex function ℓ_p-norm (i.e. direct approach). Both the solution and the non-convex penalty function can be written explicitly. In this way, we could build an online framework efficiently and our method could also be used to solve other non-convex penalty functions. Yang et al. [133] also adapted the ADMM on non-convex low-rank and sparse problems where the objective function can be nonconvex, nonsmooth, or both. A more general optimization problem was considered in [133] with different choice of inducing low-rank and sparsity but only the sparsity function is possibly non-convex. More recently, Tran et al. [121] developed a generic Gauss-Newton framework which uses the ADMM for solving a class of nonconvex optimization problems involving low-rank matrix variables. This framework can handle general smooth non-convex cost function via its surrogate.

Sun et al. [116] proposed to use the capped trace norm and the capped ℓ_1-norm as surrogates of the rank and the ℓ_0-norm in the RPCA problem. To solve this non-convex RPCA formulation, they presented two algorithms: a Difference of Convex functions (DC) based method and a greedy-based approach on sub-problems. Recently, Netrapalli et al. [98] presented a non-convex method for Robust PCA problem (**NRPCA**). In this approach, the low-rank matrix $\mathbf{L}$ and sparse matrix $\mathbf{S}$ were

obtained by alternating between the rank-k projection of the residuals $\mathbf{M} - \mathbf{S}$ and hard thresholding technique on $\mathbf{M} - \mathbf{L}$. This procedure runs until the matrix $\mathbf{L}$ reaches the target rank r or if the remaining part (i.e. singular values) has small norm, in other words, the desired low-rank matrix $\mathbf{L}$ is found. More recently, Yi et al. [135] proposed to reduce the computational complexity of a non-convex optimization approach from $O(r^2d^2\log(1/\epsilon))$ to $O(rd^2\log(1/\epsilon))$ for fully observed case, and no more than $O(r^4 d \log d \log(1/\epsilon))$, for the partially observed case (where r denoting rank and d is the dimension, $r < d$).

Lu et al. [85] proposed to solve the joint non-convex low-rank and sparse minimization problem, involving RPCA [17] and Low-Rank Representation (LRR) [81] problems, by using Iteratively Reweighted Least Squares (**IRLS**). The authors first demonstrated the use of IRLS on LRR problem. Subspace Segmentation via LRR (or LRR problem in short) aims at finding low-rank representations $\mathbf{Z}$ of a set of data vectors $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ drawn from a union of k subspaces such that $\mathbf{X} = \mathbf{XZ} + \mathbf{E}$. The coefficient matrix $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n]$, that encodes the pairwise affinities between data vectors and the data $\mathbf{X}$ itself is used as the "dictionary". The matrix $\mathbf{E}$ represents noisy, or even grossly corruption occurred in some data vectors. This formulation is a relaxed version (the equality constraint) of the non-convex LRR problem. The underlying assumption on sparse component is different from non-convex RPCA problem, i.e. $\mathbf{E}$ is "sample-specific" corruption ($\ell_{2,p}$-norm is used instead of ℓ_p-norm). Then, applying the **IRLS** algorithm solely to solve non-convex RPCA problem would be difficult since the problem involves both the ℓ_p-Schatten-norm and the ℓ_p-norm. Thus, it would be a non-trivial extension of the work in [85].

A Bayesian-based approach, named **MOG-RPCA**, is presented in [92] and [140] without explicitly forming ℓ_p-norm, but its Bayesian framework has certain properties of ℓ_p-norm.

Non-convex approximation can be applied on both components: low rank and sparse; a general objective function is given as [24].

$$\min_{\mathbf{L},\mathbf{S}} G_{\mu,p}(\sigma(\mathbf{L})) + \lambda G_{\mu,p}(\mathbf{S}), \quad \text{subject to } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{71}$$

where $\sigma(\mathbf{L})$ is the vector of singular values of $\mathbf{L}$.

Matrix Completion estimates missing values of a low-rank matrix from partial observations of

its entries. To recover a low-rank matrix, the key idea is to exploit its low-rank or approximately low-rank property by solving a matrix rank minimization problem. Given an incomplete matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, this problem can be formulated as follows:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \text{rank}(\mathbf{X}) \\ \text{s.t.} \quad & \mathbf{X}_{ij} = \mathbf{M}_{ij}, (i,j) \in \Omega \end{aligned} \tag{72}$$

where $\mathbf{X} \in \mathbb{R}^{m \times n}$ and Ω is the set of all entries (i, j) such that $M_{i,j}$ is known.

However, this rank minimization problem is NP-hard because of the non-convexity and the combinational nature of the rank function. Thus, it is hard to solve it directly and efficiently. Nuclear norm, i.e. the sum of singular values of a matrix, is the closest convex bound of the rank function of matrices [109]. Therefore, we can apply the nuclear norm as a convex surrogate of the non-convex matrix rank function which is similar to the case of ℓ_0-norm of vectors. Fazel [38] proposed to use nuclear norm to approximate the rank function in the rank minimization problem for control system. Candès et al. [14] presented a convex relaxation for Eqn. (72) which solves the following minimization problem:

$$\begin{aligned} \min_{\mathbf{X}} \quad & \|\mathbf{X}\|_* \\ \text{s.t.} \quad & P_\Omega(\mathbf{X}) = P_\Omega(\mathbf{M}) \end{aligned} \tag{73}$$

where $\|\cdot\|_*$ denotes the nuclear norm of a matrix. P_Ω is the orthogonal projector onto the span of matrices vanishing outside of Ω, in other word, the constraint only applies on non-missing entries i.e. $P_\Omega(\mathbf{X}) = \mathbf{X}_{ij}$ if $(i, j) \in \Omega$ and 0 otherwise.

A more general problem can be formulated as in Eqn. (74):

$$\min_{\mathbf{X}} \lambda \sum_{i=1}^{r} g(\sigma_i(\mathbf{X})) + f(\mathbf{X}) \tag{74}$$

where $\sigma(\mathbf{X})$ is the vector of singular values of $\mathbf{X} \in \mathbb{R}^{m \times n}$. Depending on the choice of the regularized function g and the constrained or loss function f, various types of the low-rank and sparse minimization problem can be formulated. For example, when the squared loss $f(\mathbf{X}) = \frac{1}{2}\|\mathcal{A}(\mathbf{X}) - \mathbf{b}\|_F^2$,

where $\mathbf{b} \in \mathbb{R}^n$ and $\mathcal{A}$ is a linear operator, and $g(\mathbf{x}) = \mathbf{x}$ as $\lambda \sum_{i=1}^{r} \sigma_i(\mathbf{X}) = \lambda \|\mathbf{X}\|_*$, then Eqn. (74) is the nuclear norm minimization problem $\min_{\mathbf{X}} \lambda \|\mathbf{X}\|_* + f(\mathbf{X})$. Lu et al. [84] presented the Iteratively Reweighted Nuclear Norm (IRNN) method to solve the Weighted Singular Value Thresholding (WSVT) problem. The penalty functions described in Table 2.1 were used to enhance low-rank matrix recovery.

Geng et al. [47] proposed a general matrix completion framework and applied difference of convex functions (DC) programming and DC Algorithm (DCA), a non-convex optimization algorithm, to recover effectively a corrupted image (up to 70 % missing entries). Hu et al. [63] employed the truncated nuclear norm to approximate the rank of matrix better. The truncated nuclear norm is given as the sum of the smallest $\min(m, n) - r$ singular values. In this way, r largest non-zero singluar values will not affect the rank of the matrix. The authors proposed to use different optimization algorithms to solve this truncated nuclear norm minimization problem including: ADMM, Accelerated Proximal Gradient Line (APGL) and ADMM with Adaptive Penalty (ADMMAP).

Online Approaches

An online Robust PCA method efficiently estimates the sparse and low-rank matrices in an incremental way. Thus, it has been employed in applications such as *background subtraction* and *subspace tracking*. In these applications, low-rank components are modeled as a low dimensional subspace that gradually changes over time. Although PCP was considered to be the state-of-the-art method for video background subtraction, it has some limitations, including a high computational cost, an offline processing with high memory demanding, and sensitivity to camera jitter. Some incremental algorithms have been proposed to address those issues in PCP: **ReProCS** [105] and its extensions [106], [56]. These methods reformulate PCP into a bilinear factorization form to find the low dimensional subspaces in the presence of sparse outliers. A similar approach was also developed by Mateos and Giannakis in [90]. Rodriguez et al. [110] proposed an incremental PCP algorithm for video background modeling that is robust to translational and rotational jitter.

Feng et al. [41] proposed an online optimization method **OR-PCA** for solving the convex robust PCA problem as in Eq. (2). The authors replaced the nuclear norm by an explicit factorization of the low-rank matrix $\mathbf{L}$ having a rank upper bounded by r as $\|\mathbf{L}\|_* = \inf\{\frac{1}{2}\|\mathbf{U}\|_F^2 + \frac{1}{2}\|\mathbf{V}\|_F^2 : \mathbf{L} = \mathbf{U}\mathbf{V}^\top\}$,

where $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{V} \in \mathbb{R}^{n \times r}$ denote the basis of the low-dimensional subspace and the coefficients of the samples w.r.t the basis. The paper does not mention how to find suitable value of r. This overcomes the difficulty of the nuclear norm in considering each sample separately as in typical online optimization problems. The problem (2) is then reformulated into the problem of learning the basis $\mathbf{U}$ and the representation coefficients $\mathbf{v}_i$ of each frame. Stochastic optimization algorithm was presented in [41] for solving this new problem which is quite similar to an online dictionary learning approach. Feng et al. [40] also proposed an online PCA aiming at finding sequentially Principal Components (PCs). However, this paper focuses on a totally different interpretation of PCA-related methods which is to find low-rank matrix decomposition instead. More recently, Lee et al. [78] proposed a projection based RPCA for online and real-time processing. The proposed online algorithm in this paper reduces computational complexity significantly, although the proposed algorithm has negligible performance degradation compared to conventional schemes. Hong et al. [61] proposed another online RPCA algorithm by using truncated nuclear norm as a tighter approximation of low rank constraint with an efficient online alternating optimization algorithm.

There are some works that extended OR-PCA [41] for background subtraction/foreground detection problem in various aspects such as adding continuous constraint Markov Random Field (MRF) [67], multi-feature based OR-PCA scheme [69] and integrating of depth and color information [68].

Another group of online RPCA approaches is based on subspace/manifold learning such as **GRASTA** [58], **GOSUS** [131] and **pROST** [57, 115]. They leverage the assumption of having the estimated signal lies on a Grassmannian, a manifold of fixed-dimensional subspace.

He et al. [58] proposed an incremental gradient descent method on Grassmannian manifold called Grassmanian Robust Adaptive Subspace Tracking Algorithm (GRASTA) to solve the RPCA problem in online manner. In its each iteration, GRASTA uses the gradient of the updated augmented Lagrangian function after revealing a new sample to perform the gradient descent. Results are encouraging for background modeling, but no theoretic guarantee of the algorithm convergence for GRASTA is provided and the output rank must be a known prior.

The above shows that tackling both non-convexity and incremental algorithms for solving the RPCA problem tends to be a potential direction with prominent results.

Since incremental algorithms, e.g. **GRASTA** [58], **GOSUS** [131], **OR-PCA** [41] and **pRost** [57, 115], are preferable to batch algorithms in some applications (e.g. video surveillance), this thesis proposes a novel real-time incremental ℓ_p-norm approach in addition to the offline approach. Currently, there is only few non-convex algorithm that can handle both incremental and real-time. Most of the online algorithms are not fast enough to analyze new coming large-scale data in real-time. Real-time implementation was made possible for those algorithms thanks to the parallel processing power of a graphics processing unit (GPU) but not due to an actual reduction of their complexities. The two approaches: offline and online are presented in section 3.2 and 3.3, respectively.

2.1.5 Singular Value Decomposition: A Review

The Singular Value Decomposition (SVD) has become one of the basic and most important tools of modern numerical analysis, particularly numerical linear algebra. It has underpinned numerous fundamental methods [71] such as Principal Component Analysis (PCA), Matrix Factorization, Orthogonal Procrustes Analysis, Correspondence Analysis, etc. In the SVD, given a matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, where m represents the number of variables and n denotes the number of instances, the decomposition matrices can be broken up into three components:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \tag{75}$$

where the left singular vectors $\mathbf{U}_i \in \mathbb{R}^m$ and the right singular vectors $\mathbf{V}_i \in \mathbb{R}^n$ $(i = 1, \cdots, r)$ are orthonormal. Each has a unit length and every pair is orthogonal, i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$. r denotes the rank of $\mathbf{X}$, where $r \leq \min(m, n)$. $\mathbf{\Sigma} \in \mathbb{R}^{r \times r}$ is a diagonal matrix containing the square root of the eigenvalues from $\mathbf{U}$ or $\mathbf{V}$ in descending order. The problem defined in Eqn. (75) is equivalent to the minimization of the cost function ε as follows:

$$\varepsilon(\mathbf{U}, \mathbf{V}) = \|\mathbf{X}_{m \times n} - \mathbf{U}_{m \times r}\mathbf{\Sigma}_{r \times r}\mathbf{V}^\top_{n \times r})\|_2^2 = \sum_{i=1}^{m}\sum_{j=1}^{n}(\mathbf{x}_{i,j} - \sigma\mathbf{u}_i\mathbf{v}_j^\top)^2 \tag{76}$$

where the matrix $\mathbf{X}_{m \times n}$ is defined as in Eqn. (75), σ is the singular value vector, and $\mathbf{u}_i, \mathbf{v}_j$ are the columns of the orthonormal matrices $\mathbf{U}$ and $\mathbf{V}$, respectively. The SVD problem can be simply

solved in a regular closed form using a ℓ_2-norm cost function. The ℓ_2-norm process however treats all input data equally and doesn't have ability to detect outliers or sparse components. Therefore, SVD subspaces are sensitive to outliers and noisy values from given input data. Fig. 2.3 shows an example of the limitations in SVD and other previous SVD extensions. When input data is free of noise or outliers, SVD can generate a good subspace to represent the data distribution. However, when the data contains some noise or outliers, this subspace contains a structure distortion; hence it doesn't represent well the data distribution. In addition, there is no mechanism to deal with missing values in the regular SVD representation. The decomposed matrix $\mathbf{X}$ must be completely filled with values for all $d \times n$ items; otherwise the problem is unsolvable. The SVD was established for real square matrices in the 1870's by Beltrami and Jordan and for general rectangular matrices by Eckart and Young [71]. In this section, we review recent SVD studies. They can be divided into two categories, i.e. batch and the incremental approaches.

Batch (Offline) SVD

Huang et al. [65] proposed a regularized SVD (RSVD) for dimension reduction and feature extraction. RSVD was posed as a low-rank matrix approximation problem with a squared loss function on reconstruction errors and a quadratic penalty on the factorized solutions. However, RSVD is also sensitive to outliers as showed in Fig. 2.3. Liu et al [83] presented a robust SVD (ROBSVD) that can cope with outliers and impute missing values for microarray data. Bai et al. [3] proposed

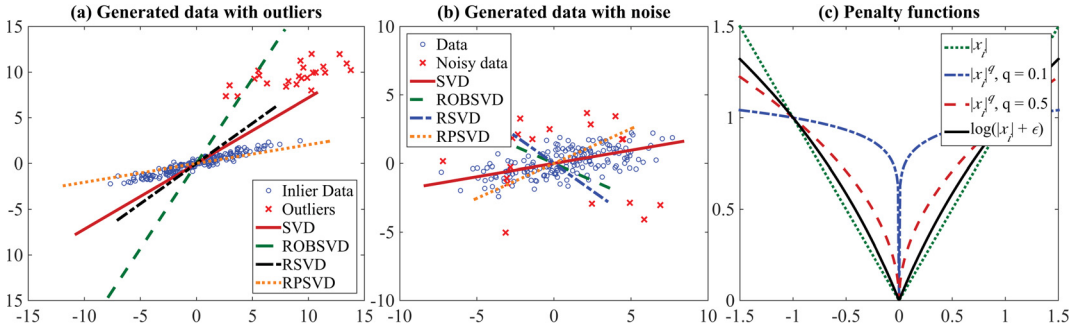

Figure 2.3: (a) and (b) show principal directions obtained by using SVD, ROBSVD [83], RSVD [65], and our proposed RP-SVD on the toy data set with outliers and noise. (c) Illustration of common convex and non-convex regularized functions.

a supervised SVD (SSVD), less sensitive to outliers, to improve the robustness of analyzing functional Magnetic Resonance Imaging (fMRI) brain images. They proposed to supervise SVD by imposing subspace constraints to find the best low-rank approximation. SSVD can be incorporated into Independent Component Analysis (ICA) for dimension reduction to explore spatio-temporal features in fMRI data. Zhang et al. [137] developed a robust regularized SVD (ROBRSVD) method to lessen the effects of outliers. The authors proposed to solve the following problem:

$$\min_{\mathbf{u},\mathbf{v}}\{\rho(\mathbf{X}-\mathbf{u}\mathbf{v}^\top)+\mathcal{P}_\lambda(\mathbf{u},\mathbf{v})\} \tag{77}$$

where $\mathbf{X}\in\mathbb{R}^{m\times n}$ is the data matrix, $\mathbf{u}$ and $\mathbf{v}$ are m-dimensional and n-dimensional vectors respectively. $\rho(\cdot)$ is a robust loss function, $\mathcal{P}_\lambda(\mathbf{u},\mathbf{v})$ is a two-way roughness penalty to ensure smoothness for $\mathbf{u}$ and $\mathbf{v}$, and λ is a vector of penalty parameters. This formulation is a generalized version of RSVD and robust SVD. In other words, ROBRSVD is a robustified RSVD method using a robust loss function instead of the non-robust squared-error loss as in [65]. It can also be considered as smoothing of a robust SVD [83] method with the penalty term in Eqn. (77). Zhang et al. suggested to iteratively impute the missing values by replacing it with values from the previous iteration, then applying the iterative reweight least square (IRLS) algorithm to solve the problem in Eqn. (77). Table 2.2 summarizes the properties of the above mentioned methods.

Incremental (Online) SVD

In some scenarios, due to the availability of data, the SVD of a data matrix must be updated as new columns of the matrix become available. This has given rise to a class of incremental methods. The goal of incremental methods is to compute the SVD of the matrix $\mathbf{X}_{\text{new}}=[\mathbf{X}\ \ \mathbf{C}]$ by updating the current SVD of the matrix $\mathbf{X}$ using the new columns $\mathbf{C}$. These methods should update the SVD in a more efficient manner so that the computational cost over all columns of the matrix may be lower than that of the batch methods.

Similar to the batch methods, in numerous applications, only the dominant singular vectors corresponding the largest singular values of a matrix are needed. Thus, the incremental methods may

Table 2.2: Comparing the properties between our proposed RP-SVD and ORP-SVD approaches and other state-of-the-art SVD methods, where ✗ denotes unknown or not directly applicable properties.

	RP-SVD	**ORP-SVD**	SVD	RSVD [65]	ROBSVD [83]	ROBRSVD [137]
Non-Convexity						
Loss-function	✓	✓	✗	✗	✓	✓
Penalty function	✓	✓	✗	✗	✗	✗
Robustness						
Outliers	✓	✓	✗	✗	✓	✓
Missing values	✓	✓	✗	✗	✓	✓
Scalability						
Online	✗	✓	✗	✗	✗	✗
Real-time	✗	✓	✗	✗	✗	✗

produce a truncated SVD of the matrix instead of a full-rank SVD. This group of incremental methods is called low-rank incremental SVD methods which relax the conventional full-rank incremental approach. The generic algorithm of the low-rank incremental SVD consists of two main steps: (1) from a rank-k approximation $\mathbf{X} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ and new columns $\mathbf{C}$, perform updating the SVD of $[\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \ \mathbf{C}]$; (2) keep only the rank-k dominant part $\mathbf{U}'\mathbf{\Sigma}'\mathbf{V}'^\top \approx [\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \ \mathbf{C}]$. There are several implementations of a low-rank incremental SVD with various updating steps [4, 10, 11, 18, 19, 20, 79].

2.2 Deep Learning

This section will briefly introduce the Boltzmann Machines and related methods.

2.2.1 From Energy-Based Models (EBM) to Restricted Boltzmann Machines (RBM)

Energy-based models assign an energy value to each configuration of the variables of interest. Model learning is to adjust that energy function to have desirable properties, such as having low energy for desirable configurations. Energy-based probabilistic models define a probability distribution via an energy function E, as follows:

$$p(x) = \frac{e^{-E(x)}}{\mathbf{Z}} \tag{78}$$

where $\mathbf{Z} = \sum_x e^{-E(x)}$ is the partition function.

Several training techniques for an energy-based model have been proposed in literature. A classical and widely used technique is the Maximum Likelihood Estimation (MLE) via (stochastic) gradient descent. Given a set of observed data $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ which is assumed to be independently and identically distributed (i.i.d) and a set of model parameters $\theta = \{\theta_1, \cdots, \theta_M\}$, the MLE approach finds the optimal θ by maximizing the log likelihood $\log p(\mathbf{X}|\theta)$.

$$\theta^\star = \arg\max_\theta \frac{1}{N} \sum_{x_i \in \mathbf{X}} \log\ p(x_i|\theta) \tag{79}$$

To increase the expressive power of the model, we may want to add some non-observed variables on top of the observed variables x. So we consider an observed part x and a hidden part h. We can then write:

$$p(x) = \sum_h p(x,h) = \sum_h \frac{e^{-E(x,h)}}{Z} \tag{80}$$

Using similar formulation as in Eq. (78), the notation of free energy is defined as follows:

$$\mathcal{F}(x) = -\log \sum_h e^{-E(x,h)} \tag{81}$$

This allows us to re-write, $p(x) = \frac{e^{-\mathcal{F}(x)}}{Z}$ with $Z = \sum_x e^{-\mathcal{F}(x)}$.

The gradient w.r.t each θ_m is given by

$$\frac{\partial \log p(x|\theta_1, \cdots, \theta_M)}{\partial \theta_m} \quad = \frac{\partial \mathcal{F}(x)}{\partial \theta_m} - \sum_{\tilde{x}} p(\tilde{x})\ \frac{\partial \mathcal{F}(\tilde{x})}{\partial \theta_m} \tag{82}$$

We refer the two terms in the above gradient as the positive and negative phase. The name of the terms reflects their effect (positive or negative) on the probability density defined by the model. Positive effect means that the probability of training data increases (as the corresponding free energy reduces), while negative effect indicates that the probability of samples generated by the model decreases. When the dimension of data becomes increasingly high, the second term is analytically infeasible to compute due to the exponential possible configurations. Therefore, it needs to be approximated by using a fixed number of samples, called negative particles (denoted as

$\mathcal{N}$). Then, the gradient can be rewritten as:

$$\frac{\partial \log p(x|\theta_1, \cdots, \theta_M)}{\partial \theta_m} \approx \frac{\partial \mathcal{F}(x)}{\partial \theta_m} - \frac{1}{|\mathcal{N}|} \sum_{\tilde{x} \in \mathcal{N}} \frac{\partial \mathcal{F}(\tilde{x})}{\partial \theta_m}. \tag{83}$$

where the elements $\tilde{x}$ of $\mathcal{N}$ should be sampled according to p. To sample these negative particles $\mathcal{N}$, we use sampling methods, e.g. Markov Chain Monte Carlo (MCMC) methods, which are especially well suited for models such as the Restricted Boltzmann Machines (RBM), a specific type of EBM.

2.2.2 RBM and Its Extensions

Boltzmann Machines (BM) [60] are an undirected graphical model with two layers of stochastic units, i.e. visible units **v** and hidden units **h**, which represent the observed data and the conditional representation of that data, respectively. All the units are connected by weighted undirected edges to interpret the pairwise constraints between them. This makes them powerful enough to represent complicated distributions. We can increase the modeling capacity of the BM by having more hidden units. The BM is actually an energy-based model which defines the joint probability distribution using an energy function. The energy function of the BM is given by

$$E_{BM}(\mathbf{v}, \mathbf{h}) = -\sum_i b_i v_i - \sum_j c_j h_j - \sum_{i,j} W_{i,j} v_i h_j - \sum_{j,j'} U_{j,j'} h_j h'_j - \sum_{i,i'} V_{i,i'} v_i v'_i \tag{84}$$

Maximum likelihood are usually used to learn BM. Due to an intractable partition function in BM, the maximum likelihood gradient must be approximated using the Monte Carlo methods.

Restricted Boltzmann Machines (RBM) [59] is a simplified version of BM without visible-to-visible and hidden-to-hidden connections. Similar to BM, the joint probability distribution of RBM is specified by its energy function:

$$P(\mathbf{v} = v, \mathbf{h} = h) = \frac{1}{Z} \exp(-E(v, h)) \tag{85}$$

where the energy function of RBM is defined as

$$E_{RBM}(\mathbf{v}, \mathbf{h}) = -\sum_i b_i v_i - \sum_j c_j h_j - \sum_{i,j} W_{i,j} v_i h_j \tag{86}$$

and Z is the normalizing constant also known as the partition function $Z = \sum_v \sum_h \exp\{-E(v, h)\}$.

Thanks to the specific structure of RBMs, the hidden units are conditionally independent given the states of visible units. Using this property, we can write conditional probability as:

$$\begin{aligned} p(\mathbf{h}|\mathbf{v}) &= \prod_j p(h_j|\mathbf{v}) \\ p(\mathbf{v}|\mathbf{h}) &= \prod_i p(v_i|\mathbf{h}). \end{aligned} \tag{87}$$

Using binary units (where v_i and $h_j \in \{0, 1\}$), we obtain from Eqns. (80) and (86) as follows.

$$\begin{aligned} p(h_j = 1|\mathbf{v}) &= \sigma(a_j + \sum_i W_{i,j} v_i) \\ p(v_i = 1|\mathbf{h}) &= \sigma(b_i + \sum_j W_{i,j} h_j) \end{aligned} \tag{88}$$

The partial derivative of the energy function w.r.t the model parameters $\theta = \{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ is given by

$$\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \mathbf{W}} = -\mathbf{v}\mathbf{h}^T \tag{89}$$

$$\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \mathbf{a}} = -\mathbf{h} \tag{90}$$

$$\frac{\partial E(\mathbf{v}, \mathbf{h})}{\partial \mathbf{b}} = -\mathbf{v} \tag{91}$$

Other Types of Restricted Boltzmann Machines

A set of BMs can be organized in several layers such that each BM is stacked on top of another to capture more complicated correlations between features in the lower layer. This approach produces a deeper network called **Deep Boltzmann Machines** (DBM) [114]. Since all connections

between units in two consecutive layers are undirected, each unit receives both bottom-up and top-down information such that it better propagates uncertainty during the inference process. The joint probability of a deep Boltzmann machine with one visible layer, $\mathbf{v}$, and two hidden layers, $\mathbf{h}^1, \mathbf{h}^2, \mathbf{h}^3$ is given by:

$$P(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2) = \frac{1}{\theta} \exp(-E(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2; \theta)) \tag{92}$$

and the energy function of DBM is defined as (the bias parameters are ignored for simplicity):

$$E_{DBM}(\mathbf{v}, \mathbf{h}^1, \mathbf{h}^2) = -\sum_{i,j} W^1_{i,j} v_i h^1_j - \sum_{j,k} W^2_{j,k} h^1_j h^2_k \tag{93}$$

Instead of using the visible binary units as in the RBM, **Gaussian RBM** (GRBM) [75] models real-valued data by assuming the visible units have real values normally distributed with mean b_i and variance σ_i^2. Its energy function is defined as:

$$E_{GRBM}(\mathbf{v}, \mathbf{h}) = -\frac{1}{2} \sum_i \frac{(v_i - b_i)^2}{\sigma^2} - \sum_j c_j h_j - \sum_{i,j} W_{i,j} v_i h_j \tag{94}$$

Denoising Gated Boltzmann Machines (DGBM) [118] and RoBM [119] were proposed to estimate noise and learn features simultaneously by distinguishing corrupted and uncorrupted pixels to find optimal latent representations. The energy function of RoBM is a combination of a binary RBM, a GRBM, a Gaussian noise model and gating terms:

$$\begin{aligned} E_{RoBM}(\mathbf{v}, \tilde{\mathbf{v}}, \mathbf{s}, \mathbf{h}, \mathbf{g}) = &\frac{1}{2} \sum_i \frac{\gamma_i^2}{\sigma_i^2} s_i (v_i - \tilde{v}_i)^2 - \sum_i d_i s_i - \sum_k e_k g_k - \sum_{i,k} U_{i,k} s_i g_k \\ &+ \frac{1}{2} \sum_i \frac{(v_i - b_i)^2}{\sigma_i^2} - \sum_j c_j h_j - \sum_{i,j} W_{ij} v_i h_j + \frac{1}{2} \sum_i \frac{(\tilde{v}_i - \tilde{b}_i)}{\tilde{\sigma}_i^2} \end{aligned} \tag{95}$$

2.2.3 Sampling in RBM via Monte-Carlo Markov Chain (MCMC)

As mentioned in Section 2.2.1, we run a sampling Markov chain converging to the target distribution to obtain samples of $p(x)$. A sampling technique, i.e. Gibbs sampling, is done on the joint of N random variables $\mathbf{S} = \{s_1, ..., s_N\}$ by performing a sequence of N sub-sampling steps of the form $s_i \sim p(s_i | s_{-i})$ where s_{-i} contains the $N - 1$ other random variables in $\mathbf{S}$ excluding s_i.

In the setting of RBMs, **S** will be the set of visible and hidden units. One can perform block Gibbs sampling such that visible units are sampled simultaneously given fixed values of the hidden units and vice versa. Thus, a step in the Markov chain is taken as follows:

$$\mathbf{h}^{(n+1)} \sim \sigma(\mathbf{W}\mathbf{v}^{(n)} + \mathbf{a}) \tag{96}$$

$$\mathbf{v}^{(n+1)} \sim \sigma(\mathbf{W}\mathbf{h}^{(n+1)} + \mathbf{b}), \tag{97}$$

where $\mathbf{v}^{(n)}$ and $\mathbf{h}^{(n)}$ denote the set of all visible and hidden units at the n-th step of the Markov chain, respectively. In other words, $h_j^{(n+1)}$ is randomly sample to be 1 or 0 with probability of $\sigma(a_j + \sum_i W_{i,j} v_i^{(n)})$, and similarly, $v_i^{(n+1)}$ is randomly sample to be 1or 0 with probability of $\sigma(b_i + \sum_j W_{i,j} h_j^{(n)})$.

The Gibbs chain for k steps is illustrated in Fig. 2.4.

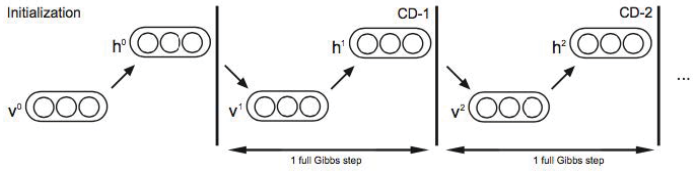

Figure 2.4: Gibbs sampling chain

As $k \to \infty$, samples of $(\mathbf{v}^{(t)}, \mathbf{h}^{(t)})$ are guaranteed to be accurate samples from target distribute $p(\mathbf{v}, \mathbf{h})$. In theory, running one such chain to convergence for each parameter update in the learning process would take very long time. Therefore, several algorithms have been developed for learning RBMs, in order to efficiently sample from $p(\mathbf{v}, \mathbf{h})$ during the learning process.

2.2.4 Constrastive Divergence (CD-k)

Due to the problem of evaluating the partition function, Contrastive Divergence proposed by Hinton [59] provides another way to estimate the gradient of the energy function without the need to reach the equilibrium distribution. The main ideas of this technique to speed up the sampling process are summarized as follows:

- Initialize the Markov chain with a training example, so that the chain will be already close to having converged to its final distribution p

- Run the Gibbs sampling for only k-steps (we have CD-k). In practice, $k = 1$ has been shown to work surprisingly well.

2.3 Conclusions

This chapter provides an overview of the recent studies related to matrix decomposition and matrix factorization and the ideas behind sparse coding leading to overcomplete dictionaries. ℓ_p-norm has been extensively used and analyzed for sparse regularized optimization methods. ℓ_p-norm has desirable properties and supporting theories to be a suitable surrogate of ℓ_0-norm. Thus, using this ℓ_p-norm approach, we can improve the performance of many problems involving sparsity and/or low-rank regularization in their objective function. However, the existing work along this direction is in their early stage since the non-convexity of the ℓ_p-norm makes it difficult to optimize directly and efficiently. Therefore, the aims of this thesis is to incorporate ℓ_p-norm regularization into two well-known problems: matrix decomposition and matrix factorization and to solve these problems efficiently on large-scale datasets. In addition, an overview of deep learning approach for face modeling focusing on RBMs is briefly introduced to give a better connection with conventional approach to matrix decomposition and matrix factorization problems. The next two chapters will present our proposed ℓ_p-norm based approach for the matrix decomposition and factorization problems; and our proposed deep learning based approach, respectively.

Chapter 3

Matrix Decomposition and Factorization: Conventional Approaches

This chapter will first present a robust face recognition (FR) framework using Robust Principal Component Analysis and Sparse Representation. Then ℓ_p Robust Principal Component Analysis approach, for matrix decomposition problem and ℓ_p Singular Value Decomposition approach for matrix factorization problem are proposed and introduced in details.

3.1 Robust Principal Component Analysis: Low-rank and Sparse Representation for Robust Face Recognition

First, our method will eliminate occlusions or corruption from face images in the training set. For face images of p subjects, we form the matrix $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_p]$ where the training data matrix $\mathbf{D}_i$ contains multi-factor face images of subject i. We then apply Low-rank (LR) matrix decomposition [17] to obtain the LR components $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_p]$ and sparse components $\mathbf{E} = [\mathbf{E}_1, \mathbf{E}_2, \cdots, \mathbf{E}_p]$ (See Fig. 3.1). Where $\mathbf{A}$ and $\mathbf{E}$ are obtained from the following minimization problem.

$$\min_{\mathbf{A}_i, \mathbf{E}_i} \|\mathbf{A}_i\|_* + \lambda \|\mathbf{E}_i\|_1 \quad s.t \quad \mathbf{D}_i = \mathbf{A}_i + \mathbf{E}_i \tag{98}$$

where $\|\cdot\|_*$ denotes the nuclear norm (the sum of singular values of a matrix). $\|\cdot\|_1$ denotes the sum of absolute value of matrix or vector entries.

We apply inexact Augmented Lagrange multipliers (ALM) [17] [80] to solve LR decomposition for each class iteratively. In ALM, the augmented Lagrangian function is defined as following:

$$\mathcal{L}(\mathbf{A}_i,\mathbf{E}_i,\mathbf{Y}_i,\mu) = \|\mathbf{A}_i\|_* + \lambda\|\mathbf{E}_i\|_1 + \langle Y_i, \mathbf{D}_i - \mathbf{A}_i - \mathbf{E}_i\rangle + \frac{\mu}{2}\|\mathbf{D}_i - \mathbf{A}_i - \mathbf{E}_i\|_F^2 \tag{99}$$

where μ is a positive penalty constant, Y_i is a Lagrange multiplier vector, and $\langle \mathbf{A}, \mathbf{B}\rangle = trace(\mathbf{A}^T\mathbf{B})$. The details of the inexact ALM algorithm are shown in Algorithm 1.

Algorithm 1 Solve LR decomposition by inexact ALM [80]

1. **Input:** Training data matrix $\mathbf{D}$ and parameter λ. Initialize $\mathbf{Y}^0$, $\mathbf{E}^0 = 0$, $\mu_0 > 0$, $\rho > 1$ and $k = 0$
2. **for** $i = 1$ **to** p **do**
3. **while** not converged **do**
   // Update $\mathbf{A}_i$
   // by solving $\mathbf{A}_i^{k+1} = \arg\min_{\mathbf{A}} \mathcal{L}(\mathbf{A}_i, \mathbf{E}_i^k, \mathbf{Y}_i^k, \mu_k)$
   $(U, S, V) = svd(\mathbf{D}_i - \mathbf{E}_i^k + \mu_k^{-1}\mathbf{Y}_i^k)$;
   $\mathbf{A}_i^{k+1} = \mathbf{U}\mathcal{S}_{\mu_k^{-1}}[\mathbf{S}]\mathbf{V}^T$; // $\mathcal{S}_\epsilon[x] = sign(x)(|x| - \epsilon)$
   // Update $\mathbf{E}_i$
   //by solving $\mathbf{E}_i^{k+1} = \arg\min_{\mathbf{E}} \mathcal{L}(\mathbf{A}_i^{k+1}, \mathbf{E}_i, \mathbf{Y}_i^k, \mu_k)$
   $\mathbf{E}_i^{k+1} = \mathcal{S}_{\lambda\mu_k^{-1}}[\mathbf{D}_i - \mathbf{A}_i^{k+1} + \mu_k^{-1}\mathbf{Y}_i^k]$;
   // Update multiplier $\mathbf{Y}_i$
   $\mathbf{Y}_i^{k+1} = \mathbf{Y}_i^k + \mu_k(\mathbf{D}_i - \mathbf{A}_i^{k+1} - \mathbf{E}_i^{k+1})$;
   // Update μ
   $\mu_{k+1} = \rho\mu_k$
   $k = k + 1$
   end while
   end for
4. **Output:** $\mathbf{A}$ and $\mathbf{E}$

The LR components contain the most common information among all faces of a person while the sparse components store the variations or occlusion across faces of each subject as shown in Fig. 3.1 (b) and (c). As a result, we eliminate the affecting factors in training images to have the LR components $\mathbf{A}$ with better representation ability. However, we cannot apply LR directly to remove those affecting factors from a test image in a similar way since it requires many images to form the

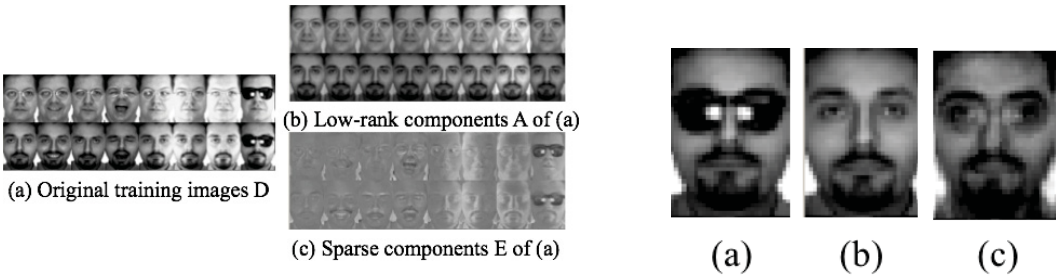

Figure 3.1: **LEFT**: An example of Robust PCA for two subjects, **RIGHT**: Result of recovering step (a) original testing image (b) neutral image in the training set (c) normalized testing images

data matrix $\mathbf{D}$. Therefore, we use the SRC method [130] to represent a test image as follow:

$$\mathrm{y} = \mathrm{y}_0 + \mathrm{e}_0 = [\mathbf{A}, \mathbf{E}] \begin{bmatrix} \alpha \\ \alpha_{\mathbf{E}} \end{bmatrix} \tag{100}$$

where $\mathrm{y} \in \mathbb{R}^{m \times 1}$ is the original testing image, y_0 is the normalized testing image, e_0 is the error or occlusions. $\mathbf{A}$ and $\mathbf{E}$ are the sample dictionary and the occlusion dictionary respectively. α and $\alpha_{\mathbf{E}}$ are the sparse coefficients corresponding to the two dictionaries $\mathbf{A}$ and $\mathbf{E}$.

As we mentioned above, it is computationally expensive and ineffective if we simply use training samples as the sample dictionary and an identity matrix as the occlusion dictionary since they are not optimized in terms of size and representation ability. As a result, we suggest that a better version for the sample and the occlusion dictionary can be learned by dictionary learning technique [134] [102]. As an example, given a set of images $\mathbf{X} = [\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_n] \in \mathbb{R}^{m \times n}$ where the i-th image is represented by an m-dimensional vector x_i, the goal of dictionary learning is to find a dictionary $\mathcal{D} = [\mathrm{d}_1, \mathrm{d}_2, \cdots, \mathrm{d}_k] \in \mathbb{R}^{m \times k}$ such that each image can be represented as a sparse linear combination of its atoms i.e. $\mathrm{x}_i = \mathcal{D}\alpha_i$, where α_i is the sparse coefficients of the image x_i. This can be done using the following formulation:

$$\begin{aligned} \{\hat{\mathcal{D}}, \hat{\Lambda}\} = \arg\min_{\mathcal{D},\Lambda} \|\mathbf{X} - \mathcal{D}\Lambda\|_F^2 + \lambda\|\Lambda\|_1 \\ s.t. \quad \mathrm{d}_j^T \mathrm{d}_j = 1, \forall j \end{aligned} \tag{101}$$

where $\Lambda = [\alpha_1, \alpha_2, \cdots, \alpha_k]^T \in \mathbb{R}^{k \times n} (k \leq n)$ and λ is the regularization parameter. $\|\cdot\|_F$ denotes the Frobenius norm (l_2-norm of a matrix).

We cannot solve the above optimization problem simply using the K-SVD algorithm [1] since we need to optimize both the dictionary $\mathcal{D}$ and the representation matrix Λ. Similar to many multi-variable optimization problems, we solve the above problem by optimizing $\mathcal{D}$ and Λ alternatively. The optimization procedures are described in Algorithm 2. We suggest that the sample dictionary $\mathbf{A}$ and the occlusion dictionary $\mathbf{E}$ should be built respectively from the LR components $\mathbf{A}$ and the sparse components $\mathbf{E}$ in the previous step. In this way, we can preserve structural information in those components and improve the discrimination ability of the dictionaries as well. Since we work on raw pixels, the feature dimension is usually large (e.g 165 $\times$ 120 = 19800). PCA subspace learning is applied to reduce the feature dimension. This will greatly improve the performance of our method. PCA subspace is usually learned from training data matrix $\mathcal{D}$, however, we realized this may not be efficient and robust when training data contains occlusion or corruptions. Therefore, we learn PCA subspace from low-rank matrix $\mathbf{A}$ instead. In this way, it will reduce the effects caused by occlusion or corruptions since PCA is often sensitive to noise and outliers. Two learned dictionaries and testing images are then projected onto this reduced dimension subspace.

Algorithm 2 Algorithm for dictionary learning [134]

1. **Input:** Image data matrix $\mathbf{X}$ and parameter λ
2. **Step 1:** Initialize $\mathcal{D}$ randomly with unit l_2-norm
for each column of $\mathcal{D}$
3. **Step 2:** Fix $\mathcal{D}$ and solve Λ
Solve the following minization problem using convex optimization technique described in [73]
$$J_\Lambda = \arg\min_\Lambda \{\|\mathbf{X} - \mathcal{D}\Lambda\|_F^2 + \lambda\|\Lambda\|_1\}$$
4. **Step 3:** Fix Λ and update $\mathcal{D}$
We update d_j one by one while fixing all the other columns of $\mathcal{D}$, i.e. $\mathrm{d}_l, l \neq j$. We can find the update by optimizing the following problem.
$$J_\mathcal{D} = \arg\min_\mathcal{D} \|\mathbf{X} - \mathcal{D}\Lambda\|_F^2 \;\; s.t. \;\; \mathrm{d}_j^T \mathrm{d}_j = 1, \forall j$$
We use Lagrange multiplier $\mathbf{Y}$ to convert the objective function. After that differentiating J_{d_j} w.r.t. d_j, and set it to 0. We have
$$\mathrm{d}_j = \mathbf{Y}\alpha_j^T(\alpha_j\alpha_j^T - \lambda)^{-1}$$
$$\mathrm{d}_j = \mathbf{Y}\alpha_j^T / \|\mathbf{Y}\alpha_j^T\|_2$$
5. **Step 4:** Go back to step 2 until the values of $J_\mathcal{D}$ and J_Λ are converged or the maximum number of iterations is reached. Finally, output $\mathcal{D}$.
6. **Output:** $\mathcal{D}$

Finally, we can remove the affecting factors (e.g. occlusion, illumination and expression) using the learned sample and occlusion dictionaries (See Fig. 3.1). The reason why we need to eliminate

these variations is to ensure that testing images are not too different from training model or training images. In this way, they are normalized and they will not become outliers, thus it will enhance the recognition rate.

The LR components of the testing image are obtained via the following minimization problem:

$$\{\hat{\Delta}, \hat{\Gamma}\} = \arg\min_{\Delta,\Gamma} \|\mathrm{y} - \mathbf{A}\Delta - \mathbf{E}\Gamma\|_2^2 + \lambda_1\|\Delta\|_1 + \lambda_2\|\Gamma\|_1 \tag{102}$$

where $\mathbf{A} \in \mathbb{R}^{m\times(k\times p)}$ and $\mathbf{E} \in \mathbb{R}^{m\times l}$ (noted that two dictionaries $\mathbf{A}$ and $\mathbf{E}$ have different size $k \neq l$). $\Delta = [\boldsymbol{\beta}_1; \boldsymbol{\beta}_2; \cdots; \boldsymbol{\beta}_p]^T \in \mathbb{R}^{(k\times p)\times 1}$ with $\boldsymbol{\beta}_i \in \mathbb{R}^{k\times 1}$. Each β_i is the sparse coefficients associated with subject i. $\Gamma = [\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \cdots, \boldsymbol{\gamma}_l]^T \in \mathbb{R}^{l\times 1}$ are the best representation for occlusion or variations in the testing image.

Eqn. (102) can be solved by l_1-minimization algorithms such as Homotopy method [132]. After obtaining sparse representation of the testing image, the normalized testing image is recovered by $\hat{\mathrm{y}_0} = \mathrm{y} - \mathbf{E}\hat{\Gamma}$ and classification is based on SRC approach by computing the residuals for each subject.

$$e_i(\mathrm{y}) = \|\hat{\mathrm{y}}_0 - \mathbf{A}\boldsymbol{\delta}_i(\hat{\Delta})\|_2, \quad for\ i = 1, \cdots, N \tag{103}$$

where $\boldsymbol{\delta}_i(\hat{\Delta}) = [0 \cdots 0 \cdots \boldsymbol{\beta}_i \cdots 0 \cdots 0]^T$ with $\boldsymbol{\beta}_j = 0; \forall j \neq i$ Then, $identity(\mathrm{y}) = \arg\min e_i(\mathrm{y})$. The testing image y is represented by a linear combination of all images in the dictionary $\mathbf{A}$. The purpose of computing residuals is to find the subject i having the most influence in the sparse representation. The smaller the residual the more influence subject i has on the outcome, so it is more likely that y has identity of subject i. Moreover, Sparsity Concentration Index (SCI) was proposed in [130] to identify the quality of test samples. The SCI of a coefficient vector $\Delta \in \mathbb{R}^{k\times p}$ is defined as

$$SCI(\Delta) = \frac{\frac{p.\max\|\boldsymbol{\delta}_i(\Delta)\|_1}{\|\Delta\|_1} - 1}{p - 1} \tag{104}$$

SCI has values from 0 to 1. The test image with a SCI value close to 1 can be represented by using only dictionary atoms from a person. This gives us a different way to identify the label of the testing image y by using l_1-norm.

$$identity(\mathrm{y}) = \arg\max_i \|\boldsymbol{\delta}_i(\Delta)\|_1 \tag{105}$$

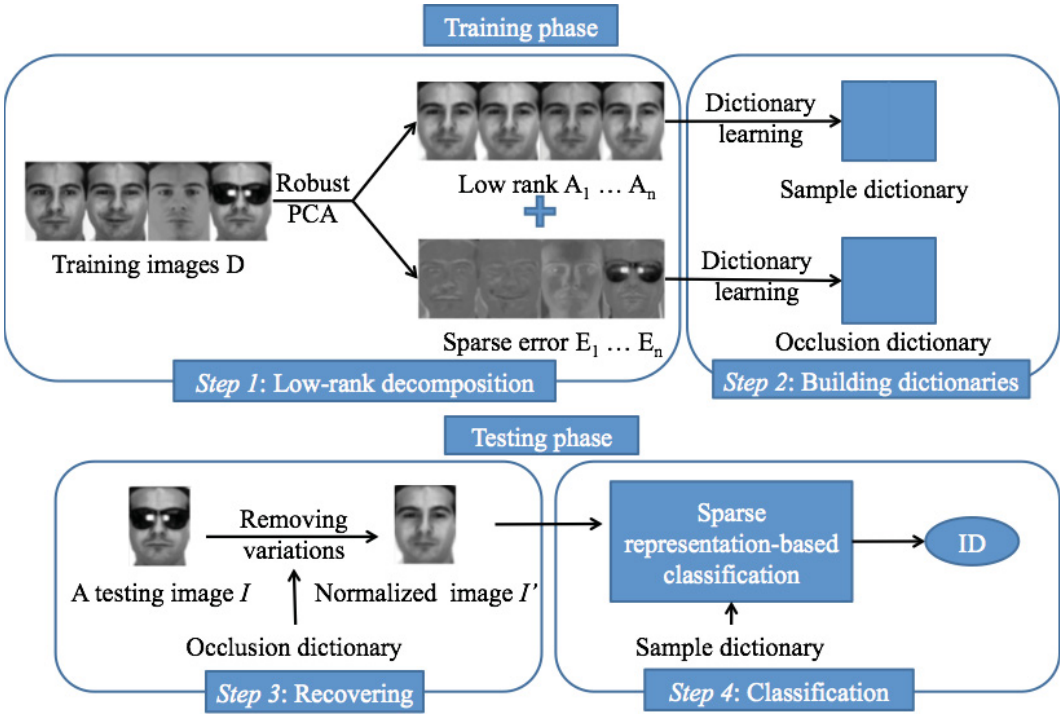

Figure 3.2: Steps in the training and testing phases

The identity of the test image y is considered as the person which has the highest value of the coefficients associated with. Because of the fact that the coefficents Δ will have high values associated with the atoms of $\mathbf{A}$ belonging to a person. In this way, we do not need to reconstruct the test image from its corresponding sparse coefficients, thus it is more efficient than the common way done in SRC. Dimension reduction using PCA is applied on learned dictionaries and testing images to reduce computational cost. PCA bases are learned from the low-rank matrix $\mathbf{A}$ rather than the data matrix $\mathbf{D}$ since $\mathbf{A}$ has less noise or corruptions than $\mathbf{D}$.

In general, the training phase can be summarized into two main steps: *low-rank decomposition* and *building dictionaries*. The testing phase can be summarized into two main steps: *recovering* and *classification*. The method described in this section is quite different from the approaches in [32] and [139] as shown in Fig. 3.2. Details of the training and testing algorithms are shown as follows.

Algorithm 3 Training phase

1. **Input:** Training data $\mathbf{D} = [\mathbf{D}_1, \mathbf{D}_2, \cdots, \mathbf{D}_p]$ from p subjects
2. **Step 1:** Perform low-rank decomposition on $\mathbf{D}$
   for $i = 1$ to p **do**
   $\min_{\mathbf{A}_i, \mathbf{E}_i} \|\mathbf{A}_i\|_* + \lambda\|\mathbf{E}_i\|_1 \quad s.t \quad \mathbf{D}_i = \mathbf{A}_i + \mathbf{E}_i$
   end for
3. **Step 2:** ***Building dictionaries***

Occlusion Dictionary Learning

Find a dictionary $\mathbf{E} \in \mathbb{R}^{m \times l}$ that provides the best representation for the sparse error $\mathbf{E}$
$\{\hat{\mathbf{E}}, \hat{\Gamma}\} = \arg\min_{\mathbf{E},\Gamma} \|\mathbf{E} - \mathbf{E}\Gamma\|_F^2 + \lambda\|\Gamma\|_1$

Sample Dictionary Learning

for $i = 1$ to p **do**
Find a dictionary $\mathbf{A}_i \in \mathbb{R}^{m \times k}$ that provides the best representation for the low-rank matrix $\mathbf{A}_i$
$\{\hat{\mathbf{A}}_i, \hat{\Delta}\} = \arg\min_{\mathbf{A}_i,\Delta} \|\mathbf{A}_i - \mathbf{A}_i\Delta\|_F^2 + \lambda\|\Delta\|_1$
end for

4. **Output:** A dictionary $\mathbf{E}$ and p dictionaries $\mathbf{A}_i \; i = 1 \cdots p$

Algorithm 4 Testing phase

1. **Input:** Learned dictionaries $\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2, \cdots, \mathbf{A}_p]$ and $\mathbf{E}$ from p subjects, and the test image y
2. **Step 1:** ***Recover the testing image*** y

Compute the sparse coefficient of y
$\{\hat{\boldsymbol{\Delta}}, \hat{\Gamma}\} = \arg\min_{\Delta,\Gamma} \|\mathrm{y} - \mathbf{A}\Delta - \mathbf{E}\Gamma\|_2^2 + \lambda_1\|\Delta\|_1 + \lambda_2\|\Gamma\|_1$
The recovered (without occlusion) face image is
$\hat{\mathrm{y}}_0 = \mathrm{y} - \mathbf{E}\hat{\Gamma}$

3. **Step 2:** ***Compute the residuals and classify***
   $e_i(\mathrm{y}) = \|\hat{\mathrm{y}}_0 - \mathbf{A}\boldsymbol{\delta}_i(\Delta)\|_2, \quad \forall i = 1, \cdots, p$

Output: Label of $\mathrm{y} = \arg\min_i\{e_i(\mathrm{y})\}$

3.2 Non-convex RPCA with ℓ_p Formulation

In our proposed approach (**LP-RPCA**), ℓ_p-norm is presented to replace the ℓ_1-norm since ℓ_p-norm is known as a measure offering a better approximation of the ℓ_0-norm than the ℓ_1-norm [44]. It is noted that the nuclear norm is a special form of the ℓ_1-norm on singular values of a matrix. Thus, we can apply ℓ_p-norm regularization on both sparse and low-rank matrices. Besides that by using the same penalty functions (and even the same p value) we can maintain the balance between low-rankness and sparsity. The parameter λ will then control this trade-off rather than penalty functions. The RPCA model in Eqn. (1) is approximated by a non-convex optimization problem:

$$\min_{\mathbf{L},\mathbf{S}} \left(\|\sigma(\mathbf{L})\|_p^p + \lambda\|\mathbf{S}\|_p^p\right) \quad \text{s.t. } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{106}$$

where $\sigma(\mathbf{L})$ denotes a vector of the singular values of the matrix $\mathbf{L}$. In general, we denote the ℓ_p-norm as a penalty function $g(\cdot) = |\cdot|^p$, thus, our proposed objective function can be redefined as follows:

$$\min_{\mathbf{L},\mathbf{S}} \left(\sum_{j=1}^{d} g(\sigma_j) + \lambda \sum_{ij=1}^{m \times n} g(s_{ij}) \right) \quad \text{s.t. } \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{107}$$

where σ_j denotes the j^{th} singular value of the matrix $\mathbf{L}$, s_{ij} denotes an element of $\mathbf{S}$ and $d \leq \min\{m, n\}$. Although we only consider ℓ_p-norm in this thesis, the general form of RPCA in Eqn. (107) can also be used with other penalty functions g. The penalty function $g : \mathbb{R} \rightarrow \mathbb{R}^+$ is assumed to be continuous, concave and monotonically increasing on $[0, \infty)$.

Chen et al. [30] proved that the penalized ℓ_p minimization problem is strongly NP-hard for any $0 \leq p < 1$. However, a solution for Eqn. (107) can be derived using the properties of gradient (or supergradient for nonsmooth points [8]) of a concave function. A vector $\mathbf{v}$ is a supergradient of a concave function g at the point $\mathbf{x} \in \mathbb{R}^n$ if $g(\mathbf{x}) + \langle \mathbf{v}, \mathbf{y} - \mathbf{x} \rangle \geq g(\mathbf{y})$ holds for every $\mathbf{y} \in \mathbb{R}^n$. Thus, the concave penalty function g can then be approximated as $g(x) \approx g(z) + \langle \nabla g(z), x - z \rangle$, where $z \in \mathbb{R}$ is sufficiently close to x. $\nabla g(z)$ denotes the gradient of g at z (first-order Taylor expansion is employed here). For the ℓ_p-norm function ($g(\cdot) = |\cdot|^p$), its gradient at z equals to $p|z|^{p-1}$.

The augmented Lagrangian form of the linearized problem in Eqn. (107) can be derived as follows:

$$\begin{aligned} \mathcal{L}(\mathbf{L}, \mathbf{S}, \mathbf{Y}, \mu) = & \sum_{j=1}^{d} \left(g(\sigma_j^k) + \left\langle \nabla g(\sigma_j^k), (\sigma_j - \sigma_j^k) \right\rangle \right) + \lambda \sum_{ij=1}^{m \times n} \left(g(s_{ij}^k) + \left\langle \nabla g(s_{ij}^k), (s_{ij} - s_{ij}^k) \right\rangle \right) \\ & + \langle \mathbf{Y}, \mathbf{M} - \mathbf{L} - \mathbf{S} \rangle + \frac{\mu^k}{2} \|\mathbf{M} - \mathbf{L} - \mathbf{S}\|_F^2 \end{aligned} \tag{108}$$

where $\mathbf{Y}$ is a Lagrangian multiplier (or dual variable) ensuring the equality constraint and μ^k is a penalty parameter used as step size for $\mathbf{Y}$ and is updated as $\mu^{k+1} = \rho \mu^k (\rho > 1)$. The matrices $\mathbf{S}$ and $\mathbf{L}$ are iteratively solved in two following convex optimization sub-problems by alternating between

fixing one and solving for the other.

$$\mathbf{S}^{k+1} = \arg\min_{\mathbf{S}} \left(\lambda \sum_{ij=1}^{m \times n} w_{ij}^{k} |s_{ij}| + \frac{\mu^k}{2} \|\mathbf{X}_S^k - \mathbf{S}\|_F^2 \right) \tag{109a}$$

$$\mathbf{L}^{k+1} = \arg\min_{\mathbf{L}} \left(\sum_{j}^{d} v_j^k \sigma_j + \frac{\mu^k}{2} \|\mathbf{X}_L^k - \mathbf{L}\|_F^2 \right) \tag{109b}$$

where $\mathbf{X}_S^k = \mathbf{M} - \mathbf{L}^k + \frac{\mathbf{Y}^k}{\mu^k}$ and $\mathbf{X}_L^k = \mathbf{M} - \mathbf{S}^{k+1} + \frac{\mathbf{Y}^k}{\mu^k}$. The weights are denoted as $w_{ij}^k = \nabla g(s_{ij}^k) = p(|s_{ij}^k| + \epsilon)^{p-1}$ and $v_j^k = \nabla g(\sigma_j^k) = p(\sigma_j^k + \epsilon)^{p-1}$ where ϵ $(0 < \epsilon \ll 1)$ is a small shifting quantity to avoid infinite values when the parameter vanishes. The matrices $\mathbf{S}$ and $\mathbf{L}$ are solved in a similar way (first update the values of the matrices via soft-thresholding [16] and singular value thresholding (SVT) [34], and then refine the corresponding weights). The soft-thresholding operator is defined as

$$\mathcal{S}_\tau(\mathbf{x})_i = \max\{|x_i| - \tau, 0\} \frac{x_i}{|x_i|} \tag{110}$$

It is well known that $\mathcal{SVT}_\tau(\mathbf{X})$ has an explicit expression as

$$\mathcal{SVT}_\tau(\mathbf{X}) = \mathbf{U}\left[\text{diag}\{(\mathbf{\Sigma} - \tau)_+\}\right]\mathbf{V}^\top \tag{111}$$

where the singular value decomposition (SVD) of $\mathbf{X}$ is $\mathbf{U\Sigma V}^\top$ and $(x)_+ = \max(x, 0)$.

Remark: Our approach can effectively isolate the weights or thresholds that are used implicitly in the generalized shrinkage/thresholding operator in [24]. Although the thresholds $|x|^{p-1}$ in [24] and our weights $p|x|^{p-1}$ are similar, the latter can achieve much better results as shown in Section 5.2.

This baseline framework can only handle the input data altogether as one big matrix without the ability to handle incremental input separately. We will formulate an online framework from this baseline procedure for the incremental data processing (i.e. decomposing the matrix as it is generated column-by-column) in the next section 3.3.

From our experiments, μ only influences the convergence rate while ϵ is a fixed number (floating-point relative accuracy). Thus, λ and p are the two parameters to be tuned. However, we found that there is an empirical relation between λ and p which is formulated as $\lambda = 1/((p/2) \times \sqrt{(\max(m, n))})$.

Therefore, only parameter p needs to be chosen, its value depends on applications.

All procedures are summarized in **Algorithm 5**.

Algorithm 5 Non-convex Robust PCA

1. **Input:** Observation matrix $\mathbf{M}$
2. **Initialize:** $k = 0$, $\mu^{(0)} > 0$, $w_{ij}^{(0)} = 1$, $v_j^{(0)} = 1$, $\mathbf{Y}^{(0)} = \frac{\mathbf{M}}{\sigma_1(\mathbf{M})}$;
3. **while** *not converged* **do**
   (I) **Sparse optimization** (Solving Eqn. (109a))
   (a) Find the value of $\mathbf{S}$ by soft thresholding:
   $\mathbf{S}^{k+1} = \mathcal{S}_{\frac{\lambda}{\mu^k} \times \mathbf{w}^k} \left(\mathbf{M} - \mathbf{L}^k + \frac{\mathbf{Y}^k}{\mu^k} \right)$;
   (b) Update the weights for each $ij = 1, \cdots, m \times n$:
   $w_{ij}^{k+1} = p(|s_{ij}^{k+1}| + \epsilon)^{p-1}$;
   (II) **Low-rank approximation** (Solving Eqn. (109b))
   (a) Find the value of $\mathbf{L}$ by weighted SVT:
   $\mathbf{L}^{k+1} = \mathcal{SVT}_{\frac{\mathbf{v}^k}{\mu^k}} \left(\mathbf{M} - \mathbf{S}^{k+1} + \frac{\mathbf{Y}^k}{\mu^k} \right)$;
   (b) Update the weights for each $j = 1, \cdots, d$:
   $v_j^{k+1} = p(\sigma_j^{k+1} + \epsilon)^{p-1}$;
   (III) **Update the parameters $\mathbf{Y}$ and μ:**
   (a) $\mathbf{Y}^{k+1} = \mathbf{Y}^k + \mu^k \left(\mathbf{M} - \mathbf{L}^{k+1} - \mathbf{S}^{k+1} \right)$;
   (b) $\mu^{k+1} = \rho \mu^k$;
   (b) $k \leftarrow k + 1$;
   end while
4. **Output:** $\mathbf{L} = \mathbf{L}^{k+1}, \mathbf{S} = \mathbf{S}^{k+1}$

3.3 Online Approach to Non-convex RPCA with ℓ_p Formulation

In this section, we first develop our online approach (**OLP-RPCA**) from the offline framework (**LP-RPCA**) and describe how to use the idea of an online ADMM [126] to solve our non-convex LP-RPCA problem. Secondly, we present a new adaptive online SVT operator. Then, we show that the computational complexity of OLP-RPCA is linear in both the sample dimension m and the number of samples n.

3.3.1 Online Optimization Method

Our aim is to decompose an input video frame-by-frame, i.e. matrix column-by-column, instead of decomposing the whole big matrix every time a new frame (column) becomes available. In this way we can deal with incremental frames effectively. Given a sample $\mathbf{m}_t$ at time t, we find a new

decomposition as $\mathbf{m}_t = \mathbf{l}_t + \mathbf{s}_t$, where $\mathbf{m}_t, \mathbf{l}_t, \mathbf{s}_t \in \mathbb{R}^m$ to solve the matrix decomposition problem formulated in Eqn. (112).

$$\min_{\mathbf{l}_t, \mathbf{s}_t} \sum_{t=1}^{n} \left(g(\sigma(\mathbf{L}_t)) + \lambda g(\mathbf{s}_t) \right) \quad \text{s.t.} \quad \mathbf{L} + \mathbf{S} = \mathbf{M} \tag{112}$$

where $\mathbf{L} = [\mathrm{l}_1\ \mathrm{l}_2 \cdots \mathrm{l}_n]$, $\mathbf{L}_t = [\mathbf{l}_1, \cdots, \mathbf{l}_t]$, $\mathbf{S} = [\mathrm{s}_1\ \mathrm{s}_2 \cdots \mathrm{s}_n]$ and $\mathbf{M} = [\mathbf{m}_1\ \mathbf{m}_2 \cdots \mathbf{m}_n]$. Our non-convex OLP-RPCA problem can be solved by a modified version of **Algorithm 5** following the idea of an online ADMM (OADMM) in [126]. Instead of having a loop until converged, at each iteration for the t-th frame/column, the online algorithm consists of just one pass through the following update steps:

Step 1: Obtain a new column $\mathbf{l}_{t+1}$ of the low-rank matrix from the new frame $\mathbf{m}_t$

$$\mathbf{l}_{t+1} = \arg\min_{\mathbf{l}} \left(\sum_{j}^{d} v_j^t \sigma_j + \frac{\mu}{2} \|\mathbf{x}_l^t - \mathbf{l}\|_2^2 + \frac{\eta}{2} \|\mathbf{l} - \mathbf{l}_t\|_2^2 \right) \tag{113}$$

where $\mu > 0$, $\eta \geq 0$ are the constants. μ can be updated at each iteration but we found that it is better when μ is fixed. $\mathbf{x}_l^t = \mathbf{m}_t - \mathbf{s}_t + \frac{\mathbf{y}_t}{\mu}$ and the weights are denoted as $v_j^t = p(\sigma_j^t + \epsilon)^{p-1}$ with σ_j^t are the singular values of the matrix $\mathbf{L}_t$.

Step 2: Obtain a new column $\mathbf{s}_{t+1}$ of the sparse matrix from the new frame $\mathbf{m}_t$

$$\mathbf{s}_{t+1} = \arg\min_{\mathbf{s}} \lambda \mathbf{w}_t |\mathbf{s}| + \frac{\mu}{2} \|\mathbf{m}_t - \mathbf{l}_{t+1} + \frac{\mathbf{y}_t}{\mu} - \mathbf{s}\|_2^2 \tag{114}$$

where the weight values for the sparse vector $\mathbf{s}_t$ are defined as $\mathbf{w}_t = p(|\mathbf{s}_t| + \epsilon)^{p-1}$.

Step 3: Update the dual variable $\mathbf{y}$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \mu \left(\mathbf{m}_t - \mathbf{l}_{t+1} - \mathbf{s}_{t+1} \right) \tag{115}$$

The update step for the sparse matrix (**Step 2**) is simply applying soft-thresholding operator defined as Eqn. (110) in Section 3.2, since it is separable for each column vector. However, the update step of the low-rank matrix (**Step 1**) is more complicated because of computing the singular values of a matrix involving in SVT operator on a vector $\mathbf{x}_t$ (defined as $\mathcal{SVT}_{\tau,\mathbf{X}}(\mathbf{x}_t) = \mathbf{U} \left[\mathrm{diag}\{(\mathbf{\Sigma} - \tau)_+\} \right] \mathbf{V}^\top$, where the singular value decomposition (SVD) of $[\mathbf{X}|\mathbf{x}_t]$ is $\mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top$ and $\mathbf{X}$ is the matrix in previous

step). Thus, we propose an adaptive online SVT operator which incorporates an incremental SVD method to update the decomposition incrementally. This operator will be described in details in the next section.

3.3.2 Adaptive Online SVT Operator

We first employ an incremental SVD (ISVD) method described in [10] to *find the singular values of the new matrix* without performing a full SVD and then *apply a thresholding operator* on this new result. The ISVD method is described briefly in the following.

Given that an existing rank-r SVD of the current matrix $\mathbf{X}_t \in \mathbb{R}^{m\times t}$ $(t \leq n)$, where n is the number of columns of the full matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$, at step t is $\mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top$ (where $\mathbf{U} \in \mathbb{R}^{m\times r}$, $\mathbf{V} \in \mathbb{R}^{t\times r}$, $\mathbf{\Sigma}_t \in \mathbb{R}^{r\times r}$ and $r \leq \min(m,t)$), the SVD of the new matrix adding c columns is derived as follows:

$$\begin{aligned}\begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_t & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^\top &= \begin{bmatrix} \mathbf{U} & \left(\mathbf{I} - \mathbf{U}\mathbf{U}^\top\right)\mathbf{C}/\mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_t & \mathbf{U}^\top\mathbf{C} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix}^\top \\ &= \begin{bmatrix} \mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_t & \mathbf{C} \end{bmatrix}\end{aligned} \tag{116}$$

where the matrix $\mathbf{C} \in \mathbb{R}^{m\times c}$ contains new data columns and the product matrix $\mathbf{JK}$ is a *QR-decomposition* of $\left(\mathbf{I} - \mathbf{U}\mathbf{U}^\top\right)\mathbf{C}$. The ISVD algorithm updates the decomposition by *diagonalizing* $\mathbf{Q} = \begin{bmatrix} \mathbf{\Sigma}_t & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} = \begin{bmatrix} \mathbf{\Sigma}_t & \mathbf{U}^\top\mathbf{C} \\ 0 & \mathbf{K} \end{bmatrix}$ where $\mathbf{Q}$ is decomposed as $\mathbf{U}'\mathbf{\Sigma}_\mathbf{Q}\mathbf{V}'^\top$. Then the new SVD is

$$\mathbf{U}''\mathbf{\Sigma}''\mathbf{V}''^\top = \begin{bmatrix} \mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_t & \mathbf{C} \end{bmatrix} \tag{117}$$

The updated matrices are obtained by *matrix multiplication* (or *subspace rotation*) as follows:

$$\mathbf{U}'' = \begin{bmatrix} \mathbf{U} & \mathbf{J} \end{bmatrix} \mathbf{U}'; \mathbf{\Sigma}'' = \mathbf{\Sigma}_\mathbf{Q}; \mathbf{V}'' = \begin{bmatrix} \mathbf{V} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \mathbf{V}' \tag{118}$$

Note that in some cases the resulting SVD will have rank r rather than rank $r+1$ singular values,

Eqn. (118) can be replaced with the truncated forms:

$$\mathbf{U}'' = \mathbf{U}\mathbf{U}'; \mathbf{\Sigma}'' = \mathbf{\Sigma}_{\mathbf{Q}}; \mathbf{V}'' = \mathbf{V}\mathbf{V}' \tag{119}$$

The basic idea of ISVD algorithm is to replace the full SVD decomposition into a series of much smaller SVD decompositions for the new columns. However, this is not an efficient way when the dimension of the inner matrix $\mathbf{Q}$ is large. There is another issue with the ISVD algorithm: the computational costs will increase column-by-column. This is because it takes $O((m+t)(r+c)^2)$ time to process a new column and t will increase when new columns are updated. The overall cost to obtain the decomposition of the full matrix $\mathbf{X}$ will be $O(mnr^2)$ (See section 3.3.3 for more detailed analysis). Therefore, to reduce the computational costs, we propose a modified ISVD involving two costliest steps: *matrix multiplication* and *diagonalizing*.

Firstly, we observe that instead of performing the costly matrix multiplication of big matrices $\mathbf{U}, \mathbf{V}$ with smaller ones $\mathbf{U}'$ and $\mathbf{V}'$ as in Eqn. (118), we can keep matrices $\mathbf{U}'$ and $\mathbf{V}'$, then update them together with $\mathbf{U}$ and $\mathbf{V}$. Thus, this can reduce the complexity of the baseline ISVD since it only performs the matrix multiplication steps on the small matrices. We form an extended SVD of the current matrix $\mathbf{X}_t$ at step t as follows:

$$\mathbf{X}_t = \mathbf{U}\mathbf{U}'\mathbf{\Sigma}_t{\mathbf{V}'}^{\top}\mathbf{V}^{\top} \tag{120}$$

where $\mathbf{U}\mathbf{U}'$, $\mathbf{V}\mathbf{V}'$, $\mathbf{U}$, and $\mathbf{U}'$ (but not $\mathbf{V}'$ or $\mathbf{V}$) are orthonormal. Then, we apply the same derivation as in Eqn. (116) as follows:

$$\begin{bmatrix} \mathbf{U}\mathbf{U}' & \mathbf{J} \end{bmatrix} \begin{bmatrix} \mathbf{\Sigma}_t & \mathbf{L} \\ 0 & \mathbf{K} \end{bmatrix} \begin{bmatrix} \mathbf{V}\mathbf{V}' & 0 \\ 0 & \mathbf{I} \end{bmatrix}^{\top} \tag{121}$$

Similarly, the matrix $\mathbf{Q}$ is diagonalized as $\mathbf{A}\mathbf{\Sigma}'_{\mathbf{Q}}\mathbf{B}$. Two large outer matrices $\mathbf{U}$ and $\mathbf{V}$ are now updated by appending columns and rows, respectively. Only the span of the left and right subspaces are maintained in these two matrices while subspace rotations are deferred to the smaller matrices $\mathbf{U}'$ and $\mathbf{V}'$ instead of multiplying $\mathbf{U}, \mathbf{V}$ each time. There are two cases depending on whether the

rank r increases or not. To simplify, we consider the case of adding a new column, i.e. $c = 1$, thus the matrix $\mathbf{C}$ will now become a vector $\mathbf{c}$.

If *the rank increases*, then for the matrices $\mathbf{U}$ and $\mathbf{U}'$,

$$\mathbf{U}_{\text{new}} = [\mathbf{U}\ \mathbf{j}] = [\mathbf{U}\ \ (\mathbf{c} - \mathbf{U}\mathbf{U}^\top \mathbf{c})/k]; \mathbf{U}'_{\text{new}} = \begin{bmatrix} \mathbf{U}' & 0 \\ 0 & 1 \end{bmatrix} \mathbf{A} \tag{122}$$

where $k = \|\mathbf{c} - \mathbf{U}\mathbf{U}^\top \mathbf{c}\|$. The matrices $\mathbf{V}$ and $\mathbf{V}'$ are updated simply as,

$$\mathbf{V}_{\text{new}} = \begin{bmatrix} \mathbf{V} & 0 \\ 0 & 1 \end{bmatrix}; \mathbf{V}'_{\text{new}} = \begin{bmatrix} \mathbf{V}' & 0 \\ 0 & 1 \end{bmatrix} \mathbf{B} \tag{123}$$

If *the rank does not increase*, then only $\mathbf{U}'_{\text{new}} = \mathbf{U}'\mathbf{A}$ while $\mathbf{U}$ is the same and the matrices $\mathbf{V}$ and $\mathbf{V}'$ are computed as,

$$\mathbf{V}'_{\text{new}} = \mathbf{V}'\mathbf{W}; \mathbf{V}'^{+}_{\text{new}} = \mathbf{W}^{+}\mathbf{V}'^{+}; \mathbf{V}_{\text{new}} = \begin{bmatrix} \mathbf{V} \\ \mathbf{V}'^{+}\mathbf{W} \end{bmatrix} \tag{124}$$

where $\mathbf{W}^{+} = (\mathbf{I} + \mathbf{w}^\top \mathbf{w}/(1 - \mathbf{w}\mathbf{w}^\top))\mathbf{W}^\top$ and $\mathbf{V}'^{+}$ is the pseudo-inverse which is computed and updated as

$$\mathbf{V}'^{+}_{\text{new}} = \mathbf{B}^\top \begin{bmatrix} \mathbf{V}'^{+} & 0 \\ 0 & 1 \end{bmatrix} \tag{125}$$

This procedure reduces the complexity of the update steps and eliminates the numerical error.

Secondly, we observe that it would be redundant to use the full decomposition ($\mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top$) of the previous column t to update the new columns since SVT operator ($\mathcal{SVT}_\tau(\mathbf{X}_t)$) would discard those singular values lower than τ and their corresponding singular vectors. Therefore, a good strategy is to use partial SVD instead of the full one, i.e. we only consider those singular values exceeding threshold τ and their associated singular vectors. We will have the reduced input for ISVD as $\tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}_t\tilde{\mathbf{V}}^\top$, where $\tilde{\mathbf{U}} \in \mathbb{R}^{m \times k}$, $\tilde{\mathbf{V}} \in \mathbb{R}^{t \times k}$, $\tilde{\mathbf{\Sigma}}_t \in \mathbb{R}^{k \times k}$ and k is the number of singular values higher than threshold τ. If k is small compared to the matrix dimension $\min(m, t)$ and the approximated rank r, then ISVD update can be computed efficiently with this partial SVD input because the size of the matrix $\mathbf{Q}$ will be smaller and less computation time is needed to decompose it. In some situations, the strategy of using only partial SVD might not be helpful to accelerate the

computation of ISVD. It is when k is not small compared to the matrix dimension $\min(m, t)$ and the approximated rank r. Therefore, we introduce the second point of accelerating the baseline ISVD.

The ISVD algorithm [10] requires an initial decomposition of the current matrix $\mathbf{X}_t$ as $\mathbf{U}\mathbf{\Sigma}_t\mathbf{V}^\top$. To obtain this initial decomposition, we suggest to run the offline LP-RPCA algorithm (**Algorithm 5**) for the first N training frames. This initialization strategy would work efficiently when the input video (or matrix) is truly low-rank, e.g. a video with static background. However, if the input video has dynamic background, e.g. water flow and trees, we can decrease the regularized parameter η gradually. This strategy will allow the new low-rank column $\mathbf{l}_{t+1}$ to change from the previous column $\mathbf{l}_t$ (See Eq. (113)).

The online LP-RPCA procedure is summarized in **Algorithm 6**.

Algorithm 6 Online Non-convex Robust PCA

1. **Input:** Given a set of frames $\mathbf{m}_t$, $1 \leq t \leq n$
2. **Initialize:** $w_i^{(1)} = 1$, $v_j^{(1)} = 1$, $\mathbf{y}_1 = \frac{\mathbf{m}_1}{|\mathbf{m}_1|}$;
3. **for** $t = 1$ to n **do**
   (I) **Low-rank approximation**
   (a) Find the value of $\mathbf{L}_t$ by online SVT:
   $\mathbf{l}_{t+1} = \mathcal{SVT}_{\mathbf{v}_t*(\mu)^{-1}}\left(\mathbf{m}_t - \mathbf{s}_t + \frac{\mathbf{y}_t}{\mu}\right)$;
   (b) Update the weights for each $j = 1, \cdots, d$:
   $v_j^{t+1} = p(\sigma_j^{t+1} + \epsilon)^{p-1}$;
   (II) **Sparse optimization**
   (a) Find the value of $\mathbf{S}_t$ by soft thresholding:
   $\mathbf{s}_{t+1} = \mathcal{S}_{\frac{\lambda}{\mu}*\mathbf{w}_t}\left(\mathbf{m}_t - \mathbf{l}_{t+1} + \frac{\mathbf{y}_t}{\mu}\right)$;
   (b) Update the weights for each $i = 1, \cdots, m$:
   $w_i^{t+1} = p(|\mathbf{s}_{t+1}| + \epsilon)^{p-1}$;
   (III) **Update the parameter $\mathbf{y}$:**
   (a) $\mathbf{y}_{t+1} = \mathbf{y}_t + \mu\left(\mathbf{m}_{t+1} - \mathbf{l}_{t+1} - \mathbf{s}_{t+1}\right)$;
   end for
4. **Output:** $\mathbf{L} = \mathbf{L}_n, \mathbf{S} = \mathbf{S}_n$

3.3.3 Complexity Analysis

In our online algorithm, the costliest step is the updating step of low-rank matrix which involves an ISVD method as described above. Thus, we will describe and analyze the complexity of this step.

For the baseline ISVD [10], the costs of its three main steps (QR-decomposition, diagonalization

and matrix multiplication) are $O(m(r+c)^2)$, $O((r+c)^3)$ and $O((m+t)(r+c)^2)$, respectively. Thus, the complexity of the whole update procedure for the new columns is $O((m+t)(r+c)^2)$. For $c=1$, this procedure is applied n times to compute the SVD of the full matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$. As a result, the overall complexity is $O(mnr^2 + n^2r^2) = O(mnr^2)$ $(m > n)$.

For the modified ISVD, considering $c = 1$, the computational cost of the QR-decomposition, the diagonalization of $\mathbf{Q}$ and the costly matrix multiplication can be reduced to $O(mk)$, $O(k^2)$, and $O(k^3)$ with the modified Gram-Schmidt algorithm, sparse diagonalizations and the above decomposition, respectively [10]. As a result, each update step has the complexity of $O(mk + k^3)$ and the overall complexity is $O(mnk)$ to perform SVD on the entire matrix $\mathbf{M}$ incrementally (comparing to the matrix size, the desired rank k is relatively small).

In comparison, batch (or offline) RPCA performs a full SVD and then a thresholding operation for updating the low-rank matrix $\mathbf{L}$ in each iteration. The complexity of the offline SVD step is $O(m^2n + mn^2 + n^3)$ [49]. For OR-PCA [41], the computational cost is $O(mr^2)$ in each iteration. Thus, the overall complexity of OR-PCA will be $O(mnr^2)$ while the overall cost of our OLP-RPCA is $O(mnk)$ (in some applications such as background subtraction k is much smaller than the estimated rank of the matrix r), which is linear in both the sample dimension and the number of samples. Compared to the batch version, it is substantially faster than $O(m^2n)$ when $k \ll m$ and $m > n$.

3.3.4 Remarks

An efficient online OLP-RPCA is proposed to solve the matrix decomposition problem incrementally. This online method is developed from a new LP-RPCA approach via ℓ_p-norm regularization on both low-rank and sparse components. This chapter has mathematically provided the complexity analysis for the OLP-RPCA method. The convergence analysis of the LP-RPCA method will be presented in Appendix A. The work in this section and its corresponding experimental results have been published in the Computer Vision and Image Understanding Journal 2017. The next section will present our proposed ℓ_p Singular Value Decomposition for matrix factorization problem.

3.4 Matrix Factorization: ℓ_p Singular Value Decomposition

Given a matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ that contains missing values, noise and outliers, this work aims to introduce a novel RP-SVD approach using ℓ_p-norm, where $0 < p < 1$, to further enhance the robustness of SVD to deal with outliers and noise. The Singular Value Decomposition problem can be then formulated by minimizing the reconstruction error as follows:

$$\begin{aligned} \min_{\mathbf{U},\mathbf{\Sigma},\mathbf{V}} \quad & \|\mathbf{M} \odot (\mathbf{X} - \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top)\|_p \\ s.t., \quad & \mathbf{U}^\top\mathbf{U} = \mathbf{I}, \mathbf{V}^\top\mathbf{V} = \mathbf{I} \end{aligned} \tag{126}$$

where the left singular vectors $\mathbf{U}_i \in \mathbb{R}^d$ and the right singular vectors $\mathbf{V}_i \in \mathbb{R}^n$ $(i = 1, \cdots, r)$ are orthonormal. Each has a unit length and every pair is orthogonal, i.e. $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$. r denotes the rank of $\mathbf{X}$, where $r \leq \min(d, n)$. $\mathbf{\Sigma} \in \mathbb{R}^{r\times r}$ is a diagonal matrix containing the square root of the eigenvalues from $\mathbf{U}$ or $\mathbf{V}$ in descending order. Far apart from the conventional SVD method, our proposed RP-SVD approach presented in Eqn. (126) allows to decompose an input matrix $\mathbf{X}$ containing missing values and outliers denoted by the weight matrix $\mathbf{M}$, where $\mathbf{M}(i, j) > 0$ if the data point $\mathbf{X}_{i,j}$ exists, otherwise $\mathbf{M}(i, j) = 0$. $\odot$ denotes the component-wise multiplication.

Generally, Eqn. (126) is a non-convex problem. When $p = 1$, the proposed SVD-ℓ_1 reformulation can be redefined as in Eqn. (127) in the form of trace norm regularization,

$$\begin{aligned} \min_{\mathbf{U},\mathbf{\Sigma},\mathbf{V},\mathbf{E}} & \|\mathbf{M} \odot (\mathbf{X} - \mathbf{E})\|_1 + \lambda\|\mathbf{E}\|_* \\ s.t., \quad & \mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top, \mathbf{U}^\top\mathbf{U} = \mathbf{I}, \mathbf{V}^\top\mathbf{V} = \mathbf{I} \end{aligned} \tag{127}$$

where the parameter λ controls the trade-off between trace norm regularization and reconstruction fidelity. Let $\mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \mathbf{L}\mathbf{R}$, where $\mathbf{L}$ is an orthogonal matrix, i.e. $\mathbf{L}^\top\mathbf{L} = \mathbf{I}$. Then, the problem in Eqn. (127) can be solved in the following form [141].

$$\begin{aligned} \min_{\mathbf{L},\mathbf{R},\mathbf{E}} \quad & \|\mathbf{M} \odot (\mathbf{X} - \mathbf{E})\|_1 + \lambda\|\mathbf{R}\|_* \\ s.t., \quad & \mathbf{E} = \mathbf{L}\mathbf{R}, \mathbf{L}^\top\mathbf{L} = \mathbf{I} \end{aligned} \tag{128}$$

where $\|\mathbf{E}\|_* = \|\mathbf{LR}\|_* = \|\mathbf{R}\|_*$ since $\mathbf{L}$ is orthogonal. In order to solve the problem in Eqn. (128), the corresponding augmented Lagrangian function can be found and then the Alternating Direction Method of Multipliers method can be employed to find the optimal values for three matrices $\mathbf{L}, \mathbf{R}, \mathbf{E}$.

In this thesis, we propose RP-SVD using ℓ_p-norm to further enhance the robustness of SVD when dealing with outliers and noise. The objective function is reformulated as follows:

$$\begin{aligned} &\min_{\mathbf{L},\mathbf{R},\mathbf{E}} \|\mathbf{M} \odot (\mathbf{X} - \mathbf{E})\|_p + \lambda \|\sigma(\mathbf{R})\|_p \\ &s.t., \;\; \mathbf{E} = \mathbf{LR}, \mathbf{L}^\top \mathbf{L} = \mathbf{I} \end{aligned} \tag{129}$$

which is equivalent to

$$\begin{aligned} &\min_{\mathbf{L},\mathbf{R},\mathbf{E}} \sum_{i,j} g\left(\mathbf{M}_{i,j}(\mathbf{X}_{i,j} - \mathbf{E}_{i,j})\right) + \lambda \sum_{j} g\left(\sigma_j(\mathbf{R})\right) \\ &s.t., \;\; \mathbf{E} = \mathbf{LR}, \mathbf{L}^\top \mathbf{L} = \mathbf{I} \end{aligned} \tag{130}$$

where $g(\cdot) = |\cdot|^p$. The corresponding augmented Lagrangian function can be formulated as follows,

$$\begin{aligned} \mathcal{L}_\beta(\mathbf{L}, \mathbf{R}, \mathbf{E}, \mathbf{Y}) \triangleq & \sum_{i,j} g\left(\mathbf{M}_{i,j}(\mathbf{X}_{i,j} - \mathbf{E}^k_{i,j})\right) + \left\langle \nabla g\left(\mathbf{M}_{i,j}(\mathbf{X}_{i,j} - \mathbf{E}^k_{i,j})\right), \mathbf{M}_{i,j}\left(\mathbf{E}^k_{i,j} - \mathbf{E}_{i,j}\right)\right\rangle \\ & + \lambda \sum_{j} g\left(\sigma_j(\mathbf{R}^k)\right) + \left\langle \nabla g\left(\sigma_j(\mathbf{R}^k)\right), \sigma_j(\mathbf{R}) - \sigma_j(\mathbf{R}^k)\right\rangle + < \mathbf{Y}, \mathbf{E} - \mathbf{LR} > + \frac{\beta}{2}\|\mathbf{E} - \mathbf{LR}\|_F^2 \end{aligned} \tag{131}$$

where $\mathbf{Y}$ is the Lagrange multipliers ensuring the linear constraints, $\beta > 0$ is the penalty parameter for the violation of the linear constraints. The problem defined in Eqn. (130) can be solved using ADMM approach to minimize the variables by iteratively solving the following convex optimization sub-problems:

$$\begin{cases} \mathbf{L}^{k+1} = \arg\min_{\mathbf{L}} \mathcal{L}_\beta(\mathbf{L}, \mathbf{R}^k, \mathbf{E}^k, \mathbf{Y}^k) \\ \mathbf{R}^{k+1} = \arg\min_{\mathbf{R}} \mathcal{L}_\beta(\mathbf{L}^{k+1}, \mathbf{R}, \mathbf{E}^k, \mathbf{Y}^k) \\ \mathbf{E}^{k+1} = \arg\min_{\mathbf{E}} \mathcal{L}_\beta(\mathbf{L}^{k+1}, \mathbf{R}^{k+1}, \mathbf{E}, \mathbf{Y}^k) \\ \mathbf{Y}^{k+1} = \mathbf{Y}^k + \beta(\mathbf{E}^{k+1} - \mathbf{L}^{k+1}\mathbf{R}^{k+1}) \end{cases}$$

3.4.1 Non-convex Optimization Method

Given $\mathbf{R}^k$ and $\mathbf{E}^k$, find $\mathbf{L}^{k+1}$

By fixing $\mathbf{R}^k$ and $\mathbf{E}^k$ in the iteration k, $\mathbf{L}^{k+1}$ can be updated by solving the sub-problem as follows:

$$\min_{\mathbf{L}} \frac{\beta}{2}\|(\mathbf{E}^k + \beta^{-1}\mathbf{Y}^k) - \mathbf{L}\mathbf{R}^k)\|_F^2 \quad s.t. \;\; \mathbf{L}^\top\mathbf{L} = \mathbf{I}$$

This optimization problem is known as the orthogonal Procrustes problem [53]. The global optimal solution can be found by first applying SVD as $[\mathbf{U}', \mathbf{S}', \mathbf{V}'] = svd((\mathbf{E}^k + \beta^{-1}\mathbf{Y}^k)\mathbf{R}^{k\top})$. Then, $\mathbf{L}^{k+1}$ can be updated as follows [141],

$$\mathbf{L}^{k+1} \leftarrow \mathbf{U}'\mathbf{V}'^\top$$

Given $\mathbf{L}^{k+1}$ and $\mathbf{E}^k$, find $\mathbf{R}^{k+1}$

In the second step, given $\mathbf{L}^{k+1}$ and $\mathbf{E}^k$, $\mathbf{R}^{k+1}$ can be found using the following formula,

$$\min_{\mathbf{R}} \lambda \sum_j v_j^k \sigma_j + < \mathbf{Y}^k, \mathbf{E}^k - \mathbf{L}^{k+1}\mathbf{R} > + \frac{\beta}{2}\|\mathbf{E}^k - \mathbf{L}^{k+1}\mathbf{R}\|_F^2 \tag{132}$$

where $v_j^k = \nabla g(\sigma_j(\mathbf{R}^k))$ and σ_j is the j-th singular values of the matrix $\mathbf{R}^k$. Since $\mathbf{L}^{k+1}$ is orthogonal, Eqn. (132) can be rewritten as,

$$\min_{\mathbf{R}} \lambda\beta^{-1} \sum_j v_j^k \sigma_j + \frac{1}{2}\|\mathbf{R} - \mathbf{L}^{k+1\top}(\mathbf{E}^k + \beta^{-1}\mathbf{Y}^k)\|_F^2 \tag{133}$$

Based on Theorem 1 in [34], the solution of (133) is given by the weighted singular value thresholding (WSVT). In WSVT, the SVD is first employed, $[\mathbf{U}', \mathbf{S}', \mathbf{V}'] = svd(\mathbf{L}^{k+1\top}(\mathbf{E}^k + \beta^{-1}\mathbf{Y}^k))$, the optimal values of $\mathbf{R}^{k+1}$ can be then updated by shrinking the diagonal matrix $\mathbf{S}'$ via the soft-thresholding (shrinkage) operator $\mathbf{T}_\tau[x]$:

$$\mathbf{R}^{k+1} \leftarrow \mathbf{U}'\mathbf{T}_{\lambda\beta^{-1}v_j^k}[\mathbf{S}']\mathbf{V}'^\top \tag{134}$$

where the weights v_j^k are updated at each iteration as $v_j^k = p(\sigma_j^k + \epsilon)^{p-1} (0 < \epsilon \ll 1)$. The shrinkage operator is defined as follows,

$$\mathbf{T}_\tau[x] = \max(|x| - \tau, 0)(sgn)(x) \tag{135}$$

where $(sgn)(x)$ is the sign function.

Given $\mathbf{L}^{k+1}$ and $\mathbf{R}^{k+1}$, find $\mathbf{E}^{k+1}$

Given $\mathbf{L}^{k+1}$ and $\mathbf{R}^{k+1}$, $\mathbf{E}^{k+1}$ can be updated using the shrinkage technique in [141],

$$\min_{\mathbf{E}} \sum_{i,j} \mathbf{W}_{i,j}^k (\mathbf{M}_{i,j}(\mathbf{X}_{i,j} - \mathbf{E}_{i,j})) + \frac{\beta}{2} \|\mathbf{E} - (\mathbf{L}^{k+1}\mathbf{R}^{k+1} - \beta^{-1}\mathbf{Y}^k)\|_F^2$$

where $\mathbf{W}_{i,j}^k = \nabla g\left(\mathbf{M}_{i,j}^k(\mathbf{X}_{i,j} - \mathbf{E}_{i,j}^k)\right) = p(\mathbf{M}_{i,j}^k(\mathbf{X}_{i,j} - \mathbf{E}_{i,j}^k) + \epsilon)^{p-1}$. Therefore, the observed $\mathbf{M} \odot \mathbf{E}$ and missing values $\overline{\mathbf{M}} \odot \mathbf{E}$ in $\mathbf{E}$ can be updated as follows,

$$\begin{cases} \mathbf{M} \odot \mathbf{E} \leftarrow \mathbf{M} \odot (\mathbf{X} - \mathbf{T}_{\beta^{-1}\mathbf{W}}[\mathbf{X} - \left(\mathbf{L}^{k+1}\mathbf{R}^{k+1} - \beta^{-1}\mathbf{Y}^k\right)]) \\ \overline{\mathbf{M}} \odot \mathbf{E} \leftarrow \overline{\mathbf{M}} \odot (\mathbf{L}^{k+1}\mathbf{R}^{k+1} - \beta^{-1}\mathbf{Y}^k) \end{cases} \tag{136}$$

where $\overline{\mathbf{M}}$ is the complement of $\mathbf{M}$.

3.4.2 Online Robust ℓ_p-norm SVD

This section describes how to extend our RP-SVD method to work online with the aims of reducing the complexity of the conventional SVD in terms of processing storage and computational time. Online Robust ℓ_p-norm SVD (ORP-SVD) factorizes an input matrix column-by-column instead of processing the whole matrix at once. Given a new column $\mathbf{c}_t$ at time t, the singular value decomposition of the new matrix $[\mathbf{X}_{t-1} \mid \mathbf{c}_t]$ is defined as $\mathbf{e}_t = \mathbf{L}_t\mathbf{r}_t$, where $\mathbf{L}_t$ and $\mathbf{r}_t$ are the decomposed matrix and vector at time t, respectively. The above decomposition is repeated until all columns of the input matrix $\mathbf{X}$ are processed. The final decomposition $\mathbf{L}_n\mathbf{R}_n$ (where $\mathbf{R}_n = [\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n]$) will be an approximated solution of the matrix decomposition problem formulated in Eqn. (130).

We extended the ADMM approach in Section 3.4.1 to solve the problem (130) incrementally.

Instead of having a loop until converge, at each iteration for the new column $\mathbf{c}_t$, the online algorithm consists of just one pass through the following update steps:

Step 1: Obtain a new vector $\mathbf{r}_t \in \mathbb{R}^r$ from the existing matrices $\mathbf{L}_{t-1}$ and $\mathbf{e}_{t-1}$

$$\mathbf{r}_t = \arg\min_{\mathbf{r}} \lambda\beta^{-1} \sum_j v_j^{t-1}\sigma_j + \frac{1}{2}\|\mathbf{r} - \mathbf{L}_{t-1}^\top(\mathbf{e}_{t-1} + \beta^{-1}\mathbf{y}_{t-1})\|_F^2 \tag{137}$$

where the weights are denoted as $v_j^{t-1} = p(\sigma_j^{t-1} + \epsilon)^{p-1}$ with σ_j^{t-1} are the singular values of the matrix $\mathbf{R}_{t-1}$.

Step 2: Obtain a new column $\mathbf{e}_t \in \mathbb{R}^m$ from the new data column $\mathbf{c}_t$ and vector $\mathbf{r}_t$

$$\mathbf{e}_t = \arg\min_{\mathbf{e}} \sum_i^m \mathbf{W}_{(t-1)}^i(\mathbf{M}_t^i(\mathbf{X}_t^i - \mathbf{e}^i)) + \frac{\beta}{2}\|\mathbf{e} - (\mathbf{L}_{t-1}\mathbf{r}_t - \beta^{-1}\mathbf{y}_{t-1})\|_F^2 \tag{138}$$

where the weight values for the frame $\mathbf{X}_{t-1}$ are defined as $\mathbf{W}_{t-1} = p(\mathbf{M}_t(\mathbf{X}_{t-1} - \mathbf{e}_{t-1}) + \epsilon)^{p-1}$.

Step 3: Obtain an updated matrix $\mathbf{L}_t \in \mathbb{R}^{m\times r}$ from the new vectors $\mathbf{r}_t$ and $\mathbf{e}_t$ as,

$$\mathbf{L}_t \leftarrow \mathbf{U}'\mathbf{V}'^\top$$

where $[\mathbf{U}', \mathbf{S}', \mathbf{V}'] = svd((\mathbf{e}_t + \beta^{-1}\mathbf{y}_{t-1})\mathbf{r}_t^\top)$ and $\mathbf{y}_{t-1} \in \mathbb{R}^m$.

Step 4: Update the dual variable $\mathbf{Y}_t$ as follows,

$$\mathbf{y}_t = \mathbf{y}_{t-1} + \mu\left(\mathbf{e}_t - \mathbf{L}_t\mathbf{r}_t\right)$$

The update step for each column $\mathbf{e}_t$ of the matrix $\mathbf{E}$ is simply to apply the soft-thresholding operator as in Eqn. (136) since it is separable for each column vector. However, the update step of the matrix $\mathbf{V}$ is more complicated because of computing the singular values of a matrix involving in SVT operator. To solve this problem, we employ the incremental SVD (ISVD) method described in [10] to find the singular values of the new matrix without performing a full SVD and then apply a thresholding operator on this matrix.

3.4.3 Remarks

A RP-SVD method for analyzing two-way functional data is proposed. This chapter also describes an online version of the method (ORP-SVD) to employ online processing data. This ORP-SVD is able to achieve real-time performance without parallelizing or implementing on a graphics processing unit. The work in this section and its corresponding experimental results have been published in the NIPS workshop 2015.

3.5 Conclusion

First, this chapter has identified the problems resisting the performance of the low-rank approximation and the sparse representation methods in face recognition. The problems are related to the testing stage and that the sparse components were not properly used. This chapter presented a new framework to make a better use of sparse components resulted from low-rank decomposition in the training phase. Using the information captured from the training stage, we successfully improve the testing stage of the recognition process. Later, this chapter has proposed an efficient online LP-RPCA to solve the matrix decomposition problem incrementally. This online method is developed from a new LP-RPCA approach via p-norm regularization on both low-rank and sparse components. In addition, this chapter presents a novel Robust ℓ_p-norm ($0 < p < 1$) Singular Value Decomposition (RP-SVD) approach to solve the SVD problem approximately using ℓ_p-norm solution. Far apart from the conventional SVD approaches, our proposed RP-SVD method is able to deal with input matrices containing missing values and can find optimal solutions for the matrix completion problems. In addition, it can also find optimal subspaces that are robust to noise and outliers.

Chapter 4

Deep Learning Approach to Image Analysis

In Section 3.1, we proposed a novel face recognition (FR) method based on learning low-rank matrix and sparse variation representation to improve the performance of FR under various affecting conditions. The main idea of this system is to learn a sample dictionary (i.e. subject identity information) and an occlusion dictionary (i.e. corrupted or contiguous occlusion and other variations), so that we can effectively eliminate those occlusions or corruption in both training and testing images. This system has provided some improvements on the face recognition performance compared to other approaches. However, it suffers from the limitation of RPCA which may not well generalize and preserve the identity of faces after removing occlusion. Moreover, well-aligned training images for each subject are required to build good dictionaries.

Thus, this chapter presents a robust generative model, called **Robust Deep Appearance Models** (RDAMs), that can separate unwanted factors while preserving identity information. The structure of RDAMs consists of two main components, i.e. the shape model and the texture model. Section 4.2 presents the shape modeling steps using DBM. The robust texture modeling using RDBM is introduced in section 4.3. Finally, our proposed robust fitting algorithms are presented in section 4.4.

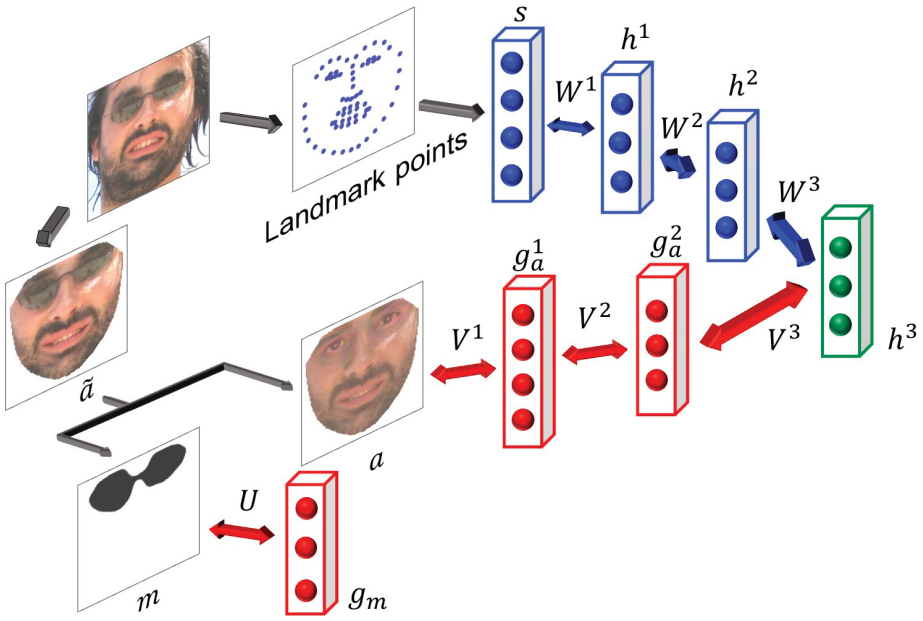

Figure 4.1: The diagram of our RDAMs approach. The blue layers present the shape model with a visible layer $\mathbf{s}$ and two hidden layers $\mathbf{h}^1$ and $\mathbf{h}^2$. The red layers denote the texture model with three visible units $\tilde{\mathbf{a}}$, $\mathbf{a}$ and $\mathbf{m}$, and three hidden layers $\mathbf{g}_m$, $\mathbf{g}_a^1$ and $\mathbf{g}_a^2$. The green layer denotes the appearance model consisting of a hidden layer $\mathbf{h}^3$

4.1 Overall Structure of RDAMs

Similar to DAMs [100], the structure of RDAMs also consists of two main components, i.e. the shape model and the texture model. Far apart from the texture model of DAMs, our texture model consists of a visible layer with three gating components: $\mathbf{a}$, $\tilde{\mathbf{a}}$, and $\mathbf{m}$, a binary RBM for the mask variable $\mathbf{m}$ and a Gaussian DBM with the real-valued input variable $\mathbf{a}$. The motivation for using this gating term is to improve modeling and fitting of the DAMs by eliminating the effects of missing, occluded or corrupted pixels. The schematic diagram of our proposed method is given in Fig. 4.1.

4.2 Shape Modeling

An n-point shape $\mathbf{s} = [x_1, y_1, \cdots, x_n, y_n]^T$ is modeled using a DBM with a visible layer and two hidden layers. Given a shape $\mathbf{s}$, the energy of the configuration $\{\mathbf{s}, \mathbf{h}^1, \mathbf{h}^2\}$ of the corresponding

layers in facial shape modeling is formulated as follows:

$$E_{DBM_s}(\mathbf{s}, \mathbf{h}^1, \mathbf{h}^2; \theta_s) = \frac{1}{2}\sum_i \frac{(s_i - b_{s_i})^2}{\sigma_{s_i}^2} - \sum_{i,j} W_{ij}^1 s_i h_j^1 - \sum_{j,l} W_{jl}^2 h_j^1 h_l^2 \tag{139}$$

where $\theta_s = \{\mathbf{W}^1, \mathbf{W}^2, \sigma_s, \mathbf{b}_s\}$ are the shape model parameters. The bias terms of hidden units in two layers in Eqn. (139) are ignored to simplify the equation. The probability distribution of the configuration $\{\mathbf{s}, \mathbf{h}^1, \mathbf{h}^2\}$ is computed as:

$$P(\mathbf{s}; \theta_s) = \sum_{\mathbf{h}^1, \mathbf{h}^2} \frac{\exp\left(-E_{DBM_s}\left(\mathbf{s}, \mathbf{h}^1, \mathbf{h}^2; \theta_s\right)\right)}{Z(\theta_s)} \tag{140}$$

where $Z(\theta_s)$ is the normalization constant. This shape model is pre-trained using one-step contrastive divergence (CD) learning.

4.3 Texture Modeling

Inspired by both RoBM [119] and DBM [114], we propose a new texture model approach named Robust Deep Boltzmann Machines. Our approach uses a DBM to model "clean" data $\mathbf{a}$ instead of a Gaussian RBM. There are good reasons for using DBM here. Firstly, it can efficiently capture variations and structures in the input data. Secondly, DBM can deal with ambiguous inputs more robustly due to its top-down feedback.

4.3.1 Robust Deep Boltzmann Machines

Given a shape-free image $\tilde{\mathbf{a}}$, the energy function of the configuration $\{\mathbf{a}, \tilde{\mathbf{a}}, \mathbf{m}, \mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2\}$ in facial texture modeling is optimized as follows:

$$\begin{aligned} E_{RDBM_g}(\mathbf{a}, \tilde{\mathbf{a}}, \mathbf{m}, \mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2; \theta_a) = & \sum_i \frac{\gamma_i^2 m_i (a_i - \tilde{a}_i)^2}{2\sigma_{g_i}^2} \\ & - \sum_{i,k} U_{ik} m_i g_{mk} + \sum_i \frac{(\tilde{a}_i - \tilde{b}_{g_i})^2}{2\tilde{\sigma}_{g_i}^2} \\ & + \sum_i \frac{(a_i - b_{g_i})^2}{2\sigma_{g_i}^2} - \sum_{i,j} V_{ij}^1 a_i g_{aj}^1 - \sum_{j,l} V_{jl}^2 g_{aj}^1 g_{al}^2 \end{aligned} \tag{141}$$

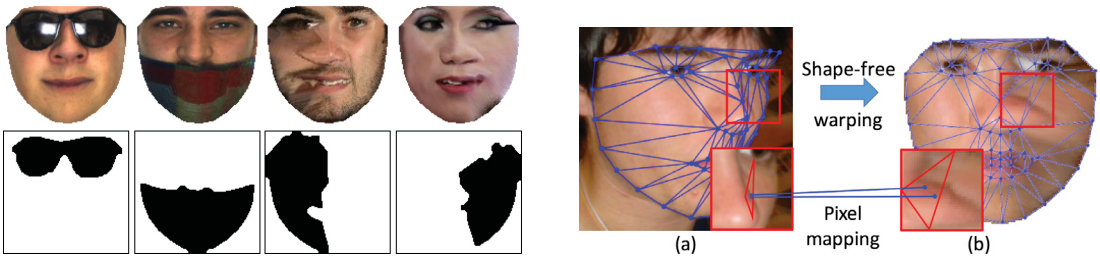

Figure 4.2: **LEFT**: Examples of automatically detected masks from the shape-free images. Top row: shape-free images. Bottom row: detected binary masks using the technique in section 4.3.3, **RIGHT**: An illustration in pose stretching detection: (a) Source image (b) Target warped shape-free image

where $\theta_a = \{\mathbf{V}^1, \mathbf{V}^2, \mathbf{U}, \sigma_g, \mathbf{b}_g, \tilde{\sigma}_g, \tilde{\mathbf{b}}_g\}$ are the texture model parameters. It is noted that all the bias terms in Eqn. (141) are ignored for simplicity. The probability distribution of the configuration $\{\mathbf{a}, \tilde{\mathbf{a}}, \mathbf{m}, \mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2\}$ is computed as follow:

$$P(\tilde{\mathbf{a}}; \theta_a) = \sum_{\mathbf{g}_a^1, \mathbf{g}_a^2} \frac{\exp\left(-E_{RDBM_g}\left(\mathbf{a}, \tilde{\mathbf{a}}, \mathbf{m}, \mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2; \theta_a\right)\right)}{Z(\theta_a)} \tag{142}$$

Given the input variables $\tilde{\mathbf{a}}$, the states of all layers can be inferred by computing the posterior probability of the latent variables, i.e. $p(\mathbf{a}, \mathbf{m}, \mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2 | \tilde{\mathbf{a}})$. Therefore, the sampling can be divided into two folds, i.e. one for the visible units and one for the hidden units. For the visible variables $\mathbf{a}$ and $\mathbf{m}$, the conditional distributions can be sampled as,

$$p(\mathbf{a}, \mathbf{m} | \mathbf{g}_m, \mathbf{g}_a^1, \tilde{\mathbf{a}}) = p(\mathbf{a} | \mathbf{m}, \mathbf{g}_a^1, \tilde{\mathbf{a}}) p(\mathbf{m} | \mathbf{g}_m, \mathbf{g}_a^1, \tilde{\mathbf{a}}) \tag{143}$$

For the hidden variables $\mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2$, the conditional distributions can be sampled as follows,

$$p(\mathbf{g}_m, \mathbf{g}_a^1, \mathbf{g}_a^2 | \mathbf{a}, \mathbf{m}, \tilde{\mathbf{a}}) = p(\mathbf{g}_m | \mathbf{m}) p(\mathbf{g}_a^1 | \mathbf{a}, \mathbf{g}_a^2) p(\mathbf{g}_a^2 | \mathbf{g}_a^1) \tag{144}$$

The sampling process can be applied on each unit separately since the distribution is factorial. Section 4.3.2 will discuss the learning procedure of this texture model.

4.3.2 Model Learning for RDBM

To pre-train our presented RDBM model, the DBM, which models "clean" faces, is first trained with some "clean" images and then the parameters in the RDBM model are optimized to maximize the log likelihood as follows,

$$\theta_a^* = \arg\max_{\theta_a} \log P(\tilde{\mathbf{a}}; \theta_a) \tag{145}$$

The optimal parameter values can then be obtained using a gradient descent procedure given by,

$$\frac{\partial}{\partial \theta_a}\mathbb{E}\left[\log P(\tilde{\mathbf{a}};\theta_a)\right] = \mathbb{E}_{P_{\text{data}}}\left[\frac{\partial E_{\text{RDBM}_g}}{\partial \theta_a}\right] - \mathbb{E}_{P_{\text{model}}}\left[\frac{\partial E_{\text{RDBM}_g}}{\partial \theta_a}\right] \tag{146}$$

where $\mathbb{E}_{P_{\text{data}}}[\cdot]$ and $\mathbb{E}_{P_{\text{model}}}[\cdot]$ are the expectations respecting to data distribution and distribution estimated by the RDBM. The two terms can be approximated using mean-field inference and Markov Chain Monte Carlo (MCMC) based stochastic approximation, respectively.

In our method, pre-training the parameters of the DBM on "clean" data first will make the process of learning the texture model faster and much easier. Similarly, we also propose to first learn the parameters of the binary RBM (to represent the mask $\mathbf{m}$) on pre-defined and extracted masks (as shown in Fig.4.2-LEFT) instead of randomizing the parameters. Then, the next question is how to generate the training masks from the training set. An automatic technique is presented to extract such training masks for the binary RBM in the next section 4.3.3.

4.3.3 Learning Binary Mask RBM

This section aims to generate masks from the training images having poses and occlusions, e.g. sunglasses and scarves. We consider learning three types of binary mask, i.e. sunglasses, scarves and pose stretching. A binary RBM is learned to represent each type of mask. We will focus on the last type, i.e. pose stretching since it is the hardest.

In 2D texture model, warping faces with a large pose (e.g. larger than $\pm 45^{\circ}$) will likely cause stretching effects on half of the faces since the same pixel values are copied over a large region (see Fig. 4.2-RIGHT). Therefore, we propose a technique that can detect such stretching regions during warping process. The main idea is to count the number of unique pixels in the source triangle that are mapped to the pixels in the target triangle. As we know, a source pixel can be mapped to multiple target pixels due to interpolation. The degree of a target triangle being stretched is equivalent to $p = (\frac{n_0}{N})$, where $p = 1$ means there is no stretching, n_0 and N are the number of unique pixels and the total number of pixels in the corresponding source triangle, respectively. Finally, we can use the detected regions as a mask to pre-train the above robust texture model.

4.4 Model Fitting in RDAMs

With the trained shape and texture models, the process of finding an optimal shape of a new image I can be formulated as finding an optimal shape $\mathbf{s}$ that maximizes the probability of the shape-free image as $\mathbf{s}^* = \arg\max_{\mathbf{s}} P(I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))|\mathbf{s}; \theta)$.

During the fitting steps, the states of hidden units $\mathbf{g}_a^1$ are estimated by clamping both the current shape $\mathbf{s}$ and the warped texture $\tilde{\mathbf{a}}$ to the model. The Gibbs sampling method is then applied to find the optimal estimated "clean" texture $\mathbf{a}$ of the testing face given the current shape $\mathbf{s}$. Let $\mathbf{a} = \sigma_g \mathbf{V}^1 \mathbf{g}_a^1 + \mathbf{b}_g$ be the mean of the Gaussian distribution, we have $P(I(W(r_{\mathcal{D}}, \mathbf{s}))|\mathbf{g}_a^1; \theta) = \mathcal{N}(\mathbf{a}, \sigma_g^2 \mathbf{I})$ where $\mathbf{I}$ is the identity matrix. The maximum likelihood can then be estimated as $\mathbf{s}^* = \arg\max_{\mathbf{s}} \mathcal{N}(I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))|\mathbf{a}, \sigma_g^2 \mathbf{I}) = \arg\min_{\mathbf{s}} \frac{1}{\sigma_g^2} \|I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}\|^2$.

This brings us to the non-linear least squares problem solved in image alignment. Notice that $\mathbf{a}$ is the reconstructed "clean" texture while $I(W(r_{\mathcal{D}}, \mathbf{s}))$ is the warped texture from the input image. If the input image contains occlusion or corruption, it is clear that the above square error will not reflect the goodness of the current shape $\mathbf{s}$. Thus, solely using ℓ_2-norm may limit the performance of shape fitting and reconstruction of the models. Since our proposed model can generate a mask of corrupted pixels, we propose to incorporate the mask $\mathbf{m}$ into the original objective function as:

$$\mathbf{s}^* = \arg\min_{\mathbf{s}} \|\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a})\|^2 \tag{147}$$

where $\odot$ is the component-wise multiplication. There are four main types of analytic shape fitting approaches: forward additive, forward compositional, inverse compositional and bi-directional. The modified forward additive, forward compositional and inverse compositional algorithms are introduced in sections 4.4.1, 4.4.2 and 4.4.3, respectively.

4.4.1 Forward Additive Algorithm

Forward Additive algorithm, also known as Lucas-Kanade algorithm, was first proposed for image alignment by Lucas and Kanade [87]. The idea of the algorithm is to find the best warp parameters that minimize the sum of squares error between a fixed template image and an input image I when warped. The warp parameters are iteratively updated by adding $\Delta\mathbf{s}$ each time, thus, the algorithm is considered as an *additive* approach. Using this idea, we solve the problem in Eqn.

(147) by linearizing it and then solve it iteratively with respect to an increment of the parameters $\Delta\mathbf{s}$. Then we minimize the following:

$$\Delta\mathbf{s} = \arg\min_{\Delta\mathbf{s}} \|\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) + \mathbf{J}_I \Delta\mathbf{s} - \mathbf{a}) \|^2 \tag{148}$$

where $\mathbf{J}_I = \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ is the Jacobian matrix of the image I.

The first step is to optimize Eqn. (148) with respect to $\Delta\mathbf{s}$ and then update $\mathbf{s} \rightarrow \mathbf{s} + \Delta\mathbf{s}$. This gives us the following:

$$\Delta\mathbf{s} = \mathbf{H}^{-1}\mathbf{J}_I^T(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a})) \tag{149}$$

where the Hessian matrices $\mathbf{H}$ are given by

$$\mathbf{H} = (\mathbf{m} \odot \mathbf{J}_I)^T(\mathbf{m} \odot \mathbf{J}_I) \tag{150}$$

In general, the computations of Hessian and Jacobian matrices are the costliest steps and they need to be re-computed at each iteration. Thus, the Lucas-Kanade algorithm is slow. The modified Forward Additive algorithm with the use of a mask $\mathbf{m}$ is summarized in Algorithm 7.

Algorithm 7 – Forward Additive

1. **Pre-compute:** the gradient, the Jacobian and the Hessian matrices need to be recomputed at each iteration.
2. **At each iteration:**
   (I) Perform warping operator $\mathbf{W}$ to obtain warped texture $I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))$
   (II) Compute the texture reconstruction error $(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (III) Compute $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{s}}(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (IV) Compute the Hessian matrix using Eqn. (150)
   (IV) Compute Δs using Eqn. (149)
   (IV) Update new shape as $\mathbf{s} \rightarrow \mathbf{s} + \Delta\mathbf{s}$

4.4.2 Forward Compositional Algorithm

For computing the warp parameters, the forward additive or Lucas-Kanade algorithm estimates a small offset from the current warp parameters. In the compositional algorithms, the composition of an incremental warp and the current warp is computed instead. Applying to our problem in Eqn. (147), we have the following minimization problem:

$$\Delta\mathbf{s} = \arg\min_{\Delta\mathbf{s}} \|\mathbf{m} \odot (I(\mathbf{W}(\mathbf{W}(r_{\mathcal{D}}, \Delta\mathbf{s}), \mathbf{s})) - \mathbf{a}) \|^2 \tag{151}$$

The forward compositional algorithm can be used to solve the problem in Eqn. (151) by first linearizing the image I around $\mathbf{s}$. An update $\Delta\mathbf{s}$ is found using least-squares, and $\mathbf{s}$ is updated from

$\mathbf{s} \leftarrow \mathbf{s} \circ \Delta\mathbf{s}$, where $\circ$ denotes the composition of two warps. Noting that the algorithm is processed with occluded/missing data being ignored while computing the residual error. The linearization applied to the test image side via first order Taylor expansion gives us:

$$\Delta\mathbf{s} = \arg\min_{\Delta\mathbf{s}} \|\mathbf{m} \odot (I(\mathbf{W}(\mathbf{W}(r_{\mathcal{D}}, 0), \mathbf{s})) + \mathbf{J}_I \Delta\mathbf{s} - \mathbf{a}) \|^2 \tag{152}$$

When $\mathbf{s} = 0$, we have an identity warp, i.e. $\mathbf{W}(r_{\mathcal{D}}, 0) = r_{\mathcal{D}}$. The key difference between forward additive and forward compositional is that the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ is computed at $(r_{\mathcal{D}}, 0)$. Thus, it is a constant and can be pre-computed. Not having to compute the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ in each iteration reduces the computational cost despite that the compositional update step is costlier.

Algorithm 8 – Forward Compositional

1. **Pre-compute:** The Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ at $(r_{\mathcal{D}}; 0)$
2. **At each iteration:**
   (I) Perform warping operator $\mathbf{W}$ to obtain warped texture $I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))$
   (II) Compute the texture reconstruction error $(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (III) Compute $\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{s}} (\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (IV) Compute $\Delta\mathbf{s}$ using Eqn. (149)
   (V) Update the shape parameters by composing the warp operator $\mathbf{s} \rightarrow \mathbf{s} \circ \Delta\mathbf{s}^{-1}$

4.4.3 Inverse Compositional Algorithm

The inverse compositional algorithm is a modification of the forward compositional algorithm where the roles of the model image and testing image are reversed. The inverse compositional algorithm tries to minimize the incremental warp computed with respect to the model image $\mathbf{a}$ instead of with respect to $I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))$. Changing the roles of $I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))$ and $\mathbf{a}$ in Eqn. (152) gives us

$$\Delta\mathbf{s} = \arg\min_{\Delta\mathbf{s}} \|\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}(\mathbf{W}(r_{\mathcal{D}}, \Delta\mathbf{s}))) \|^2 \tag{153}$$

with respect to $\Delta\mathbf{s}$ and then updating the parameters as $\mathbf{s} \leftarrow \mathbf{s} \circ \Delta\mathbf{s}^{-1}$, where $\circ$ denotes the composition of two warps. The solution of the least squares problem above is $\Delta\mathbf{s} = \mathbf{H}^{-1}\mathbf{J}_{\mathbf{a}}^T(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$ where $\mathbf{J}_{\mathbf{a}} = \nabla\mathbf{a}\frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ is the Jacobian matrix of the model image $\mathbf{a}$. The Hessian matrices $\mathbf{H}$ are then given by $\mathbf{H} = (\mathbf{m} \odot \mathbf{J}_{\mathbf{a}})^T(\mathbf{m} \odot \mathbf{J}_{\mathbf{a}})$.

Algorithm 9 – Inverse Compositional

1. **Pre-compute:** The gradient $\nabla \mathbf{a}$, the Jacobian $\frac{\partial \mathbf{W}}{\partial \mathbf{s}}$ at $(r_{\mathcal{D}}; 0)$, the steepest descent $SD = \nabla \mathbf{a} \frac{\partial \mathbf{W}}{\partial \mathbf{s}}$, the Hessian matrix $H = SD^T SD$
2. **At each iteration:**
   (I) Perform warping operator $\mathbf{W}$ to obtain warped texture $I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s}))$
   (II) Compute the texture reconstruction error $(\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (III) Compute $\nabla \mathbf{a} \frac{\partial \mathbf{W}}{\partial \mathbf{s}} (\mathbf{m} \odot (I(\mathbf{W}(r_{\mathcal{D}}, \mathbf{s})) - \mathbf{a}))$
   (IV) Compute $\Delta \mathbf{s}$ using Eqn. (153)
   (V) Update the shape parameters by composing the warp operator $\mathbf{s} \rightarrow \mathbf{s} \circ \Delta \mathbf{s}^{-1}$

4.5 Conclusion

In this chapter, the novel Robust Deep Appearance Models have been proposed to deal with large variations in the wild such as occlusions and poses. Moreover, the proposed fitting algorithms fit well with the new texture model such that it can make use of the occlusion mask generated by the proposed model.

Chapter 5

Experimental Results

This chapter compares our robust face recognition framework with other sparse representation based approaches. We also compare our LP-RPCA, OLP-RPCA, and RP-SVD methods against state-of-the-art algorithms in the problem of matrix decomposition and factorization. Finally, the proposed RDAMs approach is compared with the previous DAMs model to show our remarkable reconstruction results even when faces are occluded or having extreme poses.

5.1 Robust Face Recognition via Sparse and Low-rank Representation

This section presents experimental results to show the performance of the robust face recognition framework using RPCA and dictionary learning compared to other recent methods. All experiments are conducted using the two well-known databases: AR and Extended Yale B.

5.1.1 Datasets

We mainly test our approach using images in AR [89] and Extended Yale B (EYB) [48] face databases. AR database contains 100 subjects (50 male and 50 female) and each subject has 26 images (14 normal images with different lighting and expression, six occluded images with sunglasses and six for scarf). On the other hand, the EYB database contains images of 38 persons taken at 64 different illumination conditions and at 9 distinct viewpoints for each illumination condition except

the first 10 subjects only have one (frontal) viewpoint for each illumination.

5.1.2 Face Recognition with Standard Databases

The purpose of this experiment is to evaluate the recognition performance of our method. We compare our method with two representative methods including: SRC [130] and LR [28]. Since GSRC [134] is based on Gabor feature, we did not include it in our comparisons. Our goal is to compare with non-feature based methods only. We test those methods with real face disguises and small illumination variations on the AR database and illumination variations on the EYB database. We report the average recognition rates of a 5-fold Cross-validation.

AR Databases

Most of prior works use this database for evaluation their methods with occluded (sunglasses or scarf) images. We set up three scenarios similar to what is done in [28] where both neutral and corrupted images are used for training and testing. The three scenarios are called *sunglasses*, *scarf* and *sunglasses + scarf*. We added a fourth scenario which is *illumination + expression*. The size of original images is 165×120. PCA is applied to reduce their dimensionality to $r = 500$.

Sunglasses: We use a training set of 7 non-occluded images from session 1 and one randomly selected occluded (by sunglasses) images for each person. The algorithms are tested with a testing set of 7 non-occluded images from session 2 and 5 remaining occluded (by sunglasses) images for each person. In total, we train with 8 images and test with 12 images.

Scarf: Similar to the previous case, we also use a training set of 7 non-occluded images from session 1 and one randomly selected occluded (by scarf) images for each person. The algorithms are tested with a testing set of 7 non-occluded images from session 2 and 5 remaining occluded (by scarf) images for each person. In total, we train with 8 images and test with 12 images.

Sunglasses + scarf (Mixed): We also use a training set of 7 non-occluded images from session 1 and two randomly selected occluded images (one with sunglasses and one with scarf) for each person. The algorithms are tested with a testing set of 7 non-occluded images from session 2 and 10 remaining occluded (sunglasses or scarf) images for each person. In total, we train with nine images and test with 17 images.

Illumination + expression: We use a training set of 7 randomly selected non-occluded images for each person and a testing set of 7 randomly selected non-occluded images for each person. In total, we train with 7 images and test with 7 images.

The recognition rates compared with other methods are shown in Table 5.1

Table 5.1: Recognition rates of our method and other methods on the AR database

Dim 500	**Illumination + expression**	**Sunglasses**	**Scarf**	**Mixed**
SRC [130]	97.29 %	87.53%	77.33%	78.27%
LR[28]	98.08%	88.15%	78%	79.31%
LR+SI[28]	97.91%	88.2%	77.83%	80.87%
LRR*[139]	- -	87.3%	83.4%	82.4%
SSRC [32]	98.8 %	**94.22%**	89.25 %	90.57%
Ours	**99.09%**	93.95%	**91.72%**	**91.27%**

In all experiments, the recognition rates of our method are higher than the SSRC [32], LR [28] and SRC [130]. Moreover, the results of our method drop less significantly than other methods as more and more difficult testing images are used. For example, our method only drops about 2 - 5% (i.e. from 99.09 % to 93.95% and 93.95% to 91.72%) in the first three scenarios while other methods decrease from 4 - 10% in those three scenarios. This shows that our method is more robust to occlusions than others. Images occluded by scarf (40% occlusion) are more difficult to recognize than images occluded by sunglasses (20% occlusion). Therefore, the recognition rate of the experiment with scarf should be the lowest. The experiment with a mix of sunglasses and scarf should be the average of the sunglasses and scarf scenarios. However, when we mix images with sunglasses or scarf in the training and testing set, the results of our method in this scenario is the lowest among the four scenarios. This is because our method is based on an occlusion dictionary. It is more difficult to sparsely represent testing images over a mixed occlusion dictionary.

We did re-implement all the methods in Table 5.1 except the method in [139]. Compared to [139], the results as quoted in their paper are much lower than ours even they used higher dimensional inputs (2200). With regard to [32] our results are either comparable (for sunglasses) or better (for the other cases). Finally, from this experiment, it is worth noticing that LR [28] method with incoherence structural does not help improving the recognition rate much. It is totally different from what is claimed in [28].

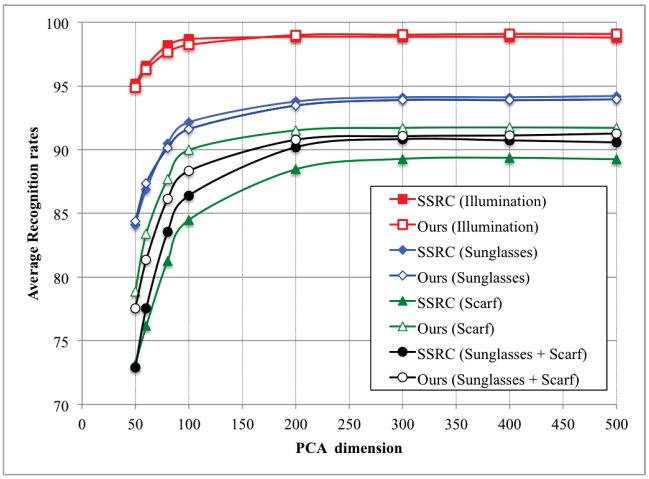

Figure 5.1: Comparison recognition rates between SSRC and our method under different scenarios on the AR database

We also did a quick comparison with the two latest works [139] and [32]. Compared to [139], our results are better (for sunglasses) or comparable (for the other cases). With regard to [32] the results as quoted in their paper are either similar to or better than ours.

YaleB

The extended Yale B (EYB) database is a commonly used database. We set up the experiment on EYB database similar to what is done in [28]. Only frontal images taken under varying illumination conditions are used for training and testing. The database is randomly split into two halves, one for training and one for testing. Each half contains 32 images for each subject.

Table 5.2: Recognition rates of our method, LR, SRC and GSRC on the Extended YaleB database

Our method	**LR [28]**	**SRC [130]**	**SSRC [32]**
97.47%	95.05%	95.03%	96.67%

Again, the recognition rate of our method is the highest among all methods under comparisons. Since the training and testing images which are used here only contain illumination variations, it is similar to the case of illumination and expression in the AR database. However, the changing of illumination in EYB database is much more severe than in AR database. Moreover, EYB database contains several bad images (i.e. completely dark) due to image aquisition process. Therefore, in general, the recognition rates on EYB are not as high as on the AR database. The basic SRC

method [130] has the lowest rate of 94.25% while LR method [28] could improve not much over this. Compared to [32], our results are better since our method is more robust to corruptions in this database.

5.1.3 Computational Time

Besides having good recognition rate, computational time is another crucial aspect for face recognition in real applications. Since the running time on testing phase is more important than on training phase, we only record the average time per one testing sample on the EYB database using a computer with Intel Core i7 3.4GHz and 8 GB RAM. All methods are re-implemented with Matlab 2012a.

Table 5.3: Average running time (seconds) of different methods on the AR and Extended YaleB database

Methods		**SRC [130]**	**SSRC [32]**	**Our method**
AR	**Time (s)**	0.18	0.16	0.07
	Dict's size	800	100 + 800	800 + 40
YaleB	**Time (s)**	0.23	0.22	0.10
	Dict's size	1216	38 + 1216	1216 + 40

From the Table 5.3, we can see that our method is faster than other methods under comparisons. Because all the methods used the same testing samples in each database, the computation time only depends on the size of dictionaries the methods used. Both SRC and LR methods use a sample dictionary with the size equal to the number of training samples. As a result, the running time of those two methods are close. Although our method uses two dictionaries i.e. the sample dictionary and occlusion dictionary, the learned dictionaries have the size considerably smaller than the other two methods. Therefore, the average running time of our method is less than others.

We conducted another experiment to show the effect of the size of the occlusion dictionary. Increasing the size of the dictionary could help improving the recognition rate, but more time is needed for testing. This is because one need to find the best representation over a bigger dictionary with more variables to optimize. The results are showed in Fig. 5.2.

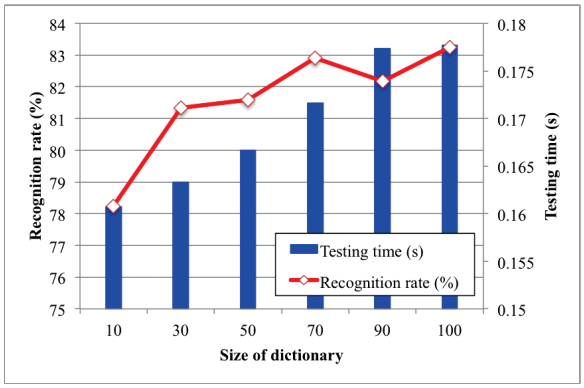

Figure 5.2: Relationship between recognition rate, testing time and the size of dictionary on the AR database

5.2 Matrix Decomposition: LP-RPCA and OLP-RPCA

In this section, we evaluate our proposed method in numerous applications, i.e. *matrix decomposition* on synthetic data, *face modeling*, *online background subtraction* and *video inpainting*.

In these experiments, the competing methods include various Robust PCA methods: ℓ_1-based representative methods, e.g. RPCA via inexact Augmented Lagrange Multiplier (ALM) Method [80], BRPCA [33], VBRPCA [2], PRMF (parallelized) [127] ; ℓ_p-based representative non-convex methods, e.g. NCADMM [24], pRost [57], NRPCA [98]; and online representative algorithms, e.g. OR-PCA [41], GRASTA [58], GOSUS [131], OPRMF [127]. All methods have Matlab codes available online, except for the NCADMM [24] method, which we re-implemented by ourselves. All the experiments were run on a system of Core i7@2.5GHz CPU,16.00GB RAM.

5.2.1 Evaluations on Synthetic Data

We conducted four experiments using synthetic data to evaluate the performance of our method (LP-RPCA) with different types of noise. First, we randomly generated a low-rank matrix $\mathbf{L}^\star \in \mathbb{R}^{400\times 400}$ with a rank $r = 20$. The matrix $\mathbf{L}^\star$ is computed as $\mathbf{L}^\star = \mathbf{A}\mathbf{B}^T$ where two random matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{400\times r}$ drawn from $\mathcal{N}(0,1)$. Then, we added the matrix $\mathbf{L}^\star$ and a sparse noisy matrix $\mathbf{S}^\star \in \mathbb{R}^{400\times 400}$ together to form an input matrix $\mathbf{M}$. We used different types of randomly generated sparse matrix $\mathbf{S}^\star$ in each experiment including: (1) *no noise* (Gaussian noise with $\mathcal{N}(0,0)$) (2) 10% of *uniform noise* ranged within $[-10, 10]$ (3) *Gaussian noise* with $\mathcal{N}(0,1)$ (4) *mixture* of 20%

Table 5.4: Results of the evaluation on synthetic data. The best results in terms of RE and time are highlighted in bold. The sign "–" represents given as an input.

Methods		**Convex approaches**			**Non-Convex approaches**			
		RPCA [80]	BRPCA [33]	VBRPCA [2]	NRPCA [98]	NCADMM [24]	pRost [57]	**LP-RPCA**
No noise	RE	3.8e-7±4.5e-7	1.8e-1±5.1e-2	3.9e-6±5.4e-8	8e-5±2.4e-4	2.1e-8±1.5e-8	**7.9e-16±1.2e-16**	3.9e-11±3.7e-12
	ER	20	400	20	–	20	–	**20**
	t(s)	0.44	46.02	0.55	**0.25**	1.82	3.22	0.3
Impulsive noise	RE	7.8e-7±2.4e-7	3.8e-3±8.8e-3	8.1e-6±2.7e-6	1.2e-3±4.5e-5	9.3e-7±3.1e-7	1.9e-4±6.2e-6	**6.3e-7±2.2e-7**
	ER	20	400	3.15	–	167	–	**20**
	t(s)	0.97	43.94	1.02	**0.34**	1.98	1.95	0.77
Gaussian noise	RE	1.8e-1±2.2e-3	2.2e-1±3.5e-3	1.7e-1±2.4e-3	1.5e-1±2.2e-3	2.4e-1±2.9e-3	1.9e-1±2.7e-3	**1.2e-1±1.3e-3**
	ER	232	400	20	–	400	–	**20**
	t(s)	2.00	44.07	6.6	**0.86**	1.73	2.56	2.23
Mixture noise	RE	1.6e-3±2.2e-5	7.9e-3±9.6e-3	6.2e-1±6.8e-3	1.8e-3±3.7e-5	1.1e-3±5.7e-6	1.3e-1±2.72e-3	**8.3e-4±1.4e-5**
	ER	230	400	3.05	–	237	–	**20**
	t(s)	1.57	43.93	6.6	**0.53**	1.88	5.12	3.68

uniform noise and 80% Gaussian noise $\mathcal{N}(0, 0.01)$. Finally, the matrix $\mathbf{M}$ was used as input for all competing methods to recover the original low-rank matrices $\mathbf{L}^\star$ and the sparse matrices $\mathbf{S}^\star$. Each experiment was run 20 times (with different generated matrices) to record the averaging results.

We evaluated the performance of all methods using three criteria: (1) *relative error* (**RE**): $\frac{\|(\hat{\mathbf{L}},\hat{\mathbf{S}})-(\mathbf{L}^\star,\mathbf{S}^\star)\|_F}{\|(\mathbf{L}^\star,\mathbf{S}^\star)\|_F+1}$ where $(\hat{\mathbf{L}}, \hat{\mathbf{S}})$ and $(\mathbf{L}^\star, \mathbf{S}^\star)$ denote the reconstructed matrices and the ground truth matrices, respectively. (2) *estimated rank* (**ER**): the rank of $\hat{\mathbf{L}}$ is computed based on SVD of $\hat{\mathbf{L}}$ as the number of eigenvalues greater than T, where $T = 400 \times \text{eps}(\|\hat{\mathbf{L}}\|_2)$ (eps is the floating-point relative accuracy). (3) *computational time* (***t***): average running time in seconds on each matrix.

All methods were run with the best parameters and the average results in each experiment are reported in Table 5.4. Except for RPCA, we chose $\lambda = \frac{1}{\sqrt{(400)}} = 0.05$ and for LP-RPCA, the parameter λ is set as $\lambda = \frac{1}{((p/2)*\sqrt{(400)})}$ (we empirically determined this formula) and the parameter p is chosen as described in the next section. The proposed LP-RPCA achieves better reconstruction results in terms of RE comparing with the other methods. Particularly, the advantage of LP-RPCA tends to be in the cases of impulsive noise, Gaussian noise and mixture noise. Because the ℓ_p-norm allows the hypothesis that the underlying noise is sparse to be violated (e.g. Gaussian and mixture noise) as compared to the ℓ_1-norm, this helps LP-RPCA handle Gaussian noise and mixture noise well enough. It is worth noting that although LP-RPCA does not perform as good as pRost in the noiseless case, it shows that our linearization scheme for ℓ_p-norm (i.e. the first-order Taylor expansion) brings the reconstruction result closer to the perfect reconstruction (i.e. the ideal ℓ_0 case) than the ℓ_1-norm. On the other hand, LP-RPCA always predicts the rank correctly. Since NRPCA and pRost methods require the rank to be provided before running the algorithms, ER does not apply to those two methods.

Selection of the parameter p

In this section, we show how the value of p affects the convergence properties and the reconstruction errors. The sketch of the convergence proof of our method is given in the Appendix. Here, we only show empirically the convergence properties of the LP-RPCA algorithm by using synthetic data. Theorem 1 (see Appendix A) shows that the objective function is monotonically decreasing. When a suitable parameter p is chosen, LP-RPCA algorithm converges fast and leads to an accurate

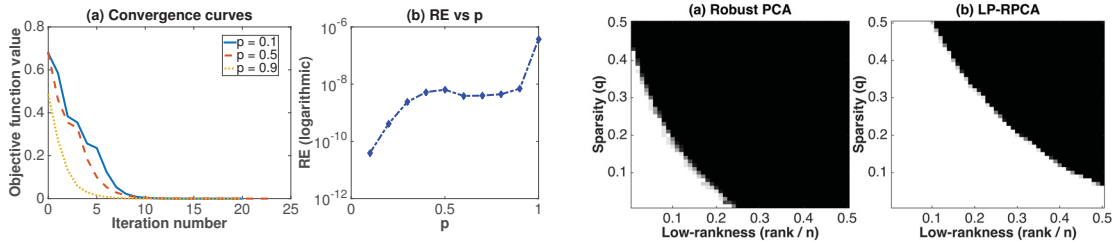

Figure 5.3: **LEFT**: Objective function value and relative error (RE) of LP-RPCA algorithm on the synthetic data while varying p. (a) Shows the convergence curves of LP-RPCA algorithm. (b) Shows the performance (RE) of of LP-RPCA algorithm. **RIGHT**: Illustration of successfully recovered cases for varying ranks and sparsity, computed by RPCA and LP-RPCA. Given a pair (r, q), the white region represents all the 10 folds are successfully recovered, and black means all folds are failed.

solution. Fig. 5.3-LEFT: (a) shows the objective function value for varying $p = 0.1, 0.5$ and 0.9. We also plotted the graph (see Fig. 5.3-LEFT (b)) illustrating how the performance in terms of RE changes while varying p, where $p \in [0.1, 1]$. From Fig. 5.3-LEFT (a) and (b), we can observe that a large value of p will lead to faster convergence, while a small value of p will lead to more accurate solution. Thus, p should not be too large nor too small. We empirically observe that $p = 0.1$ is a suitable value for synthetic data.

Phase Transition in rank and sparsity

The aim of this experiment is to demonstrate the recovery ability of our LP-RPCA method on various rank of matrices corrupted with different sparsity errors. We randomly generated a low-rank matrix $\mathbf{L}^\star \in \mathbb{R}^{m \times n}$ with a rank r. We considered $m = n = 400$ in this experiment. The matrix $\mathbf{L}^\star$ was computed as $\mathbf{L}^\star = \mathbf{A}\mathbf{B}^T$ where two random matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times r}$ drawn from $\mathcal{N}(0, 1/n)$. Then, for the sparse error matrix $\mathbf{S}^\star \in \mathbb{R}^{m \times n}$, the values of its entries were drawn from a Bernoulli distribution with a probability $1 - q$ for zero values and a probability $q/2$ for ± 1 values. Finally, we added the matrix $\mathbf{L}^\star$ and a sparse noisy matrix $\mathbf{S}^\star$ together to form an input matrix $\mathbf{M}$ and then decomposed $\mathbf{M}$ using **Algorithm 1**. For each experiment, i.e. each pair of (r, q), the algorithm was run 10 times (with different randomly generated matrices) to record the averaging results. An experiment is marked as being successful if the recovered $\hat{\mathbf{L}}$ satisfies $\frac{\|\hat{\mathbf{L}} - \mathbf{L}^\star\|_F}{\|\mathbf{L}^\star\|_F} \leq 10^{-3}$. We chose 50 values of $r \in [0.01, 0.5] \times n$ and 50 values of $q \in [0.01, 0.5]$ to compare with the original

RPCA method. The results are shown in Fig. 5.3-RIGHT. A larger white region in Fig. 5.3-RIGHT (b) means that our method can handle matrices with lower sparsity (i.e. less zero values) and higher rank. In other words, our LP-RPCA is more tolerant to the violation of the assumption that decomposed matrices are low-rank and sparse than RPCA. This is gained from the ℓ_p-norm that we used in our objective function.

5.2.2 Face Modeling

This experiment evaluates LP-RPCA in the face modeling application to remove unwanted factors, e.g. noise, shadows, darkness, etc., and produce better looking images. We used the extended Yale B face database, which consists of 64 face images (in different lighting conditions) per subject with the size of 64×64. For each subject, we created a data matrix $\mathbf{M} \in \mathbb{R}^{4096 \times 64}$. Each column of the matrix $\mathbf{M}$ is a face image of the corresponding subject. We then decompose the matrix as $\mathbf{M} = \mathbf{L} + \mathbf{S}$ where each column of $\mathbf{L}$ is a reconstructed face without the shadows. Fig. 5.4 shows typical reconstructed faces of the subject No. 13 from all methods. The average Peak Signal-to-Noise Ratio (PSNR) values between the normal frontal face and the reconstructed faces are shown for each method in Fig. 5.4. We chose the best parameters (p and λ) in the same way as described in the synthetic data evaluation section. Our proposed method and others are able to remove shadows and dark areas from the faces. However, our method performs better in the last two rows (types) in the way that it does not create any artificial effects on the faces. In addition to qualitative evaluation, we computed the average PSNR metric to provide a better insight for quantitative evaluation of the reconstructed faces. We chose the first face image of this subject without having any lighting condition, i.e. normal or standard illumination, as the reference image for calculating the PSNR. Our proposed method achieves the highest PSNR value among all competing methods. This shows that face reconstructed from our method is closer to the reference face image. Thus LP-RPCA method successfully eliminates shadows or lighting conditions from face images. Moreover, we can apply OLP-RPCA to reconstruct new images without running LP-RPCA method on the whole training data matrix $\mathbf{M}$ again. This is one of the potential applications of our online approach.

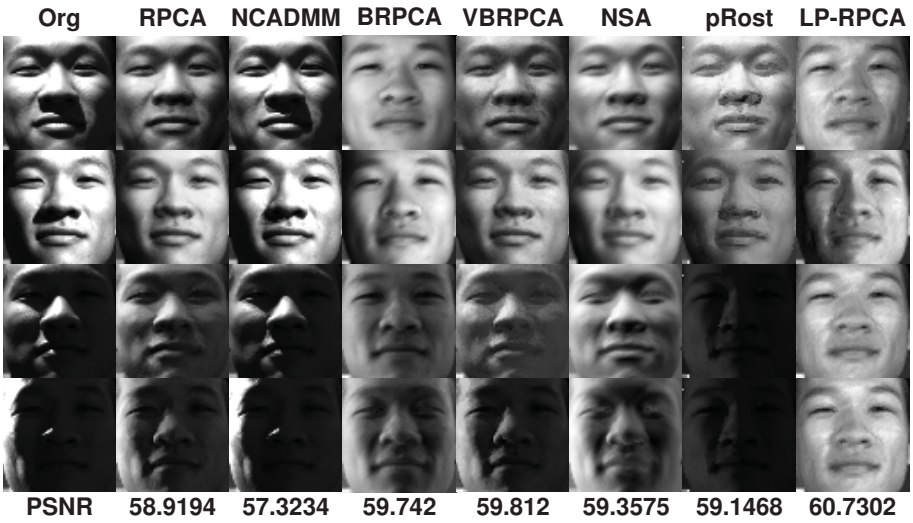

Figure 5.4: Columns from left to right: original face images of subject No. 13, reconstructed faces using ℓ_1-RPCA, non-convex ADMM (NCADMM), BRPCA, VBRPCA, NSA, pRost and our method (LP-RPCA). Rows from top to bottom: typical types of illumination.

5.2.3 Online Background Subtraction via OLP-RPCA

In this section, we evaluate our method on the online background subtraction. This application involves the detection of foreground and background in a video from a surveillance camera capturing moving objects in a frame-by-frame manner. Our method and other methods were tested on three types of real sequences: baseline and intermittent object motion from CDW-2014 dataset [54]. We also compared with other offline algorithms which process the whole video at every step. All benchmark results for both offline and online methods are shown in Fig. 5.5 and Table 5.5 with the average F-measures of the illustrated frames. We down-sampled all frames to 160×120 and used the same subspace dimension (rank $r = 2$) for all requiring methods (i.e. pRost, GRASTA, OR-PCA and OPRMF). Our offline and online methods both achieve the best F-measures for baseline and intermittent object motion categories. Although the precision of our methods is not as high as other methods, we have the highest recall rate. In other words, our methods capture more foreground pixels with a good accuracy (See Fig. 5.5) and result in a balanced F-measure.

From Tables 5.5 and 5.6, we can see that our online method processes each video frame in the shortest time ($0.003\text{s}\pm$ 6e-4 and $0.001\text{s}\pm$ 2e-5) despite the length of the input videos ranging from 1000 to 2000 frames. To empirically verify the complexity analysis in Section 3.3.3, we

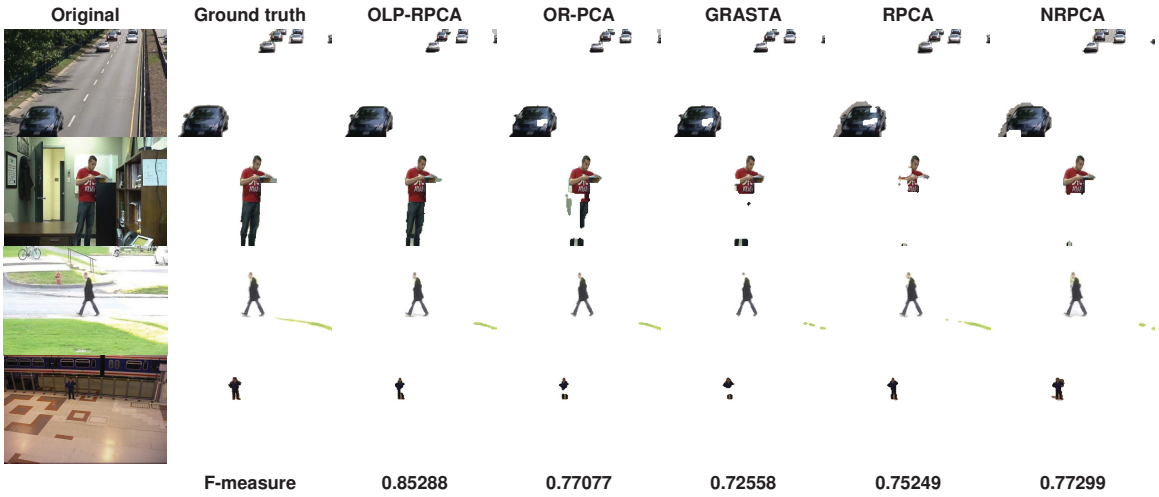

Figure 5.5: From top to bottom: the "highway", "office", "pedestrians" and "PETS2006"' video frames No. 690, 900, 630 and 880, respectively. From left to right: original frames, ground truth and foreground estimated by **OLP-RPCA** (online version), OR-PCA, GRASTA, RPCA and NRPCA.

Table 5.5: Average results of the background subtraction on baseline videos ("highway", "office", "pedestrians" and "PETS2006") (more than 1000 frames per video with the size of 160 $\times$ 120) in the dataset CDW 2014 [54]. TPF - Time per frame (second)

Methods	Recall	Precision	FMeasure	TPF (s)
Offline processing				
RPCA (Lin et al. 2010)	0.699	**0.856**	0.739	0.16
PRMF (Wang et al. 2012)	0.827	0.772	0.793	**0.014**
VBRPCA (Babacan et al. 2012)	0.753	0.855	0.779	0.23
pROST (Hage et al. 2014)	0.792	0.809	0.785	4.065
NRPCA (Netrapalli et al. 2014)	0.688	0.782	0.702	0.035
LP-RPCA	**0.845**	0.793	**0.81**	0.139
Online processing				
OR-PCA (Feng et al. 2013)	0.693	**0.942**	0.791	0.007
GRASTA (He et al. 2012)	0.637	0.857	0.728	0.032
GOSUS (Xu et al. 2013)	0.784	0.399	0.492	1.298
OPRMF (Wang et al. 2012)	0.736	0.762	0.739	3.937
OLP-RPCA	**0.898**	0.829	**0.858**	**0.003**

conduct another experiment. In this experiment, we evaluated the complexity of online methods under various data scaling (see Fig. 5.6) on the video "PETS2006". The results show that the complexity of OLP-RPCA is linear in both the sample dimension and the number of samples since the processing time for each frame increases linearly as bigger frame size is processed.

Table 5.6: Average results of the background subtraction on intermittent object motion videos ("abandonedBox", "parking", "sofa", "streetLight", "tramstop", "winterDriveway") in the dataset CDW 2014 [54]

Methods	Recall	Precision	FMeasure	TPF (s)
Offline processing				
RPCA [80]	0.391	**0.72**	0.474	0.185
PRMF [127]	0.482	0.505	0.436	**0.008**
VBRPCA [2]	0.506	0.62	**0.504**	0.068
pROST [57]	0.551	0.393	0.381	0.212
LP-RPCA	**0.655**	0.351	0.349	0.068
Online processing				
OR-PCA [41]	0.655	0.351	0.349	0.068
GRASTA [58]	0.319	**0.448**	0.341	0.017
GOSUS [131]	0.444	0.3	0.307	0.175
OLP-RPCA	**0.739**	0.379	**0.428**	**0.001**

5.2.4 Video Inpainting via OLP-RPCA

In this section, we apply our OLP-RPCA method to video inpainting. Fig. 5.7-LEFT shows the results of applying our method on the video "jumping girl" taken from [129]. Video inpainting first needs a mask for the object being removed. This mask is usually created manually. In this experiment, we want to remove the moving (or jumping) girl in the video. Thus, to create the mask for inpainting, we used the foreground (i.e. moving objects) detected by our OLP-RPCA (see Fig. 5.5 for an example). However, the foreground may include some unwanted parts of the standing girl since her hands are moving. Thus, we used the provided mask of the waving girl from [129] to exclude the unwanted moving parts from the foreground. After having the correct mask, video inpainting finds suitable information to fill the masked areas. The background image (i.e. the low-rank matrix) obtained from OLP-RPCA method is used to fill the missing regions in each incoming frame. Thus, unwanted moving objects can be removed completely without leaving any artifact in the recovered regions. This experiment shows another potential application of our online approach.

Furthermore, we use the horizontal slices (or xt projection) [99] from the sequence to analyze and compare the input and the inpainted sequence (See Fig. 5.7-RIGHT). A horizontal slice (i.e. a row in the xt projection) is the projection along the x-axis of a frame and all slices are stacked to form the xt projection. As we can see that Fig. 5.7-RIGHT (a) shows the low-rank property of the input video (i.e. a large green area). We used this property to remove the jumping girl (highlighted

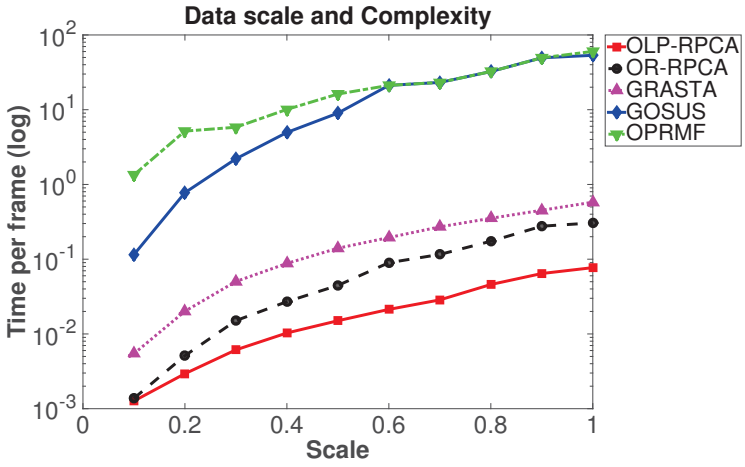

Figure 5.6: Processing time per frame **TPF** (seconds in log scale) of the online methods for several image scalings. Scaling is relative to 720×576 videos having 1200 frames.

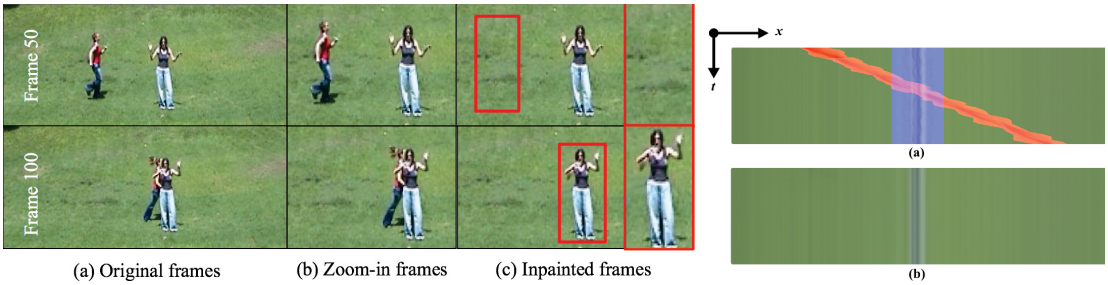

Figure 5.7: **LEFT**: Video inpainting application using the video "jumping girl" from [129]. Our OLP-RPCA method removes the moving girl while keeping the other girl and background without any artifact. **RIGHT**: (a) shows the xt projection of the input video with the position of the jumping girl (red) and the waving girl (blue) highlighted. (b) shows the xt projection of the inpainted video without any trace of the jumping girl.

in red) from the video while keeping the standing girl (highlighted in blue) (see Fig. 5.7-RIGHT (b)). In addition, a highlight video ("Video_inpainting_demo.mp4") attached in the supplementary material will emphasize the advantages of our method.

5.2.5 Image Denoising

This section aims at demonstrating the strength of our method (LP-RPCA) on image denoising. Two common types of noise are investigated: impulsive and Gaussian noise. We used three testing

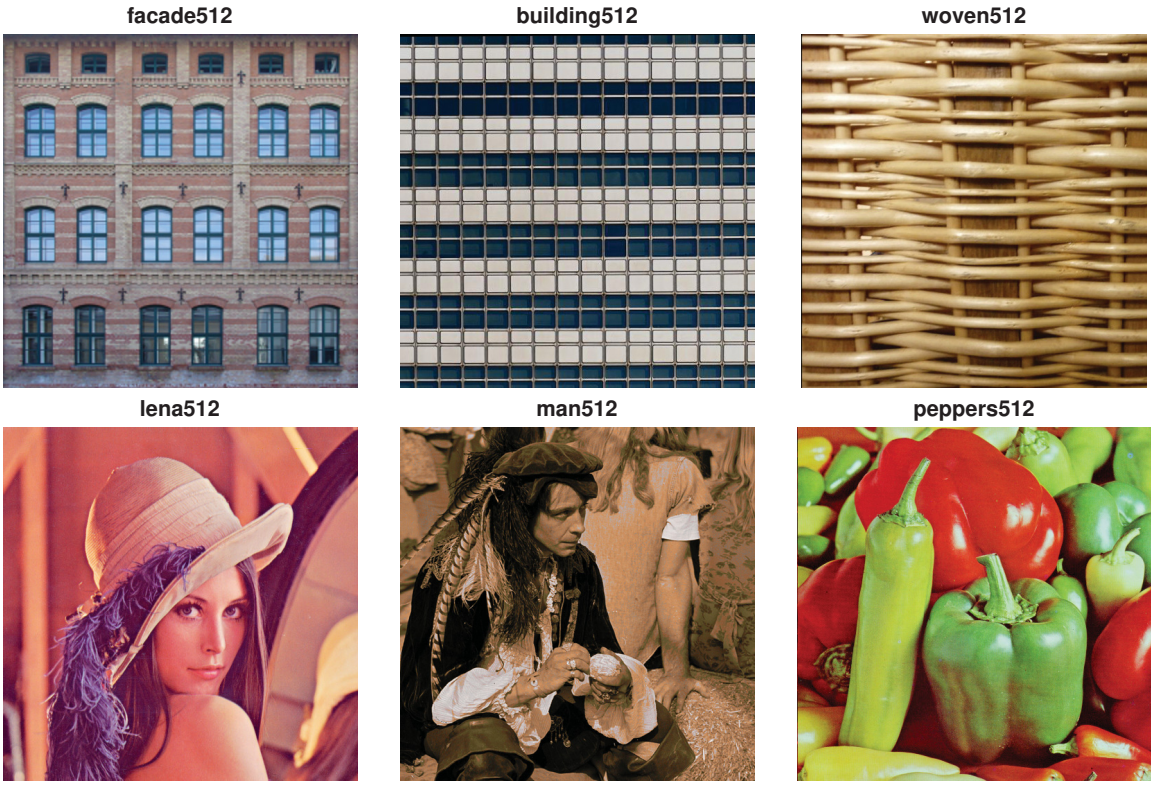

Figure 5.8: 1st row: our three testing images ("facade512", "building512" and "woven512"), 2nd row: three standard testing images ("lena512", "man512" and "pepper512")

Figure 5.9: Illustration of noisy and denoised images: 1^{st} *row* are text and Gaussian noise ($p_\sigma = 0.95$) added images, 2^{nd} *row* are denoised images using K-SVD, 3^{rd} *row* are denoised images using our method (LP-RPCA).

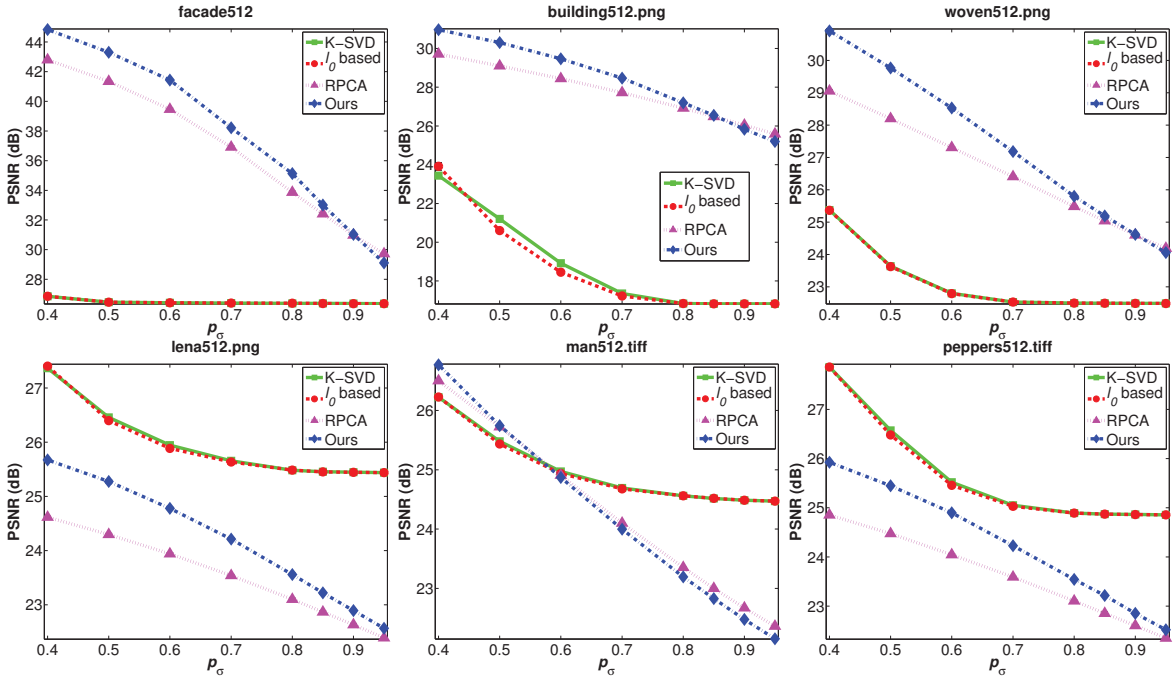

Figure 5.10: PSNR results for image denoising. Gaussian noise, taken up to 95% of the pixels in the testing image, was added. There are big differences in terms of PSNR in the first three images.

images ("facade512", "building512" and "woven512"), which show certain repeating patterns, together with three standard testing images ("lena512", "man512" and "pepper512") (see Fig. 5.8). All testing images have the size of 512×512. Our method successfully removes the text as well as the Gaussian noise added to all the testing images as shown in Fig. 5.9. The denoised results of the last three images are not as good as the first three due to the difference in image structures.

To show the influence of image structures on denoising algorithms, we conduct another experiment by adding various percentages of Gaussian noise to the testing images. We compared our method with two dictionary-based denoising methods: K-SVD based dictionary learning method [35], the ℓ_0-norm based dictionary learning method [5] and the low-rank approach: the ℓ_1 Robust PCA using ALM [80]. The PSNR values for all the methods are shown in Fig. 5.10. For the RPCA-based approaches, the matrix $\mathbf{M}$, which is the entire image, will be decomposed into a clean image $\mathbf{L}$ and noise $\mathbf{S}$. K-SVD and the ℓ_0-norm based dictionary learning method give higher PSNR results compared to two other RPCA approaches in the standard testing images. On the contrary, two RPCA approaches yield better PSNR results when denoising the images with patterns. This

shows that RPCA can denoise well on certain types of images (i.e. repeated pattern images) that have the low-rank properties. Our method improves the results of the original ℓ_1 RPCA and brings PSNR closer to the results of those dictionary based methods.

5.3 Matrix Factorization: RP-SVD

In this section, we will evaluated our proposed RP-SVD method on synthetic data and real-world data, i.e. face images and 3D structure from motion.

5.3.1 Synthetic Data

In this experiment, an input matrix $\mathbf{X}_0 \in R^{400\times 500}$ is randomly generated. Elements $\mathbf{X}_{0i,j}$ are drawn from an uniform distribution between $[-1, 1]$ independently. Some elements are then randomly selected as missing values by setting the corresponding entries in the mask matrix $\mathbf{M}$ to zeros. The missing ratio is set to 20% of the number of entries. In addition, in order to simulate outliers and/or noise, uniformly distributed noise over $[-5, 5]$ are added to 10% of the observed elements in $\mathbf{X}_0$ and Gaussian noise with $\sigma = 0.01$ are also added to all elements, respectively, to form a new matrix $\mathbf{X}$. The comparison algorithms, i.e. SVD (Matlab), ROBSVD [83], RSVD [65], ROBRSVD [137] and our proposed RP-SVD, factorize the noisy/outlier matrix $\mathbf{X}$ into subspaces. Then, the reconstructed matrices $\hat{\mathbf{X}}$ are computed. The reconstruction errors are measured as $\text{OER}_{\ell_1} = \|\mathbf{X}_0 - \hat{\mathbf{X}}\|_1/(m \times n)$. Table 5.7 shows the average errors and processing time (in second) on 500 different matrices $\mathbf{X}$. We also perform two experiments with various missing data and outlier ratios. First, the missing data ratios are set from 10% to 90%. The average ℓ_1-norm errors (OER_{ℓ_1}) over observed entries are recorded with the outlier ratios fixed at 20% of the observed entries. The first experiment is repeated 100 times for each level of missing data. Then, the missing data ratios are fixed to be 30%, and the outlier ratio is varied from 10% to 25%. Similarly, we repeat 100 times for each outlier ratio level. The results (the average ℓ_1-norm errors in log scale) of two experiments are shown in Fig. 5.11 (a).

Table 5.7: Evaluation Results on Synthetic Data.

Methods	$\mathbf{X}_0$		$\mathbf{X}_0$ + noise		$\mathbf{X}_0$ + outlier		$\mathbf{X}_0$ + noise + outlier	
	OER_{ℓ_1}	Time	OER_{ℓ_1}	Time	OER_{ℓ_1}	Time	OER_{ℓ_1}	Time
SVD	**1.6e-15** (±1.3e-16)	0.022	0.005 (±6.4e-6)	0.041	0.5 (±3e-3)	0.04	0.5 (±3.3e-4)	0.04
ROBSVD [83]	0.088 (±3e-3)	0.34	0.088 (±3e-3)	0.36	0.137 (±2e-3)	0.23	0.14 (±2e-3)	0.25
RSVD [65]	0.305 (±8e-3)	667.3	0.305 (±9e-3)	694	0.55 (±6e-3)	728	0.55 (±4e-3)	613
ROBRSVD [137]	0.3 (±8e-3)	884	0.302 (±9e-3)	677	0.33 (±8e-3)	795.4	0.33 (±8e-3)	771.4
RP-SVD	9e-9 (±1.9e-9)	1.53	**0.005** (±3.4e-5)	3.11	**8.11e-8** (±1.6e-8)	2.37	**0.005** (±2.8e-5)	3.13

5.3.2 Eigenfaces

One of the classical applications of SVD is facial images analyzing using eigenfaces. The eigenface discovers the underlying low K-dimensional subspace best describing the training data. In this experiment, we aim at showing the robustness of our RP-SVD method in reconstructing eigenface decomposition in the presence of outliers. A set of 30 randomly selected 64×64 face images from the Extended Yale B face database [48] are used as training set (i.e. a 4096×30 training data matrix). A 32×32 outlier image (i.e. an image of a football) is added to a random training image at a random location. The comparison methods, i.e. SVD, ROBSVD, RSVD and RP-SVD, are then applied to reconstruct the occluded facial image with $K = 10$. We repeat this procedure for 100 times. Fig. 5.11 (b) shows the resulting reconstructed facial images using those methods and the average PSNR also reported in this figure. Our method achieves the best PSNR value (52.79).

5.3.3 Structure from Motion

This experiment evaluates the proposed method in a real-world application named Structure from Motion. The standard Dinosaur sequence [1] containing projections of 195 points tracked over 36 views, was used in this experiment. Each tracked point is located in at least 16 views but their locations in other views are unknown. Thus, the measurement matrix has 74.26% of its elements missing and the originally measured tracks are illustrated in Fig. 5.12 (a). Fig. 5.12 (b), (c) and (d)

[1] available from http://www.robots.ox.ac.uk/~vgg/data/data-mview.html

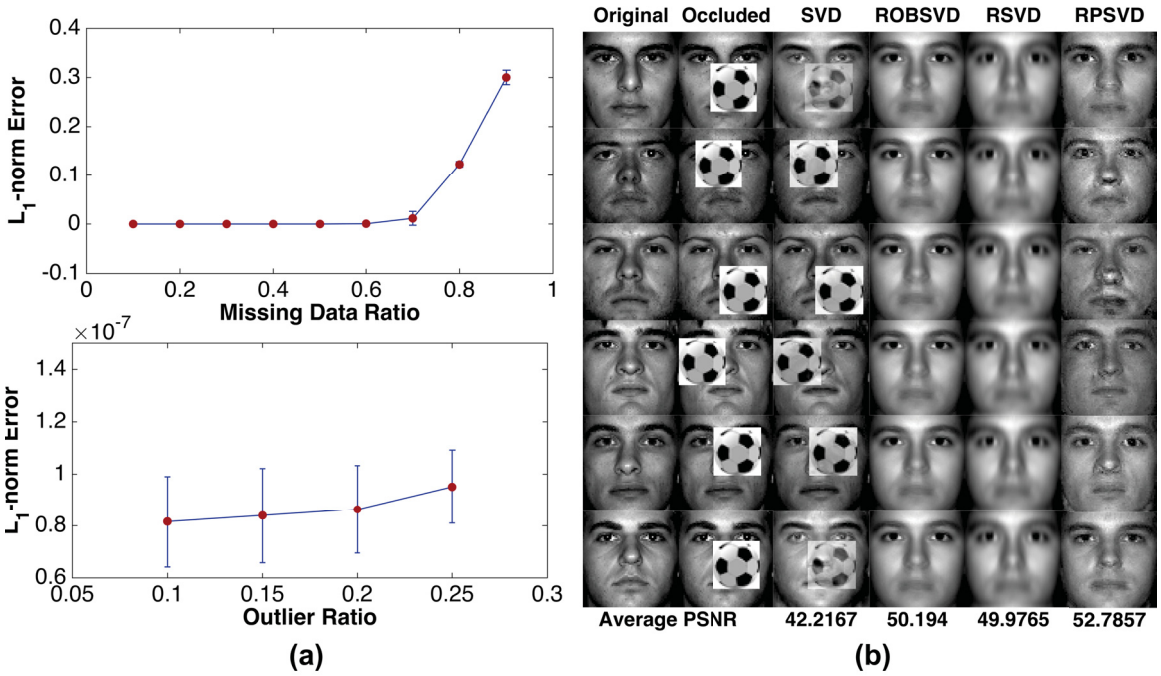

Figure 5.11: Experiments with outlier and missing data. (a) the average errors on synthetic data with varying missing data and outlier ratios. (b) An experiment on Extended Yale-B face database.

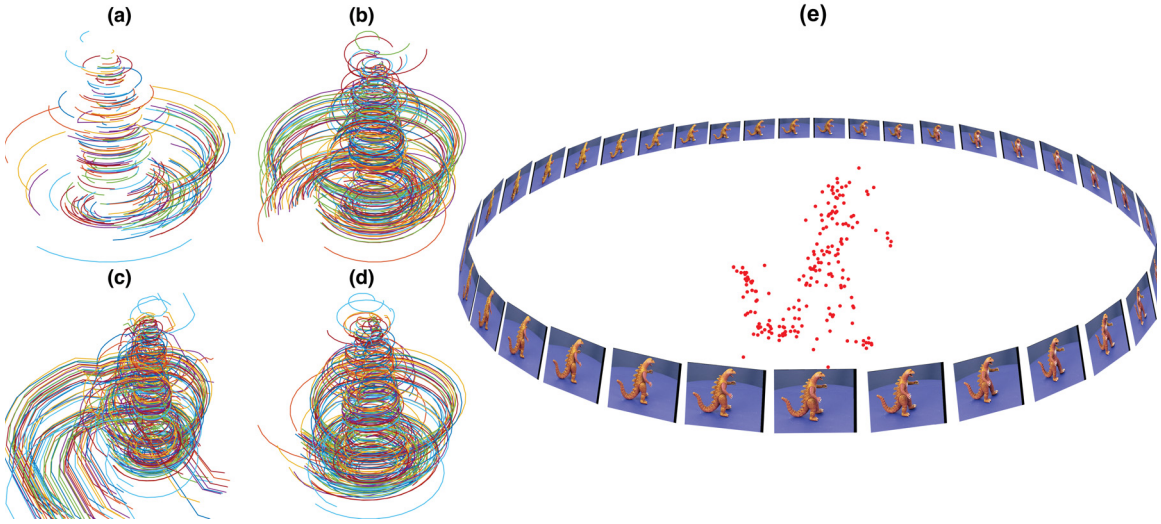

Figure 5.12: The experiment on the Dinosaur sequence reconstruction (a) shows the original tracks in the measurement matrix. (b) (c) and (d) show the recovered tracks using the Damped Newton [12], Damped Wiberg [101] and our RP-SVD method. (e) plots 3D reconstruct cloud points

shows the result obtained by Damped Newton [12] method, Damped Wiberg method [101] and our RP-SVD method, respectively. We should have close and circular tracks from Dinosaur sequence since the images of Dinosaur was captured while rotating on a table. Our method achieves the best

reconstructions with the completely closed circular tracks. Fig. 5.12 (e) illustrates 3D reconstructed points from tracked points in the Dinosaur sequence.

5.4 Robust Deep Appearance Models

In this section, we evaluate the performance of our proposed framework in face modeling tasks using data "in the wild" (sections 5.4.3 and 5.4.4).

5.4.1 Databases

The **LFPW** [7] database consists of 1400 images but only about 1000 images are available (811 for training and 224 for testing). For each image, we have 68 landmark points provided by 300-W competition [113].

The **Helen** [76] database contains about 2300 high-resolution images (2000 for training and 330 for testing). 68 landmark points are annotated for all faces. The facial images contain different poses, expressions and occlusions.

The **AR** database [89] contains 134 people (75 males and 59 females) and each subject has 26 frontal images (14 normal images with different lighting and expressions, six occluded images with sunglasses and six for scarves).

The **EURECOM** database [93] consists of facial images of 52 people (38 males and 14 females). Each person has different expressions, lighting and occlusion conditions. We only use images wearing sunglasses in our experiments.

5.4.2 RDAMs: Model Training

RDAMs are trained in two steps: pre-train each layer and train the whole model. The training set includes 1000 clean and 200 posed images from LFPW and Helen, 534 clean, 95 sunglasses, and 95 scarf images from 95 subjects in AR, 104 images from 52 subjects in EURECOM. For the pre-training steps, we first train shape DBM using all shapes. Then, we train RDBM by first separately training GRBM with clean images and learning binary mask RBM with masks generated from occluded and posed images in AR, EURECOM or LFPW. After that, we can train the RDBM

with pre-initialized weights of GRBM and mask RBM. The joint layer is later trained with all training images. Finally, the whole model is trained to update its weights. Each step above is trained using Contrastive Divergence learning in 600 epochs on a system of Xeon@3.6GHz CPU, 32.00GB RAM. The computational costs (without parallel processing) are as follows. The training time is **14.2** hours. Fitting on average is **17.4**s. Reconstructing faces on average is **1.53**s.

5.4.3 Facial Occlusion Removal

In this section, we demonstrate the ability of RDAMs to handle extreme cases of occlusions such as sunglasses or scarves. First, RDAMs is pre-trained using 1000 "clean" training images from LFPW and Helen database, 534 "clean" training images of 95 subjects (45 males and 50 females) from AR databases. Then, two texture models were trained using 95 images with sunglasses and 95 images with scarves, respectively. As shown in Fig. 5.13, RDAMs can remove those occlusions successfully without leaving any severe artifact comparing with the baseline AAMs method and the state-of-the-art DAMs method. We measure the reconstruction quality in terms of Root Mean Square Error (RMSE) on LFPW, Helen, AR and EURECOM databases in different ways.

In AR database, we choose two subsets of 210 images with sunglasses and 210 images with scarves from 38 subjects not in the training set, i.e. 30 males and eight females. The corresponding normal face images, i.e. frontal and without occlusions, of the same person are used as the references to compute the RMSE. In LFPW and Helen databases, we select a subset of 23 images with sunglasses and 100 images with some occlusions around the mouth. A mask is used to ignore occluded/corrupted pixels in the testing images so that we have an unbiased metrics.

The average masked-RMSEs of AAMs, DAMs and our RDAMs are shown in Table 5.8. The average unmasked-RMSEs are also reported for reference (i.e. the numbers inside the brackets).

Table 5.8: The average RMSEs of reconstructed images using different methods on LFPW and AR databases with sunglasses (SG) and scarf (SF)

Methods	AAMs [124]	DAMs [100]	**RDAMs**
LFPW	12.91 (18.98)	11.15 (14.98)	**8.58** (23.98)
AR - SG	56.55	55.48	**41.67**
AR - SF	63.16	60.96	**47.65**

Figure 5.13: Reconstruction results on images with occlusions (i.e. sunglasses or scarves) in LFPW, Helen and AR databases. The first row: input images, the second row: shape-free images, from the third to fifth rows: reconstructed results using AAMs, DAMs and RDAMs, respectively.

(a) (b)

Figure 5.14: (a) Facial pose recovery results on images from LFPW and Helen databases. The first row is the input images. The second row is the shape-free images. From the third to fifth rows are AAMs, DAMs and RDAMs reconstruction, respectively. (b) Example faces with significant variations, i.e. occlusions and poses, and the modeling results. From top to bottom: original images, shape free images, reconstructed faces using DAMs and reconstructed faces using our RDAMs approach.

Our RDAMs achieve the best reconstruction results compared against AAMs and DAMs. Note that the unmasked-RMSE is always higher than masked-RMSE since some corrupted pixels are recovered during reconstruction. Since our RDAMs can recover more corrupted/occluded pixels, it makes the un-masked RMSE higher than the ones from AAMs and DAMs.

5.4.4 Facial Pose Recovery

This section illustrates the capability of RDAMs to deal with facial poses. Using the same pre-trained model presented in Section 5.4.3, the texture model was trained using 280 images with different pose variations from LFPW and Helen databases. The reconstruction results of facial images with different poses are presented in Fig. 5.14a. In this experiment, our RDAMs also achieve the best reconstruction results comparing to AAMs and DAMs especially in the cases of extreme poses (more than $45°$). Our proposed RDAMs method can handle those extreme poses in a more natural way. From Fig. 5.14a, RDAMs give reconstructed faces that look more similar to the original faces while DAMs or AAMs make the face look younger or change its identity.

5.4.5 Model Fitting

The aim of this experiment is to evaluate the performance of different model fitting algorithms that are described in section 4.4. and to show that the use of mask could help improve model fitting rather than to compete with other works on the problem of face alignment. Our model fitting aims at finding the shape parameters that best minimize the reconstruction error. The best reconstruction error could result from shape parameters corresponding to the ground truth shape if the testing image was in the training set of the model. The initial shape is the mean shape placed inside the face's bounding box.

We evaluated our model fitting algorithms incorporating a corrupted pixel mask with the baseline fitting methods without using the mask on the LFPW and the AR databases. Three model fitting algorithms (i.e. Forward Additive (FA), Inverse Compositional (IC) and Forward Compositional (FC)) are compared on two types of occlusions including sunglasses (SG) and scarf (SF). The average errors are reported in Table 5.10.

We also compare our results with Active Orientation Models [125] and the method in [124] in the following modeling fitting experiment. We evaluated model fitting using AR database. The average errors are showed in Table 5.9. RDAMs achieve comparable performance compared to other methods.

Table 5.9: The average MSE between estimated shape and ground truth shape (68 landmark points). Tested on about 300 images (23 images from LFPW database and 268 images from AR database)

Method	SG	SF
Initialization	0.195	0.211
RDAMs with FC	0.1672	0.0756
Fast-SIC [124]	0.1218	0.0756
AOMs [125]	0.1705	0.0962

Table 5.10: The average MSE between estimated shape and ground truth shape (68 landmark points). Tested on about 300 images (23 images from LFPW database and 268 images from AR database)

Type	Method	Initial	With Mask	Without Maks
SG	FA	0.0406	**0.0353**	0.0361
	IC	0.0406	**0.038**	0.039
	FC	0.0406	**0.0372**	0.0373
SF	FA	0.0874	0.0873	**0.0849**
	IC	0.0874	**0.0853**	0.0864
	FC	0.0874	0.0873	**0.0849**

5.5 Conclusion

The experiments show that ℓ_p-norm can help to improve the results of matrix decomposition without sacrificing too much computational cost and the online version (OLP-RPCA) can be efficiently employed for online background subtraction and video inpainting in real-time. In addition, it is able to achieve real-time performance without parallelizing or implementing on a graphics processing unit. The proposed RP-SVD method is evaluated in various applications, i.e. noise and outlier removal, estimation of missing values, structure from motion reconstruction and facial image reconstruction. This chapter shows that RP-SVD method can achieve better results compared to the state-of-the-art SVD and its extensions, i.e. ROBSVD, RSVD and ROBRSVD. The proposed RDAMs are evaluated on occlusion removal and pose correction to show the robustness of the model against large occlusions and poses.

Chapter 6

Conclusion and Future Work

This chapter draws some conclusions, summarize the thesis' contributions and provide discussions on future directions related to the topics in this thesis.

6.1 Conclusions

In this thesis, two sets of approaches: conventional matrix decomposition and deep learning-based for image and video analysis are proposed.

For conventional approaches, this thesis first proposes a novel face recognition framework to make a better use of sparse components resulted from a low-rank matrix decomposition via Robust PCA in the training phase. Using the information captured from the training stage, we have successfully improved the testing stage of the face recognition process with the combination of low-rank approximation and sparse representation methods. We have presented experimental results showing the performance of our approach compared to other recent sparse representation based methods. All experiments are conducted using the two well-known databases: AR and Extended Yale B. Matrix decomposition approach, i.e. Robust PCA technique, has shown its potential in the face recognition framework. To further apply this technique in other applications, this thesis proposes the novel offline and online non-convex ℓ_p-norm based Robust PCA (LP-RPCA and OLP-RPCA) approaches for matrix decomposition, where $0 < p < 1$. The proposed OLP-RPCA and LP-RPCA approaches

have demonstrated the robustness and efficiency in various applications including real-time background subtraction, video inpainting, Gaussian/non-Gaussian image denoising and face modeling. In addition, a novel Robust ℓ_p-norm Singular Value Decomposition (RP-SVD) method for matrix factorization is proposed. The proposed RP-SVD is formulated as an ℓ_p-norm based penalized loss minimization problem where a robust loss function is employed to measure the reconstruction error of a low-rank matrix approximation of the data. The ADMM is then used to find appropriate solutions to this problem. The proposed method achieves better performance in face image reconstruction compared to the state-of-the-art SVD and its extensions, i.e. Robust SVD, Regularized SVD and Robust Regularized SVD, in various scenarios and the proposed method can also estimate missing values for structure from motion reconstruction.

For deep learning based approaches, this thesis proposes a novel Robust Deep Appearance Models to deal with large variations in the wild such as occlusions and poses. The main idea of the proposed model is to exploit the ability of RDBM to decompose and reconstruct a face with occlusion. Comparing with the previous DAMs model, the proposed approach can produce remarkable reconstruction results even when faces are occluded or having extreme poses. Moreover, the proposed fitting algorithms fit well with the new texture model such that it can make use of the occlusion mask generated by the proposed model. Experimental results in occlusion removal, pose correction and model fitting have shown the robustness of the model against large occlusions and poses.

6.2 Future Directions

Overall, the contributions in this thesis are major advancements in the direction of extracting useful features for image and video analysis problems with matrix decomposition and factorization approaches. This section provides future directions and discusses some open issues in image and video analysis and deep learning based framework. The aim of this thesis and the proposed directions is to improve the presented framework to achieve the ultimate goal of efficiency and robustness.

Video background subtraction with moving camera: there is an increasing demand for processing data captured by portable/handheld cameras since they are becoming more popular in different scenarios, e.g. police equipped with handheld devices. It would be beneficial to develop a more robust real-time OLP-RPCA that can handle dynamic background changes, i.e. in the case of moving cameras. Another direction is to incorporate ORP-SVD into OLP-RPCA to improve the performance of OLP-RPCA

Extracting and localizing facial micro-expression: Micro-expression is a special kind of facial expression which happens extremely rapid and brief. This type of expression is usually uncontrollable and reveals true emotion of a person. Thus, there are several applications using facial micro-expression such as medical studies/diagnosis, national safety and police interrogation. Matrix decomposition via OLP-RPCA can tackle this problem efficiently and provide better pre-processed features for later tasks, i.e. recognition/classification emotions.

Deep learning based matrix decomposition: The conventional matrix decomposition methods are linear methods because the low-rankness and sparsity are based on linear latent variable model. Therefore, separating matrices in which the data are from nonlinear latent variable model may not be effective. To handle the non-linear problem, Robust Deep Boltzmann Machines (RDBMs) were proposed for face modeling in Chapter 4. However, it only models occlusion or noise using a binary mask RBM while conventional matrix decomposition can separate facial features and occlusion/noise into two different matrices. One possible approach is to model occlusion or noise using a Gaussian RBM so that we can model occlusion or noise directly instead of just borrowing the idea of matrix decomposition for texture modeling.

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54 (11):4311–4322, 2006.

[2] S. D. Babacan, M. Luessi, R. Molina, and A. K. Katsaggelos. Sparse bayesian methods for low-rank matrix estimation. *IEEE Transactions on Signal Processing*, 60(8):3964–3977, 2012.

[3] P. Bai, H. Shen, X. Huang, and Y. Truong. A supervised singular value decomposition for independent component analysis of fMRI. *Statistica Sinica*, 18:1233–1252, 2008.

[4] C. G. Baker, K. A. Gallivan, and P. Van Dooren. Low-rank incremental methods for computing dominant singular subspaces. *Linear Algebra and its Applications*, 436(8):2866–2888, 2012.

[5] C. Bao, H. Ji, Y. Quan, and Z. Shen. ℓ_0 norm based dictionary learning by proximal methods with global convergence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014*, pages 3858–3865. IEEE, 2014.

[6] S. Becker, E. Candès, and M. Grant. TFOCS: Flexible first-order methods for rank minimization. In *SIAM Conference on Optimization*, 2011.

[7] P. N. Belhumeur, D. W. Jacobs, D. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011*, pages 545–552. IEEE, 2011.

[8] K. Border. The supergradient of a concave function. 2001. URL http://www.hss.caltech.edu/kcb/Notes/Supergrad.pdf.

[9] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In *International Conference on Machine Learning (ICML)*, volume 98, pages 82–90, 1998.

[10] M. Brand. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision*, pages 707–720. Springer, 2002.

[11] M. Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear algebra and its applications*, 415(1):20–30, 2006.

[12] A. M. Buchanan and A. W. Fitzgibbon. Damped newton algorithms for matrix factorization with missing data. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005*, volume 2, pages 316–322. IEEE, 2005.

[13] E. J. Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.

[14] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[15] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[16] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier analysis and applications*, 14(5-6):877–905, 2008.

[17] E. J. Candès, X. Li, Y. Ma, and J. Wright. Roust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[18] Y. Chahlaoui, K. Gallivan, and P. Van Dooren. Recursive calculation of dominant singular subspaces. *SIAM Journal on Matrix Analysis and Applications*, 25(2):445–463, 2003.

[19] Y. Chahlaoutf, K. A. Gallivant, and P. Van Dooren. An incremental method for computing dominant singular spaces. *Computational information retrieval*, 106:53, 2001.

[20] S. Chandrasekaran, B. Manjunath, Y.-F. Wang, J. Winkeler, and H. Zhang. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing*, 59(5):321–332, 1997.

[21] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

[22] R. Chartrand. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Processing Letters*, 14(10):707–710, 2007.

[23] R. Chartrand. Nonconvex compressed sensing and error correction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2007.*, volume 3, pages 889–892. IEEE, 2007.

[24] R. Chartrand. Nonconvex splitting for regularized low-rank + sparse decomposition. *IEEE Transactions on Signal Processing*, 60(11):5810–5819, 2012.

[25] R. Chartrand and V. Staneva. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24(3):035020, 2008.

[26] R. Chartrand and B. Wohlberg. A nonconvex admm algorithm for group sparsity with sparse groups. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2013*, pages 6009–6013. IEEE, 2013.

[27] R. Chartrand and W. Yin. Iteratively reweighted algorithms for compressive sensing. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2008*, pages 3869–3872. IEEE, 2008.

[28] C.-F. Chen, C.-P. Wei, and Y.-C. Wang. Low-rank matrix recovery with structural incoherence for robust face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, pages 2618–2625. IEEE, 2012.

[29] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61, 1998.

[30] X. Chen, D. Ge, Z. Wang, and Y. Ye. Complexity of unconstrained $\ell_2 - \ell_p$ minimization. *Mathematical Programming*, 143(1-2):371–383, 2014.

[31] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Interprettting Face Images using Active Appearance Models. In *Proc. of the* 3^{rd} *Intl. Conf. on Automatic Face and Gesture Recognition*, pages 300–305, 1998.

[32] W. Deng, J. Hu, and J. Guo. In defense of sparsity based face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013*, pages 399–406. IEEE, 2013.

[33] X. Ding, L. He, and L. Carin. Bayesian robust principal component analysis. *IEEE Transactions on Image Processing*, 20(12):3419–3430, 2011.

[34] W. Dong, G. Shi, X. Li, Y. Ma, and F. Huang. Compressive sensing via nonlocal low-rank regularization. *IEEE Transactions on Image Processing*, 23(8):3618–3632, August 2014.

[35] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[36] K. Engan, S. O. Aase, and J. H. Husoy. Method of optimal directions for frame design. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 1999*, volume 5, pages 2443–2446. IEEE, 1999.

[37] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

[38] M. Fazel. *Matrix rank minimization with applications*. PhD thesis, Stanford University, 2002.

[39] M. Fazel, H. Hindi, and S. P. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of the American Control Conference, 2003.*, volume 3, pages 2156–2162. IEEE, 2003.

[40] J. Feng, H. Xu, S. Mannor, and S. Yan. Online PCA for contaminated data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 764–772, 2013.

[41] J. Feng, H. Xu, and S. Yan. Online robust PCA via stochastic optimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 404–412, 2013.

[42] S. Foucart and M.-J. Lai. Sparsest solutions of underdetermined linear systems via ℓ_q-minimization for $0 < q < 1$. *Applied and Computational Harmonic Analysis*, 26(3):395–407, 2009.

[43] C. Gao, N. Wang, Q. Yu, and Z. Zhang. A feasible nonconvex relaxation approach to feature selection. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2011.

[44] G. Gasso, A. Rakotomamonjy, and S. Canu. Recovering sparse signals with a certain family of nonconvex penalties and dc programming. *IEEE Transactions on Signal Processing*, 57 (12):4686–4698, 2009.

[45] D. Ge, X. Jiang, and Y. Ye. A note on the complexity of l_p minimization. *Mathematical programming*, 129(2):285–299, 2011.

[46] D. Geman and C. Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.

[47] J. Geng, L. Wang, and Y. Wang. A non-convex algorithm framework based on dc programming and dca for matrix completion. *Numerical Algorithms*, pages 1–19, 2014.

[48] A. S. Georghiades, P. N. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[49] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[50] I. Gorodnitsky and B. Rao. A new iterative weighted norm minimization algorithm and its applications. In *IEEE Sixth SP Workshop on Statistical Signal and Array Processing, 1992.*, pages 412–415. IEEE, 1992.

[51] I. F. Gorodnitsky and B. D. Rao. Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, 45(3):600–616, 1997.

[52] I. F. Gorodnitsky, J. S. George, and B. D. Rao. Neuromagnetic source imaging with FOCUSS: a recursive weighted minimum norm algorithm. *Electroencephalography and clinical Neurophysiology*, 95(4):231–251, 1995.

[53] J. C. Gower and G. B. Dijksterhuis. *Procrustes problems*, volume 3. Oxford University Press Oxford, 2004.

[54] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection.net: A new change detection benchmark dataset. In *CVPR Workshops 2012*, pages 1–8. IEEE, 2012.

[55] R. Gribonval and M. Nielsen. Highly sparse representations from dictionaries are unique and independent of the sparseness measure. *Applied and Computational Harmonic Analysis*, 22 (3):335–355, 2007.

[56] H. Guo, C. Qiu, and N. Vaswani. An online algorithm for separating sparse and low-dimensional signal sequences from their sum. *IEEE Transactions on Signal Processing*, 62 (16):4284–4297, 2014.

[57] C. Hage and M. Kleinsteuber. Robust PCA and subspace tracking from incomplete observations using ℓ_0-surrogates. *Computational Statistics*, 29(3-4):467–487, 2014.

[58] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, pages 1568–1575. IEEE, 2012.

[59] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[60] G. E. Hinton and T. J. Sejnowski. Optimal perceptual inference. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1983*, pages 448–453, 1983.

[61] B. Hong, L. Wei, Y. Hu, D. Cai, and X. He. Online robust principal component analysis via truncated nuclear norm regularization. *Neurocomputing*, 175:216–222, 2016.

[62] R. Horst and N. V. Thoai. DC programming: overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

[63] Y. Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(9):2117–2130, 2013.

[64] J. Huang, J. L. Horowitz, and S. Ma. Asymptotic properties of bridge estimators in sparse high-dimensional regression models. *The Annals of Statistics*, pages 587–613, 2008.

[65] J. Z. Huang, H. Shen, and A. Buja. The analysis of two-way functional data using two-way regularized singular value decompositions. *Journal of the American Statistical Association*, 104(488), 2009.

[66] T. Ince, A. Nacaroglu, and N. Watsuji. Nonconvex compressed sensing with partially known signal support. *Signal Processing*, 93(1):338–344, 2013.

[67] S. Javed, S. H. Oh, A. Sobral, T. Bouwmans, and S. K. Jung. OR-PCA with MRF for robust foreground detection in highly dynamic backgrounds. In *Asian Conference on Computer Vision*, pages 284–299. Springer, 2014.

[68] S. Javed, T. Bouwmans, and S. K. Jung. Depth extended online RPCA with spatiotemporal constraints for robust background subtraction. In *Frontiers of Computer Vision (FCV), 2015 21st Korea-Japan Joint Workshop on*, pages 1–6. IEEE, 2015.

[69] S. Javed, A. Sobral, T. Bouwmans, and S. K. Jung. OR-PCA with dynamic feature selection for robust background subtraction. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 86–91. ACM, 2015.

[70] B. D. Jeffs and M. Gunsay. Restoration of blurred star field images by maximally sparse optimization. *IEEE Transactions on Image Processing*, 2(2):202–211, 1993.

[71] V. C. Klema and A. J. Laub. Singular Value Decomposition: Its Computation and Some Applications. *IEEE Transactions on Automatic Control*, 25(2):164–176, 1980.

[72] K. Knight and W. Fu. Asymptotics for lasso-type estimators. *Annals of statistics*, pages 1356–1378, 2000.

[73] K. Koh, S.-J. Kim, and S. Boyd. An interior-point method for large-scale ℓ_1-regularized logistic regression. *Journal of Machine learning research*, 8(7), 2007.

[74] K. Kreutz-Delgado and B. D. Rao. FOCUSS-based dictionary learning algorithms. In *International Symposium on Optical Science and Technology*, pages 459–473. International Society for Optics and Photonics, 2000.

[75] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009.

[76] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang. Interactive facial feature localization. In *European Conference on Computer Vision 2012*, pages 679–692. Springer, 2012.

[77] R. M. Leahy and B. D. Jeffs. On the design of maximally sparse beamforming arrays. *IEEE Transactions on Antennas and Propagation*, 39(8):1178–1187, 1991.

[78] H. Lee and J. Lee. Online update techniques for projection based robust principal component analysis. *ICT Express*, 1(2):5–8, 2015.

[79] A. Levey and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, 9(8):1371–1374, 2000.

[80] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

[81] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 663–670, 2010.

[82] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):171–184, 2013.

[83] L. Liu, D. M. Hawkins, S. Ghosh, and S. S. Young. Robust singular value decomposition analysis of microarray data. *Proceedings of the National Academy of Sciences*, 100(23): 13167–13172, 2003.

[84] C. Lu, J. Tang, S. Yan, and Z. Lin. Generalized nonconvex nonsmooth low-rank minimization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014*, pages 4130–4137, Jun 2014. doi: 10.1109/CVPR.2014.526.

[85] C. Lu, Z. Lin, and S. Yan. Smoothed low rank and sparse matrix recovery by iteratively reweighted least squares minimization. *IEEE Transactions on Image Processing*, 2015.

[86] C. Lu, C. Zhu, C. Xu, S. Yan, and Z. Lin. Generalized singular value thresholding. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2015.

[87] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence (IJCAI)*, volume 81, pages 674–679, 1981.

[88] G. Marjanovic and V. Solo. On ℓ_q optimization and matrix completion. *IEEE Transactions on Signal Processing*, 60(11):5714–5724, 2012.

[89] A. M. Martinez. The AR face database. *CVC Technical Report*, 24, 1998.

[90] G. Mateos and G. B. Giannakis. Robust PCA as bilinear decomposition with outlier-sparsity regularization. *IEEE Transactions on Signal Processing*, 60(10):5176–5190, 2012.

[91] R. Mazumder, J. H. Friedman, and T. Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495), 2011.

[92] D. Meng and F. D. L. Torre. Robust matrix factorization with unknown noise. In *International Conference on Computer Vision (ICCV) 2013*, pages 1337–1344. IEEE, 2013.

[93] R. Min, N. Kose, and J.-L. Dugelay. Kinectfacedb: A kinect database for face recognition. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 44(11):1534–1548, Nov 2014. ISSN 2168-2216. doi: 10.1109/TSMC.2014.2331215.

[94] K. Mohan and M. Fazel. Iterative reweighted algorithms for matrix rank minimization. *The Journal of Machine Learning Research*, 13(1):3441–3473, 2012.

[95] H. Mohimani, M. Babaie-Zadeh, and C. Jutten. A fast approach for overcomplete sparse decomposition based on smoothed ℓ_0 norm. *IEEE Transactions on Signal Processing*, 57(1): 289–301, 2009.

[96] J. F. Murray and K. Kreutz-Delgado. An improved FOCUSS-based learning algorithm for solving sparse linear inverse problems. In *Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers, 2001.*, volume 1, pages 347–351. IEEE, 2001.

[97] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[98] P. Netrapalli, U. Niranjan, S. Sanghavi, A. Anandkumar, and P. Jain. Non-convex robust PCA. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1107–1115, 2014.

[99] C.-W. Ngo, T.-C. Pong, and H.-J. Zhang. Motion analysis and segmentation through spatio-temporal slices processing. *IEEE Transactions on Image Processing*, 12(3):341–355, 2003.

[100] C. Nhan Duong, K. Luu, K. Gia Quach, and T. D. Bui. Beyond principal components: Deep boltzmann machines for face modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015*, pages 4786–4794, June 2015.

[101] T. Okatani, T. Yoshida, and K. Deguchi. Efficient algorithm for low-rank matrix factorization with missing components and performance comparison of latest algorithms. In *IEEE International Conference on Computer Vision (ICCV), 2011*, pages 842–849. IEEE, 2011.

[102] V. M. Patel, T. Wu, S. Biswas, P. J. Phillips, and R. Chellappa. Dictionary-based face recognition under variable lighting and pose. *IEEE Transactions on Information Forensics and Security*, 7(3):954–965, 2012.

[103] A. Pietsch. Approximation spaces. *Journal of Approximation Theory*, 32(2):115–134, 1981.

[104] S. Qaisar, R. M. Bilal, W. Iqbal, M. Naureen, and S. Lee. Compressive sensing: from theory to applications, a survey. *Journal of Communications and Networks*, 15(5):443–456, 2013.

[105] C. Qiu and N. Vaswani. Recursive sparse recovery in large but correlated noise. In *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 752–759. IEEE, 2011.

[106] C. Qiu, N. Vaswani, B. Lois, and L. Hogben. Recursive robust PCA or recursive sparse recovery in large but structured noise. *IEEE Transactions on Information Theory*, 60(8): 5007–5039, 2014.

[107] B. D. Rao. Analysis and extensions of the FOCUSS algorithm. In *Signals, Systems and Computers, 1996. Conference Record of the Thirtieth Asilomar Conference on*, pages 1218–1223. IEEE, 1996.

[108] B. D. Rao and K. Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Transactions on Signal Processing*, 47(1):187–200, 1999.

[109] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[110] P. Rodríguez and B. Wohlberg. Translational and rotational jitter invariant incremental principal component pursuit for video background modeling. In *IEEE International Conference on Image Processing (ICIP), 2015*, pages 537–541. IEEE, 2015.

[111] R. Saab and Ö. Yılmaz. Sparse recovery by non-convex optimization–instance optimality. *Applied and Computational Harmonic Analysis*, 29(1):30–48, 2010.

[112] R. Saab, R. Chartrand, and O. Yilmaz. Stable sparse approximations via nonconvex optimization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2008.*, pages 3885–3888. IEEE, 2008.

[113] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic. A semi-automatic methodology for facial landmark annotation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).*, pages 896–903. IEEE, 2013.

[114] R. Salakhutdinov and G. E. Hinton. Deep Boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455, 2009.

[115] F. Seidel, C. Hage, and M. Kleinsteuber. pROST: A Smoothed ℓ_p-norm Robust Online Subspace Tracking Method for Realtime Background Subtraction in Video. *Machine Vision and Applications*, 25(5):1227–1240, 2014.

[116] Q. Sun, S. Xiang, and J. Ye. Robust principal component analysis via capped norms. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 311–319. ACM, 2013.

[117] G. Tang and A. Nehorai. Robust principal component analysis based on low-rank and block-sparse matrix decomposition. In *Conference on Information Sciences and Systems (CISS) 2011*, pages 1–5. IEEE, 2011.

[118] Y. Tang. Gated boltzmann machine for recognition under occlusion. In *NIPS Workshop on Transfer Learning by Learning Rich Generative Models*, 2010.

[119] Y. Tang, R. Salakhutdinov, and G. Hinton. Robust boltzmann machines for recognition and denoising. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, pages 2264–2271. IEEE, 2012.

[120] M. E. Tipping. Sparse bayesian learning and the relevance vector machine. *The journal of machine learning research*, 1:211–244, 2001.

[121] Q. Tran-Dinh and Z. Zhang. Extended Gauss-Newton and Gauss-Newton-ADMM algorithms for low-rank matrix optimization. *arXiv preprint arXiv:1606.03358*, 2016.

[122] J. Tropp, A. C. Gilbert, et al. Signal recovery from partial information via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655– 4666, 2007.

[123] J. Trzasko and A. Manduca. Highly undersampled magnetic resonance image reconstruction via homotopic-minimization. *IEEE Transactions on Medical imaging*, 28(1):106–121, 2009.

[124] G. Tzimiropoulos and M. Pantic. Optimization problems for fast aam fitting in-the-wild. In *International Conference on Computer Vision (ICCV)*, pages 593–600. IEEE, 2013.

[125] G. Tzimiropoulos, J. Alabort-i Medina, S. P. Zafeiriou, and M. Pantic. Active orientation models for face alignment in-the-wild. *IEEE transactions on information forensics and security*, 9(12):2024–2034, 2014.

[126] H. Wang and A. Banerjee. Online alternating direction method (longer version). *preprint arXiv:1306.3721*, 2013.

[127] N. Wang, T. Yao, J. Wang, and D.-Y. Yeung. A probabilistic approach to robust matrix factorization. In *European Conference on Computer Vision 2012*, pages 126–139. Springer, 2012.

[128] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *The Journal of Machine Learning Research*, 3:1439–1461, 2003.

[129] Y. Wexler, E. Shechtman, and M. Irani. Space-time video completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2004*, volume 1, pages 120–128, 2004. URL http://www.wisdom.weizmann.ac.il/~vision/VideoCompletion.html.

[130] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (2):210–227, 2009.

[131] J. Xu, V. K. Ithapu, L. Mukherjee, J. M. Rehg, and V. Singh. GOSUS: grassmannian online subspace updates with structured-sparsity. In *International Conference on Computer Vision (ICCV)*, pages 3376–3383, 2013.

[132] A. Y. Yang, S. S. Sastry, A. Ganesh, and Y. Ma. Fast l_1-minimization algorithms and an application in robust face recognition: A review. In *IEEE International Conference on Image Processing (ICIP)*, pages 1849–1852. IEEE, 2010.

[133] L. Yang, T. K. Pong, and X. Chen. Alternating direction method of multipliers for nonconvex background/foreground extraction. *arXiv preprint arXiv:1506.07029*, 2015.

[134] M. Yang and L. Zhang. Gabor feature based sparse representation for face recognition with gabor occlusion dictionary. In *European Conference on Computer Vision 2010*, pages 448–461. Springer, 2010.

[135] X. Yi, D. Park, Y. Chen, and C. Caramanis. Fast Algorithms for Robust PCA via Gradient Descent. *arXiv preprint arXiv:1605.07784*, 2016.

[136] C.-H. Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics*, pages 894–942, 2010.

[137] L. Zhang, H. Shen, J. Z. Huang, et al. Robust regularized singular value decomposition with application to mortality data. *The Annals of Applied Statistics*, 7(3):1540–1561, 2013.

[138] T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *The Journal of Machine Learning Research*, 11:1081–1107, 2010.

[139] Y. Zhang, Z. Jiang, and L. S. Davis. Learning structured low-rank representations for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013*, pages 676–683. IEEE, 2013.

[140] Q. Zhao, D. Meng, Z. Xu, W. Zuo, and L. Zhang. Robust principal component analysis with complex noise. In *International Conference on Machine Learning (ICML)*, pages 55–63. IEEE, 2014.

[141] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, and M. Okutomi. Practical low-rank matrix approximation under robust ℓ_1-norm. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012*, pages 1410–1417. IEEE, 2012.

[142] Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma. Stable principal component pursuit. In *IEEE International Symposium on Information Theory Proceedings (ISIT) 2010*, pages 1518–1522. IEEE, 2010.

Appendix A

Convergence Analysis

In this section, we will first show that our cost function is monotonically decreasing and that the generated sub-sequences eventually reach an accumulation point. Finally, any accumulation point of the sequence is a stationary point of the problem (107). We can express the problem (107) as follows.

$$\min_{\mathbf{L},\,\mathbf{S},\,\mathbf{L}+\mathbf{S}=\mathbf{M}} \mathbf{F}(\mathbf{L},\mathbf{S}) \tag{154}$$

where $\mathbf{F}(\mathbf{L},\mathbf{S})$ is defined as

$$\mathbf{F}(\mathbf{L},\mathbf{S}) = \sum_{j=1}^{d} g(\sigma_j) + \lambda \sum_{ij=1}^{m\times n} g(|s_{ij}|) + \frac{\mu}{2}\|\mathbf{M}-\mathbf{L}-\mathbf{S}\|_F^2 \tag{155}$$

The Lagrangian function of Eqn. (154) is the same as defined in Eqn. (108). We denote $f(\mathbf{L},\mathbf{S}) = \frac{\mu}{2}\|\mathbf{M}-\mathbf{L}-\mathbf{S}\|_F^2$. The loss function is a smooth, convex function.

Proposition 1 Given $\mathbf{X},\mathbf{Z} \in \mathbb{R}^{m\times n}$ For any $\mathbf{X}',\mathbf{Z}' \in \mathbb{R}^{m\times n}$ it holds $f(\mathbf{X}',\mathbf{Z}') \geq f(\mathbf{X},\mathbf{Z}) + \langle \nabla_{\mathbf{X}} f(\mathbf{X},\mathbf{Z}), \mathbf{X}'-\mathbf{X}\rangle + \langle \nabla_{\mathbf{Z}} f(\mathbf{X},\mathbf{Z}), \mathbf{Z}'-\mathbf{Z}\rangle$

Proposition 2 For any $\mathbf{L}^{k+1}$ and $\mathbf{L}^{k}$ generated by **Algorithm 1**, it holds $\mathbf{U}^{k+1}{\mathbf{V}^{k+1}}^{\top} \leq \mathbf{U}^{k}{\mathbf{V}^{k}}^{\top}$

Theorem 1 Let $\left\{\mathbf{L}^k,\mathbf{S}^k\right\}$ be the sequence generated in **Algorithm 5**. Then $\mathbf{F}(\mathbf{L}^k,\mathbf{S}^k)$ is monotonically decreasing i.e. $\mathbf{F}(\mathbf{L}^k,\mathbf{S}^k) - \mathbf{F}(\mathbf{L}^{k+1},\mathbf{S}^{k+1}) \geq 0$; the sequence $\left\{\mathbf{L}^k,\mathbf{S}^k\right\}$ is bounded and has at least one accumulation point.

Proof: Following from the fact that $\{\mathbf{L}^{k+1},\mathbf{S}^{k+1}\}$ is the local optimal solution to (109a) and (109b), respectively. We know that the Karush-Kuhn-Tucker (KKT) condition is satisfied, i.e. $\mathbf{L}^{k+1}$

minimizes $\mathcal{L}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Y}^k, \mu^k)$, similarly for $\mathbf{S}^{k+1}$, we have:

$$0 \in \nabla_{\mathbf{S}} \mathcal{L}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Y}^k, \mu^k) \tag{156}$$

$$0 \in \nabla_{\mathbf{L}} \mathcal{L}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Y}^k, \mu^k) \tag{157}$$

Taking partial derivative of the Lagrangian function in (108), we have $\nabla_{\mathbf{L}} \mathcal{L}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Y}^k, \mu^k) = \frac{\partial \mathcal{L}^{k+1}}{\partial \mathbf{L}}$ and $\nabla_{\mathbf{S}} \mathcal{L}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}, \mathbf{Y}^k, \mu^k) = \frac{\partial \mathcal{L}^{k+1}}{\partial \mathbf{S}}$

$$\frac{\partial \mathcal{L}^{k+1}}{\partial \mathbf{L}_{ij}} = \sum_l v_l^k \mathbf{u}_{il}^{k+1} \mathbf{v}_{jl}^{k+1} + \frac{\partial_{\mathbf{L}} f(\mathbf{L}^{k+1}, \mathbf{S}^{k+1})}{\partial \mathbf{L}_{ij}} - \mathbf{Y}_{ij}^k \tag{158}$$

where $\mathbf{L}^{k+1} = \mathbf{U}^{k+1} \mathbf{\Sigma} \mathbf{V}^{k+1^\top}$ is SVD of the matrix $\mathbf{L}^{k+1}$.

$$\frac{\partial \mathcal{L}^{k+1}}{\partial \mathbf{S}_{ij}} = \lambda w_{ij}^k \partial |s_{ij}^{k+1}| + \frac{\partial_{\mathbf{S}} f(\mathbf{L}^{k+1}, \mathbf{S}^{k+1})}{\partial \mathbf{S}_{ij}} - \mathbf{Y}_{ij}^k \tag{159}$$

This means that

$$\frac{\partial_{\mathbf{L}} f(\mathbf{L}^{k+1}, \mathbf{S}^{k+1})}{\partial \mathbf{L}_{ij}} = \mathbf{Y}_{ij}^k - \sum_l v_l^k \mathbf{u}_{il}^{k+1} \mathbf{v}_{jl}^{k+1} = \mathbf{J}_{\mathbf{L}_{ij}}^{k+1} \tag{160a}$$

$$\frac{\partial_{\mathbf{S}} f(\mathbf{L}^{k+1}, \mathbf{S}^{k+1})}{\partial \mathbf{S}_{ij}} = \mathbf{Y}_{ij}^k - \lambda w_{ij}^k c_{ij}^{k+1} = \mathbf{J}_{\mathbf{S}_{ij}}^{k+1} \tag{160b}$$

where c_{ij}^{k+1} denotes the sign of (s_{ij}^{k+1}). $\mathbf{J}_{\mathbf{L}}$ and $\mathbf{J}_{\mathbf{S}}$ are the gradient matrices.

From the above, we can form the objective function difference as follows:

$$\begin{aligned}
&\mathbf{F}(\mathbf{L}^k, \mathbf{S}^k) - \mathbf{F}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}) \\
&= \sum_j^d \left(g(\sigma_j^k) - g(\sigma_j^{k+1}) \right) + \lambda \sum_{ij}^{m \times n} \left(g(|s_{ij}^k|) - g(|s_{ij}^{k+1}|) \right) + f(\mathbf{L}^k, \mathbf{S}^k) - f(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}) \\
&\geq \sum_j^d v_j^k \left(\sigma_j^k - \sigma_j^{k+1} \right) + \lambda \sum_{ij}^{m \times n} w_{ij}^k \left(|s_{ij}^k| - |s_{ij}^{k+1}| \right) + \left\langle \mathbf{J}_{\mathbf{L}_{ij}}^{k+1}, \mathbf{L}^k - \mathbf{L}^{k+1} \right\rangle + \left\langle \mathbf{J}_{\mathbf{S}_{ij}}^{k+1}, \mathbf{S}^k - \mathbf{S}^{k+1} \right\rangle \\
&= \lambda \sum_{ij}^{m \times n} \left(w_{ij}^k \left(\left(|s_{ij}^k| - c_{ij}^{k+1} s_{ij}^k \right) - \left(|s_{ij}^{k+1}| - c_{ij}^{k+1} s_{ij}^{k+1} \right) \right) \right) + \left\langle \mathbf{Y}^k, \mathbf{L}^k + \mathbf{S}^k \right\rangle - \left\langle \mathbf{Y}^k, \mathbf{L}^{k+1} + \mathbf{S}^{k+1} \right\rangle \\
&+ \sum_{ij}^{m \times n} \sum_l^d v_l^k \sigma_l^{k+1} \left(\sum_l^d \mathbf{u}_{il}^{k+1} \mathbf{v}_{jl}^{k+1} \mathbf{v}_{jl}^{k+1} \mathbf{u}_{il}^{k+1} - 1 \right) + \sum_{ij}^{m \times n} \sum_l^d v_l^k \sigma_l^k \left(1 - \sum_l^d \mathbf{u}_{il}^{k+1} \mathbf{v}_{jl}^{k+1} \mathbf{v}_{jl}^k \mathbf{u}_{il}^k \right) \geq 0
\end{aligned} \tag{161}$$

The last inequality follows from the facts that

- $(\mathbf{L}^k + \mathbf{S}^k = \mathbf{M})$ and $(\mathbf{L}^{k+1} + \mathbf{S}^{k+1} = \mathbf{M})$
  $\Rightarrow \left\langle \mathbf{Y}^k, \mathbf{L}^k + \mathbf{S}^k \right\rangle - \left\langle \mathbf{Y}^k, \mathbf{L}^{k+1} + \mathbf{S}^{k+1} \right\rangle = 0$

- $\sum_{ij}^{m\times n} w_{ij}^k(|s_{ij}^k| - c^{k+1}s_{ij}^k) \geq 0$
- $\sum_{ij}^{m\times n} w_{ij}^k(|s_{ij}^{k+1}| - c^{k+1}s_{ij}^{k+1}) = 0$
- $\sum_{ij}^{m\times n}\sum_{l}^{d} v_l^k\sigma_l^{k+1}\left(\sum_l^d \mathbf{u}_{il}^{k+1}\mathbf{v}_{jl}^{k+1}\mathbf{v}_{jl}^{k+1}\mathbf{u}_{il}^{k+1} - 1\right) = 0$
- $\sum_{ij}^{m\times n}\sum_{l}^{d} v_l^k\sigma_l^{k}\left(1 - \sum_l^d \mathbf{u}_{il}^{k+1}\mathbf{v}_{jl}^{k+1}\mathbf{v}_{jl}^{k}\mathbf{u}_{il}^{k}\right) \geq 0$ (Follows from Proposition 2)

We conclude that $\mathbf{F}(\mathbf{L}^k, \mathbf{S}^k) - \mathbf{F}(\mathbf{L}^{k+1}, \mathbf{S}^{k+1}) \geq 0$. This shows that the sequence $\mathbf{F}(\mathbf{L}^k, \mathbf{S}^k)$ is monotonically decreasing. Then we have

$$\begin{aligned}
\|\mathbf{L}^k\|_{Sp} &= \sum_j g(\sigma_j^k) \leq \sum_j g(\sigma_j^k) + \lambda\sum_{ij} g(|s_{ij}^k|) = \mathbf{F}(\mathbf{L}^k, \mathbf{S}^k) \leq \mathbf{F}(\mathbf{L}^1, \mathbf{S}^1) \triangleq \mathbf{D} \\
\|\mathbf{S}^k\|_{p} &= \sum_{ij} g(|s_{ij}^k|) \leq \sum_j g(\sigma_j^k) + \lambda\sum_{ij} g(|s_{ij}^k|) = \mathbf{F}(\mathbf{L}^k, \mathbf{S}^k) \leq \mathbf{F}(\mathbf{L}^1, \mathbf{S}^1) \triangleq \mathbf{D}
\end{aligned} \tag{162}$$

Thus, the sequence $\left\{\mathbf{L}^k, \mathbf{S}^k\right\}$ is bounded. Furthermore, $\mathbf{F}(\mathbf{L}^k, \mathbf{S}^k)$ is monotonically decreasing (Theorem 1) and $\mathbf{F}(\mathbf{L}^k, \mathbf{S}^k) \geq 0$. As a result, by applying the theorem of Bolzano-Weierstrass, we can conclude the existence of an accumulation point. ■

Theorem 2 Let $\mathbf{G} = (\mathbf{L}, \mathbf{S}, \mathbf{Y})$ and $\{\mathbf{G}^k\}_{k=1}^{\infty}$ be generated by LP-RPCA. Assume that $\lim_{k\to\infty}\{\mathbf{G}^{k+1} - \mathbf{G}^k\} = 0$. Then, any accumulation point of $\{\mathbf{G}^k\}_{k=1}^{\infty}$ is a stationary point.

Proof: Theorem 1 shows that the sequence $\left\{\mathbf{L}^k, \mathbf{S}^k\right\}$ is bounded. Thus, there exists an accumulation point $\left\{\hat{\mathbf{L}}, \hat{\mathbf{S}}\right\}$ and a subsequence $\left\{\mathbf{L}^{k_j}, \mathbf{S}^{k_j}\right\}$, where $\lim_{j\to\infty}\mathbf{L}^{k_j} \to \hat{\mathbf{L}}$ and $\lim_{j\to\infty}\mathbf{S}^{k_j} \to \hat{\mathbf{S}}$. From Eqns. (160a) and (160b), we have

$$\begin{aligned}
\frac{\partial_{\mathbf{L}} f(\mathbf{L}^{k_j+1}, \mathbf{S}^{k_j+1})}{\partial \mathbf{L}_{ij}} - \mathbf{Y}_{ij}^{k_j} + \sum_l v_l^{k_j}\mathbf{u}_{il}^{k_j+1}\mathbf{v}_{jl}^{k_j+1} = 0 \\
\frac{\partial_{\mathbf{S}} f(\mathbf{L}^{k_j+1}, \mathbf{S}^{k_j+1})}{\partial \mathbf{S}_{ij}} - \mathbf{Y}_{ij}^{k_j} + \lambda w_{ij}^{k_j} c_{ij}^{k_j+1} = 0
\end{aligned} \tag{163}$$

From the above, we can conclude that $\left\{\mathbf{L}^{k_j}, \mathbf{S}^{k_j}\right\}$ also converges to any $\left\{\tilde{\mathbf{L}}, \tilde{\mathbf{S}}\right\}$ when $j \to \infty$. From the fact that $\lim_{k\to\infty}\|\mathbf{L}^k - \mathbf{L}^{k+1}\|_F = 0$ and $\lim_{k\to\infty}\|\mathbf{S}^k - \mathbf{S}^{k+1}\|_F = 0$, we can conclude that $\|\hat{\mathbf{L}} - \tilde{\mathbf{L}}\|_F = \lim_{k\to\infty}\|\mathbf{L}^{k_j} - \mathbf{L}^{k_j+1}\|_F = 0$ and $\|\hat{\mathbf{S}} - \tilde{\mathbf{S}}\|_F = \lim_{k\to\infty}\|\mathbf{S}^{k_j} - \mathbf{S}^{k_j+1}\|_F = 0$. This means

that $\hat{\mathbf{L}} = \tilde{\mathbf{L}}$ and $\hat{\mathbf{S}} = \tilde{\mathbf{S}}$. With $j \to \infty$, we can rewritten Eqn. (163) as

$$\begin{aligned} \frac{\partial_{\mathbf{L}} f(\hat{\mathbf{L}}, \hat{\mathbf{S}})}{\partial \mathbf{L}_{ij}} - \hat{\mathbf{Y}}_{ij} + \sum_{l} \hat{v}_l \hat{\mathbf{u}}_{il} \hat{\mathbf{v}}_{jl} = 0 \\ \frac{\partial_{\mathbf{S}} f(\hat{\mathbf{L}}, \hat{\mathbf{S}})}{\partial \mathbf{S}_{ij}} - \hat{\mathbf{Y}}_{ij} + \lambda \hat{w}_{ij} \hat{c}_{ij} = 0 \end{aligned} \tag{164}$$

As a result, $\left\{\hat{\mathbf{L}}, \hat{\mathbf{S}}\right\}$ is a stationary point that satisfies the the Karush-Kuhn-Tucker (KKT) conditions of (107). ■

Remark Although it is difficult to guarantee the algorithm convergence to a global minimum, experiments and examples suggest that the proposed method has a strong convergence behavior (See Fig. 5.3 - LEFT). We provide a simple proof of convergence of LP-RPCA to show that any accumulation point of the iteration sequence generated by the algorithm is a stationary point that satisfies the KKT conditions. This result provides an insight about the behavior of the proposed algorithm.