Imperial College London
Department of Computing

Side Information in Robust Principal Component Analysis: Algorithms and Applications

Niannan Xue

August, 2019

Supervised by Dr. Stefanos Zafeiriou

Submitted in part fulfilment of the requirements for the degree of PhD in Computing and the Diploma of Imperial College London. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Abstract

Dimensionality reduction and noise removal are fundamental machine learning tasks that are vital to artificial intelligence applications. Principal component analysis has long been utilised in computer vision to achieve the above mentioned goals. Recently, it has been enhanced in terms of robustness to outliers in robust principal component analysis. Both convex and non-convex programs have been developed to solve this new formulation, some with exact convergence guarantees. Its effectiveness can be witnessed in image and video applications ranging from image denoising and alignment to background separation and face recognition. However, robust principal component analysis is by no means perfect. This dissertation identifies its limitations, explores various promising options for improvement and validates the proposed algorithms on both synthetic and real-world datasets.

Common algorithms approximate the NP-hard formulation of robust principal component analysis with convex envelopes. Though under certain assumptions exact recovery can be guaranteed, the relaxation margin is too big to be squandered. In this work, we propose to apply gradient descent on the Burer-Monteiro bilinear matrix factorisation to squeeze this margin given available subspaces. This non-convex approach improves upon conventional convex approaches both in terms of accuracy and speed. On the other hand, oftentimes there is accompanying side information when an observation is made. The ability to assimilate such auxiliary sources of data can ameliorate the recovery process. In this work, we investigate in-depth such possibilities for incorporating side information in restoring the true underlining low-rank component from gross sparse noise. Lastly, tensors, also known as multi-dimensional arrays, represent real-world data more naturally than matrices. It is thus advantageous to adapt robust principal component analysis to tensors. Since there is no exact equivalence between tensor rank and matrix rank, we employ the notions of Tucker rank and CP rank as our optimisation objectives. Overall, this dissertation carefully defines the problems when facing real-world computer vision challenges, extensively and impartially evaluates the state-of-the-art approaches, proposes novel solutions and provides sufficient validations on both simulated data and popular real-world datasets for various mainstream computer vision tasks.

Dedicated to my loving parents

Acknowledgements

First and foremost, I would like to express my greatest gratitude to my supervisor Dr Stefanos Zafeiriou for his excellent guidance, valuable suggestions and endless support through my entire PhD life. He gave me light when I was in my darkest time at the beginning of my PhD at Imperial. To this day, I still feel extremely lucky to be able to work under his guidance. He is such a genuine person for whom my words will not be enough to express.

I would also like to thank Dr Ioannis Panagakis, who introduced me to the fields of linear algebra, optimisation, etc. which are fundamental and pivotal to my research in machine learning. I am very much in debt to him for investing his precious time going through my code and manuscripts.

I also enjoy working in the lovely environment of iBUG. My colleagues are always patient and helpful. Without their encouragement and help, I will still be troubled by IT headaches. Especially, I want to thank my collaborators, Dr Shiyang Cheng and Mr Jiankang Deng for their thoughtful discussions and work on several recent projects.

I want give my special thanks to Dr El-Kholy Amani, whom I have been troubling since the beginning of my PhD. She is always nice and helpful and provides me with emotional support even when it is not working hours. I am also grateful to Ms Teresa Ng her countless support.

My mum and dad have been great and I would like to thank them for always being on my side and support whatever decision I make. Their endless love and kindness almost spoil me. I will always love them and wish them happiness every day.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

1.1 Background

Principal component analysis (PCA) is a very popular machine learning method for dimensionality reduction. It is also known as the discrete Karhunen–Loève transform (KLT), proper orthogonal decomposition (POD), singular value decomposition (SVD) and eigenvalue decomposition (EVD). The ability of PCA to transform orthogonally a large set of observations from correlated variables into a small set of principal components from uncorrelated variables has seen applications in subgroup discovery from single-cell RNA-Seq data [1], structural characterization of the full-length HCV IRES in solution [90], dementia prediction using miRNA expression data [113], identification of novel loci for body shape [99] and so forth. Developed in 1901 by Karl Pearson, formally PCA linearly transforms data into a new orthogonal coordinate system such that it will have the largest variance on the first coordinate, second largest variance on the second coordinate, etc. However, the discriminative power of PCA decreases as the signal/noise (S/N) ratio of the data decreases. Specifically, the underlining structure can only be exactly reconstructed by minimizing squared error if there is small Gaussian noise. In today's world, data comes from diverse sources so the associated noise is not necessarily normally distributed. As a result, generalisations of PCA are demanded to clean contaminated data.

Robust principal component analysis (RPCA) is one modification of PCA aimed at recovering grossly corrupted observations from outliers in a unified approach, such that no pre-processing steps are needed to reduce the scale of noise first. For a highly corrupted measurement, RPCA finds a low-rank component as well as the sparse noise simultaneously. The most common algorithm, principal component pursuit (PCP), adopts a convex relaxation approach, which replaces the rank constraint with a nuclear norm and the sparsity constraint with a l_1 norm. The resulting program solves RPCA using the alternating direction method of multipliers. Under certain incoherence conditions, PCP exactly recovers the desired components of RPCA with convergence guarantees.

Nevertheless, RPCA is still sometimes limited in its effect in addressing real-world applications, e.g. background subtraction, face recognition, etc. Moreover, RPCA takes no account of extra information that might be helpful towards the end goal. In light of these challenges, this research carefully defines the problems, fully explores the potential solutions and deeply analyses the results. The main contributions of this dissertation are:

- Features have been incorporated in robust principal component analysis in a convex way. We revisit this problem by proposing a non-convex algorithm to decompose the observation matrix into a low-rank component and the corresponding sparse noise in a novel and elegant manner. The proposed approach carries low computational complexity and has exact recovery guarantee from rigorous theoretical analysis. Simulation data proves the validity of the proposed algorithm and shows its superiority over traditional convex methods with regard to recoverability and speed. In other words, our non-convex method is more accurate and faster than the best-known solver for this problem. We apply the proposed algorithm to image classification and face denoising as examples of real-world applications to showcase its effectiveness.

  N. Xue, J. Deng, Y. Panagakis, S. Zafeiriou. Informed Non-Convex Robust Principal Component Analysis with Features. *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

- It is known that robust principal component analysis is a good tool to separate a low-rank subspace from gross noise. As such, it has become a cornerstone in many machine learning and computer vision applications for dealing with high-dimensional data. Despite the success of RPCA in solving many rank minimisation problems, sometimes it unavoidably produces degenerate and sub-optimal solutions. It is argued that domain-dependent prior side information could be the cure to some extent. In this research, we explore two candidate models to address the problem of RPCA with side information knowledge for better low-rank recovery. For greater assurance, four computer vision tasks of background subtraction, facial image denoising, face and facial expression recognition have been used to test the proposed algorithms. A total of five real-world datasets together with simulation experiments demonstrate that these algorithms outperform previous state-of-the-art approaches in terms of both robustness and effectiveness.

  N. Xue, Y. Panagakis, S. Zafeiriou. Side Information in Robust Principal Component Analysis: Algorithms and Applications. *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

- Domain-specific side information or prior knowledge has been capitalised upon to obtain a better low-rank feature representation from robust principal component analysis, which might be otherwise unable to generate satisfactory outcomes for visual data with error corruption of certain types. In this research, we extend the two models for RPCA with side information to the case of missing values. In particular, we focus on the UV completion task which is crucial in pose-invariant face recognition. We are the first to use a generative adversarial network to construct side information and extract subspaces. We show that our framework have both recovery and speed advantage over existing solutions. The assessment is conducted through both simulation and large-scale real-world datasets.

  N. Xue, J. Deng, S. Cheng, Y. Panagakis, S. Zafeiriou. Side Information for Face Completion: a Robust PCA Approach. *IEEE Transactionson Pattern Analysis and Machine Intelligence (T-PAMI)*, 2019.

- Tensors naturally arise in modern signal processing applications. Therefore it is key to have a framework that can separate a low-rank subspace reliably from multi-dimensional signals. Existing methods are under-performing either because the simplification is too radical or data has been drastically transformed. With such motivation, we propose two new robust low-rank tensor models: Tensor Orthonormal Robust PCA (TORPCA) and Tensor Robust CP Decomposition (TRCPD). They rely on Tucker and CP decomposition of a tensor respectively and a l_p norm is utilised to regularise for noise. The proposed methods are benchmarked against other recent top-performing low-rank models on both simulated and real-world data. Our methods achieve the state-of-the-art performance in terms of speed and accuracy compared to others in these experiments.

  N. Xue, G. Papamakarios, M. Bahri, Y. Panagakis, S. Zafeiriou. Robust Low-rank Tensor Modelling Using Tucker and CP Decomposition. *The 25th European Signal Processing Conference (EUSIPCO)*, 2017.

This thesis is organised as follows. Basic concepts are reviewed first in Chapter 1. Then in Chapter 2, a non-convex optimisation approach to RPCA with features is introduced such that both the computational complexity and recovery are greatly improved. In Chapter 3, an extension to RPCA is made to assimilate useful side information that can help with the data recovery process, which is followed by a chapter to discuss the generative adversarial network as a source of side information for the task of face completion. Afterwards, two models of tensor-based robust principal component analysis are examined in depth in Chapter 5. Finally, Chapter 6 concludes the dissertation with a discussion of the contributions, open questions and future research directions.

1.2 Notations

We introduce notations used throughout the thesis unless otherwise stated. Lowercase Latin and Greek letters denote scalars, e.g. r, γ. Bold lowercase Latin letters denote vectors, e.g. $\boldsymbol{a}$. Bold uppercase Latin letters denote matrices, e.g. $\boldsymbol{A}$. Bold uppercase calligraphic Latin letters denote tensors, e.g. $\boldsymbol{\mathcal{L}}$. Bold uppercase Greek letters denote operators on tensors and matrices, e.g. $\boldsymbol{\Theta}(\boldsymbol{\mathcal{S}})$, $\boldsymbol{\Phi}(\boldsymbol{U})$. $\boldsymbol{A}_{i\cdot}$ and $\boldsymbol{A}_{\cdot j}$ represent the i^{th} row and the j^{th} column of $\boldsymbol{A}$. Projection onto support set Ω is given by $\boldsymbol{\Pi}_\Omega$. $\langle \boldsymbol{A}, \boldsymbol{B} \rangle$ represents $\text{tr}(\boldsymbol{A}^\intercal \boldsymbol{B})$. $|\boldsymbol{A}|$ is the element-wise absolute value of matrix $\boldsymbol{A}$; $\|\boldsymbol{A}\|_F$ is the Frobenius norm; $\|\boldsymbol{A}\|_*$ is the nuclear norm; $\|\boldsymbol{A}\|_1$ is the sum of absolute values of all matrix entries; $\|\boldsymbol{A}\|_\infty$ is the maximum absolute value among all matrix entries; $\|\boldsymbol{A}\|_2$ is the largest singular value; otherwise, $\|\boldsymbol{A}\|_p$ is the l_p-norm of vectorized $\boldsymbol{A}$; and $\|\boldsymbol{A}\|_{2,\infty}$ is the maximum of matrix row l_2-norms. σ_i is the ith largest singular value of a matrix and $\sigma_{j\%}$ is the singular value at the jth percentile. $\boldsymbol{A} \circ \boldsymbol{B}$ symbolises element-wise multiplication of two matrices of the same dimension. Additionally, $\boldsymbol{A} \odot \boldsymbol{B}$ denotes the Khatri-Rao product between matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ and $\boldsymbol{\mathcal{X}} \times_i \boldsymbol{U}$ is the i-mode product [60].

1.3 Norms

A vector norm $\|\cdot\|$ is a function from $\mathbb{R}^n$ to $\mathbb{R}$ such that:

- $\|\boldsymbol{x}\| \geq \boldsymbol{0}$ for all $\boldsymbol{x} \in \mathbb{R}^n$, and $\|\boldsymbol{x}\| = 0$ if and only if $\boldsymbol{x} = \boldsymbol{0}$;
- $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$;
- $\|\alpha \boldsymbol{x}\| = |\alpha| \|\boldsymbol{x}\|$ for all $\alpha \in \mathbb{R}$, $\boldsymbol{x} \in \mathbb{R}^n$.

We can deduce the reverse triangle inequality $|\|\boldsymbol{x}\| - \|\boldsymbol{y}\|| \leq \|\boldsymbol{x} - \boldsymbol{y}\|$.

For $\boldsymbol{x} = (x_1 \ldots x_n) \in \mathbb{R}^n$, the vector norm

$$\|\boldsymbol{x}\|_p = (\sum_{j=1}^{n} |x_j|^p)^{\frac{1}{p}}, p \geq 1,$$

is called the l_p norm. Special cases include:

- l_1 norm: $\|\boldsymbol{x}\|_1 = \sum_{j=1}^n |x_j|$;
- l_2 (or Euclidean) norm: $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{j=1}^n |x_j|^2} = \sqrt{\boldsymbol{x}^\intercal \boldsymbol{x}}$;
- Infinity (or maximum) norm: $\|\boldsymbol{x}\|_\infty = \max_{1 \leq j \leq n} |x_j|$.

For $\boldsymbol{x}$, $\boldsymbol{y} \in \mathbb{R}^n$, the following relationships hold:

- Holder inequality: $|\boldsymbol{x}^\intercal \boldsymbol{y}| \leq \|\boldsymbol{x}\|_1 \|\boldsymbol{y}\|_\infty$;
- Cauchy-Schwarz inequality: $|\boldsymbol{x}^\intercal \boldsymbol{y}| \leq \|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2$, with equality if and only if $\boldsymbol{x}$ and $\boldsymbol{y}$ are multiples of each other.

Together, for $\boldsymbol{x} \in \mathbb{R}^n$, they imply

$$|\sum_{i=1}^{n} x_i| \leq n \max_{1 \leq i \leq n} |x_i|, \quad |\sum_{i=1}^{n} x_i| \leq \sqrt{n} \|\boldsymbol{x}\|_2.$$

Furthermore, for $\boldsymbol{x} \in \mathbb{R}^n$, we have the following inequalities and equalities:

- $\|\boldsymbol{x}\|_2 \leq \sqrt{\|\boldsymbol{x}\|_1 \|\boldsymbol{x}\|_\infty}$;
- $\|\boldsymbol{P}\boldsymbol{x}\|_p = \|\boldsymbol{x}\|_p$, where $\boldsymbol{P}$ is a permutation matrix;
- $\|\boldsymbol{R}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$, where $\boldsymbol{R}^\intercal \boldsymbol{R} = \boldsymbol{I}$;
- $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{n} \|\boldsymbol{x}\|_2$;
- $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_2 \leq \sqrt{n} \|\boldsymbol{x}\|_\infty$;
- $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_1 \leq n \|\boldsymbol{x}\|_\infty$.

A matrix norm $|| \cdot ||$ is a function from $\mathbb{R}^{m \times n}$ to $\mathbb{R}$ such that:

- $||\boldsymbol{A}|| \geq 0$ for all $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, and $||\boldsymbol{A}|| = 0$ if and only if $\boldsymbol{A} = \boldsymbol{0}$;
- $||\boldsymbol{A} + \boldsymbol{B}|| \leq ||\boldsymbol{A}|| + ||\boldsymbol{B}||$ for all $\boldsymbol{A}$, $\boldsymbol{B} \in \mathbb{R}^{m \times n}$;
- $||\alpha \boldsymbol{A}|| = |\alpha| ||\boldsymbol{A}||$ for all $\alpha \in \mathbb{R}$, $\boldsymbol{A} \in \mathbb{R}^{m \times n}$.

We can deduce that $|\|\boldsymbol{A} + \boldsymbol{B}\| - \|\boldsymbol{A}\|| \leq \|\boldsymbol{B}\|$.

For $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, the matrix norm

$$||\boldsymbol{A}||_p = \max_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{||\boldsymbol{A}\boldsymbol{x}||_p}{||\boldsymbol{x}||_p} = \max_{||\boldsymbol{x}||_p = 1} ||\boldsymbol{A}\boldsymbol{x}||_p$$

is called the matrix l_p norm. For $\boldsymbol{y} \in \mathbb{R}^n$, we have $||\boldsymbol{A}\boldsymbol{y}||_p \leq ||\boldsymbol{A}||_p||\boldsymbol{y}||_p$ by definition. For $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times p}$, we have the submultiplicative inequality:

$$\|\boldsymbol{A}\boldsymbol{B}\|_p \leq \|\boldsymbol{A}\|_p\|\boldsymbol{B}\|_p.$$

For $\boldsymbol{B}$, a submatrix of $\boldsymbol{A}$, $\|\boldsymbol{B}\|_p \leq \|\boldsymbol{A}\|_p$.

For nonsingular matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\|\boldsymbol{A}\|_p\|\boldsymbol{A}^{-1}\|_p \geq 1$. If $\boldsymbol{P}$ is a permutation matrix, then $\|\boldsymbol{P}\|_p = 1$. For permutation matrices $\boldsymbol{P} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$, $\|\boldsymbol{P}\boldsymbol{A}\boldsymbol{Q}\|_p = \|\boldsymbol{A}\|_p$. If $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ is a diagonal matrix with diagonal elements d_{jj}, then $\|\boldsymbol{D}\|_p = \max_{1\leq j\leq n} |d_{jj}|$.

The matrix l_1 norm is equal to the maximal absolute column sum:

$$\|\boldsymbol{A}\|_1 = \max_{1\leq j\leq n} \|\boldsymbol{A}\boldsymbol{e}_j\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{m} |a_{ij}|.$$

The matrix infinity norm is equal to the maximal absolute row sum:

$$\|\boldsymbol{A}\|_\infty = \max_{1\leq j\leq m} \|\boldsymbol{A}^\intercal\boldsymbol{e}_i\|_1 = \max_{1\leq i\leq m} \sum_{j=1}^{n} |a_{ij}|.$$

It can be shown that $\|\boldsymbol{A}^\intercal\|_2 = \|\boldsymbol{A}\|_2$ and $\|\boldsymbol{A}^\intercal\boldsymbol{A}\|_2 = \|\boldsymbol{A}\|_2^2$. And for orthogonal matrices $\boldsymbol{U} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times n}$, $\|\boldsymbol{U}\|_2 = \|\boldsymbol{V}\|_2 = 1$ and $\|\boldsymbol{U}\boldsymbol{A}\boldsymbol{V}\|_2 = \|\boldsymbol{A}\|_2$. If $\boldsymbol{A} \neq \boldsymbol{0}$ is idempotent, then $\|\boldsymbol{A}\|_p \geq 1$ and $\|\boldsymbol{A}\|_2 = 1$.

$\boldsymbol{A} \in \mathbb{R}^{m\times n}$ admits the following inequalities:

- $\|\boldsymbol{A}\|_2 \leq \sqrt{\|\boldsymbol{A}\|_1\|\boldsymbol{A}\|_\infty}$;
- $\frac{1}{\sqrt{n}}\|\boldsymbol{A}\|_\infty \leq \|\boldsymbol{A}\|_2 \leq \sqrt{m}\|\boldsymbol{A}\|_\infty$;
- $\frac{1}{\sqrt{m}}\|\boldsymbol{A}\|_1 \leq \|\boldsymbol{A}\|_2 \leq \sqrt{n}\|\boldsymbol{A}\|_1$.

For vectors $\boldsymbol{x} \in \mathbb{R}^m$ and $\boldsymbol{y} \in \mathbb{R}^n$, the norms of outer products satisfy:

$$\|\boldsymbol{x}\boldsymbol{y}^\intercal\|_2 = \|\boldsymbol{x}\|_2\|\boldsymbol{y}\|_2, \quad \|\boldsymbol{x}\boldsymbol{y}^\intercal\|_\infty = \|\boldsymbol{x}\|_\infty\|\boldsymbol{y}\|_1.$$

If $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ has singular values $\sigma_1 \geq \cdots \geq \sigma_p$, where $p = \min\{m, n\}$, then

$$\|\boldsymbol{A}\|_2 = \max_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \sigma_1, \quad \min_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2} = \sigma_p.$$

And if $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is nonsingular with singular values $\sigma_1 \geq \cdots \sigma_n > 0$, then $\|\boldsymbol{A}^{-1}\|_2 = \frac{1}{\sigma_n}$.

Let $\boldsymbol{A}$,$\boldsymbol{A} + \boldsymbol{E} \in \mathbb{R}^{m \times n}$, $p = \min\{m, n\}$, and let $\sigma_1 \geq \cdots \geq \sigma_p$ be the singular values of $\boldsymbol{A}$ and $\tilde{\sigma}_1 \geq \cdots \geq \tilde{\sigma}_p$ the singular values of $\boldsymbol{A} + \boldsymbol{E}$. Then

$$|\tilde{\sigma}_1 - \sigma_1| \leq \|\boldsymbol{E}\|_2, \quad |\tilde{\sigma}_p - \sigma_p| \leq \|\boldsymbol{E}\|_2.$$

Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ have singular values σ_j, left singular vectors $\boldsymbol{u}_j$ and right singular vectors $\boldsymbol{v}_j$. If $k <$ rank($\boldsymbol{A}$), then the absolute distance of $\boldsymbol{A}$ to the set of rank k matrices is

$$\sigma_{k+1} = \min_{\boldsymbol{B} \in \mathbb{R}^{m \times n}, \text{rank}(\boldsymbol{B}) = k} \|\boldsymbol{A} - \boldsymbol{B}\|_2 = \|\boldsymbol{A} - \boldsymbol{A}_k\|_2,$$

where $\boldsymbol{A}_k = \sum_{j=1}^{k} \sigma_j \boldsymbol{u}_j \boldsymbol{v}_j^{\mathsf{T}}$.

For $\boldsymbol{u} \in \mathbb{R}^n$ and $\boldsymbol{v} \in \mathbb{R}^n$ with $\boldsymbol{v} \neq \boldsymbol{0}$, $\|\boldsymbol{u}\boldsymbol{v}^{\mathsf{T}}\|_2 = \|\boldsymbol{u}\|_2 / \|\boldsymbol{v}\|_2$.

If $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ has rank$(\boldsymbol{A}) = r$, then $\|\boldsymbol{A}^{\mathsf{T}}\|_2 = 1/\sigma_r$.

Assuming that $\boldsymbol{A}$ and $\boldsymbol{E} \in \mathbb{R}^{m \times n}$ with rank$(\boldsymbol{A}) = n$, if $\|\boldsymbol{E}\|_2 \|\boldsymbol{A}^{\mathsf{T}}\|_2 < 1$, then rank$(\boldsymbol{A} + \boldsymbol{E}) = n$.

For $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with r nonzero singular values, the Frobenius norm is the l_2 norm of its vectorisation:

$$\|\boldsymbol{A}\|_F = \|\text{vec}(\boldsymbol{A})\|_2 = \sqrt{\text{trace}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})} = \sqrt{\sum_{i=1}^{r} \sigma_i^2},$$

which is submultiplicative and invariant under rotation, i.e. $\|\boldsymbol{A}\|_F^2 = \|\boldsymbol{A}\boldsymbol{R}\|_F^2 = \|\boldsymbol{R}\boldsymbol{A}\|_F^2$ for $\boldsymbol{R}^{\mathsf{T}}\boldsymbol{R} = \boldsymbol{I}$; and the Schatten p-norm is given by

$$\|\boldsymbol{A}\|_{S(p)} = (\sum_{i=1}^{r} \sigma_i^p)^{1/p}.$$

These are submultiplicative and rotationally invariant as well. The special case of $p = 1$ is referred to as the nuclear norm

$$\|\boldsymbol{A}\|_* = \sum_{i=1}^{r} \sigma_i.$$

For $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, the $l_{2,1}$ norm is the sum of column two norms:

$$\|\boldsymbol{A}\|_{2,1} = \sum_{j=1}^{n} (\sum_{i=1}^{m} |a_{ij}|^2)^{1/2};$$

the max norm is the infinity norm of its vectorisation:

$$\|\boldsymbol{A}\|_{\max} = \|\text{vec}(\boldsymbol{A})\|_{\infty};$$

and the l_1 norm is the one norm of its vectorisation:

$$\|\boldsymbol{A}\|_{l_1} = \|\text{vec}(\boldsymbol{A})\|_{1}.$$

Overall, for matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ of rank r, we have the following relationships:

- $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_F \leq \sqrt{r}\|\boldsymbol{A}\|_2$;
- $\|\boldsymbol{A}\|_F \leq \|\boldsymbol{A}\|_* \leq \sqrt{r}\|\boldsymbol{A}\|_F$;
- $\|\boldsymbol{A}\|_{\max} \leq \|\boldsymbol{A}\|_2 \leq \sqrt{mn}\|\boldsymbol{A}\|_{\max}$.

Every norm and norms of affine functions are convex functions. The squared Frobenius norm is strongly convex.

1.4 Constrained Optimisation

A general formulation of constrained optimisation is

$$\begin{aligned} &\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}), \\ &\text{s.t. } c_i(\boldsymbol{x}) = 0, i \in \mathcal{E}, \\ &\quad\;\; c_i(\boldsymbol{x}) \geq 0, i \in \mathcal{I}, \end{aligned}$$

where f and the functions c_i are all smooth, real-valued functions on a subset of $\mathbb{R}^n$, and $\mathcal{I}$ and $\mathcal{E}$ are two finite sets of indices.

The active set $\mathcal{A}(\boldsymbol{x})$ at any feasible $\boldsymbol{x}$ consists of the equality constraint indices from $\mathcal{E}$ together with the indices of the inequality constraints i for which $c_i(\boldsymbol{x}) = 0$; that is,

$$\mathcal{A}(\boldsymbol{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | c_i(\boldsymbol{x}) = 0\}.$$

Constraint qualifications ensure that the linearised feasible set is similar to the tangent cone. The linear independence constraint qualification (LICQ) is used most often.

Given the point $\boldsymbol{x}$ and the active set $\mathcal{A}(\boldsymbol{x})$, LICQ holds if the set of active constraint gradients $\{\nabla c_i(\boldsymbol{x}), i \in \mathcal{A}(\boldsymbol{x})\}$ is linearly independent.

The condition that all active constraints be linear is another possible constraint qualification. It is neither weaker nor stronger than the LICQ condition. Other exceedingly weaker conditions include the Mangasarian-Fromovitz constraint qualification, the Abadie constraint qualification and the Guignard constraint qualification.

The Lagrangian function for the problem is defined as

$$\mathcal{L}(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(\boldsymbol{x}).$$

The first-order necessary conditions are known as the Karush-Kuhn-Tucker (KKT) conditions and are provided below. Suppose that $\boldsymbol{x}^*$ is a local solution, that the functions f and c_i are continuously differentiable, and that the LICQ holds at $\boldsymbol{x}^*$. Then there is a Lagrange multiplier vector λ^*, with components $\lambda_i^*, i \in \mathcal{E} \cup \mathcal{I}$, such that the following conditions are satisfied at $(\boldsymbol{x}^*, \lambda^*)$

$$\begin{aligned}
\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \lambda^*) &= 0, \\
c_i(\boldsymbol{x}^*) &= 0, \quad \text{for all } i \in \mathcal{E}, \\
c_i(\boldsymbol{x}^*) &\geq 0, \quad \text{for all } i \in \mathcal{I}, \\
\lambda_i^* &\geq 0, \quad \text{for all } i \in \mathcal{I}, \\
\lambda_i^* c_i(\boldsymbol{x}^*) &= 0,\text{for all } i \in \mathcal{E} \cup \mathcal{I}.
\end{aligned}$$

For a given problem and solution point $\boldsymbol{x}^*$, there may be many vectors λ^* for which the conditions are satisfied. When the LICQ holds, however, the optimal λ^* is unique. As such, in problems such as Lasso, where a bound in an inequality needs to be tuned,

$$\begin{aligned}
&\min f(\boldsymbol{x}) \\
&\text{s.t. } g(\boldsymbol{x}) \leq t,
\end{aligned}$$

we can tune the Lagrange multiplier in the unconstrained Lagrangian form instead,

$$\min f(\boldsymbol{x}) + \lambda g(\boldsymbol{x}).$$

Though there is in general no analytic relationship between λ and t.

1.5 The Alternating Direction Method of Multipliers

A common approach to constrained optimisation is to replace the original problem by a sequence of subproblems. One subproblem consists of the original objective and one additional term for each constraint. The additional non-negative penalty terms are multiplied by a positive coefficient. The larger the coefficient, the closer to the allowed region for the minimiser.

As an example, in the context of the equality-constrained problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x}) \quad \text{subject to } c_i(\boldsymbol{x}) = 0,\ i \in \xi,$$

where ξ is a finite set of indices, the quadratic penalty function for the subproblem is

$$Q(\boldsymbol{x},\mu) \equiv f(\boldsymbol{x}) + \frac{\mu}{2}\sum_{i\in\xi} c_i^2(\boldsymbol{x}),$$

where μ is the penalty parameter. To search for the optimal $\boldsymbol{x}$, we can consider a sequence of values $\{\mu_k\}$ such that $\mu_k \to \infty$ as $k \to \infty$. Then, the minimisers $\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}$ can provide an initial guess for $\boldsymbol{x}_k$.

Algorithms like quasi-Newton, conjugate gradient and Newton's method can be used to solve such smooth objectives. But when μ_k becomes large, some eigenvalues of the Hessian matrix are of order μ_k. This ill conditioning of the Hessian upsets quasi-Newton and conjugate gradient methods directly. On the other hand, Newton's method is disturbed by large μ_k for other reasons.

Another drawback of the quadratic penalty function is that although the sequence $\{\boldsymbol{x}_k\}$ converges as $\mu_k \to \infty$, the intermediate approximate minimisers, $\{\boldsymbol{x}_k\}$, suffer systematic error:

$$c_i(\boldsymbol{x}_k) \approx -\frac{\lambda_i^*}{\mu_k}, \quad \text{for all } i \in \xi,$$

from relaxed constraints at moderate μ_k.

The method of multipliers enforces the feasibility conditions $c_i(\boldsymbol{x}) = 0$, $i \in \xi$, by explicitly including the Lagrange multiplier term to arrive at the objective of the augmented Lagrangian function:

$$\mathcal{L}_A(\boldsymbol{x},\boldsymbol{\lambda};\mu) \equiv f(\boldsymbol{x}) - \sum_{i\in\xi}\lambda_i c_i(\boldsymbol{x}) + \frac{\mu}{2}\sum_{i\in\xi} c_i^2(\boldsymbol{x}).$$

From the optimality condition, we can deduce that

$$\lambda_i^* \approx \lambda_i^k - \mu_k c_i(\boldsymbol{x}_k), \quad \text{for all } i \in \xi.$$

Upon rearranging, we see that

$$c_i(\boldsymbol{x}_k) \approx -\frac{1}{\mu_k}(\lambda_i^* - \lambda_i^k).$$

So if λ_k is chosen to be close to the optimal multiplier vector $\boldsymbol{\lambda}^*$, the infeasibility in $\boldsymbol{x}_k$ will be improved to be much smaller than $1/\mu_k$. Hence, we update the Langrange multiplier vector according to the following

$$\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(\boldsymbol{x}_k), \quad \text{for all } i \in \xi.$$

This leads to the following algorithm

Algorithm 1 Augmented Lagrangian Method

```
Input: Penalty Parameter $\mu_0$ ($\mu_0 > 0$), Tolerances $\{\tau_k\}$ ($\tau_k > 0$), Update Ratios $\{\rho_k\}$ ($\rho_k > 1$)
    Initial Guesses $\boldsymbol{x}_0^s$ & $\boldsymbol{\lambda}_0$
while not converged or $|| \bigtriangledown_{\boldsymbol{x}} \mathcal{L}_A(\boldsymbol{x}_k, \boldsymbol{\lambda}_k; \mu_k) > \tau_k||$ do
    $\boldsymbol{x}_k = \underset{\boldsymbol{x}}{\operatorname{argmin}} \mathcal{L}_A(\boldsymbol{x}, \boldsymbol{\lambda}_k; \mu_k)$
    $\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(\boldsymbol{x}_k)$
    $\mu_{k+1} = \rho_k \mu_k$
    $\boldsymbol{x}_{k+1}^s = \boldsymbol{x}_k$
end while
return $\boldsymbol{x}_k$
```

This algorithm can be shown to converge with a finite increase in μ. So ill conditioning is less of a problem. Moreover, choice of the starting point $\boldsymbol{x}^s$ is less critical. As a result, simply starting at the previous approximate minimiser $\boldsymbol{x}_k$ will suffice.

The variable $\boldsymbol{x}$ can be split into two parts in the example problem, $\boldsymbol{y}$ and $\boldsymbol{z}$:

$$\min_{\boldsymbol{y},\boldsymbol{z}} f(\boldsymbol{x}, \boldsymbol{y}) \quad \text{subject to } c_i(\boldsymbol{y}, \boldsymbol{z}) = 0, i \in \xi.$$

If $f(\cdot,\cdot)$ is bi-convex, separable and $g(\cdot,\cdot)$ is bi-affine, it can be transformed to the following:

$$\min_{\boldsymbol{y},\boldsymbol{z}} g(\boldsymbol{y}) + h(\boldsymbol{z}) \quad \text{subject to } c_i(\boldsymbol{y}, \boldsymbol{z}) = 0, i \in \xi,$$

where $g(\cdot)$ and $h(\cdot)$ are convex respectively. As in the method of multipliers, we consider the augmented Lagrangian

$$\mathcal{L}_A(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\lambda}; \mu) = g(\boldsymbol{y}) + h(\boldsymbol{z}) - \sum_{i\in\xi} \lambda_i c_i(\boldsymbol{y}, \boldsymbol{z}) + \frac{\mu}{2} \sum_{i\in\xi} c_i^2(\boldsymbol{y}, \boldsymbol{z}).$$

Instead of minimising the proximity function $\mathcal{L}_A(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\lambda}; \mu)$ w.r.t. $\boldsymbol{y}$ and $\boldsymbol{z}$ jointly in the dual update, the alternating direction method of multipliers (ADMM) minimises primal variables in an alternating manner, that is, first solving for $\boldsymbol{y}$ with $\boldsymbol{z}$ fixed and then solving for $\boldsymbol{z}$ with $\boldsymbol{y}$ fixed. Rather than iterate until convergence, the algorithm proceeds directly to updating the dual variable and moves on to the next subproblem. The algorithm is presented next

Algorithm 2 Alternating Direction Method of Multipliers

```
Input: Penalty Parameter $\mu_0$ ($\mu_0 > 0$), Update Ratios $\{\rho_k\}$ ($\rho_k > 1$) Initial Guesses $\boldsymbol{y}_0^s$, $\boldsymbol{z}_0^s$ & $\boldsymbol{\lambda}_0$
  while not converged do
    $\boldsymbol{y}_k = \underset{\boldsymbol{y}}{\operatorname{argmin}} \mathcal{L}_A(\boldsymbol{y}, \boldsymbol{z}_k^s, \boldsymbol{\lambda}_k; \mu_k)$
    $\boldsymbol{z}_k = \underset{\boldsymbol{z}}{\operatorname{argmin}} \mathcal{L}_A(\boldsymbol{y}_k, \boldsymbol{z}, \boldsymbol{\lambda}_k; \mu_k)$
    $\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(\boldsymbol{y}_k, \boldsymbol{z}_k)$
    $\mu_{k+1} = \rho_k \mu_k$
    $\boldsymbol{z}_{k+1}^s = \boldsymbol{z}_k$
  end while
  return $\boldsymbol{y}_k$, $\boldsymbol{z}_k$
```

It can be shown that ADMM achieves residual convergence, objective convergence and dual variable convergence. It is also known as the inexact ALM. Among other connections, ADMM is a special case of the Douglas-Rachford splitting method for monotone operators [17], which in turn belongs to the proximal point algorithm [17].

1.6 Robust Principal Component Analysis

We are given a large observation matrix $\boldsymbol{M}$ that is known to decompose into two components such that $\boldsymbol{M} = \boldsymbol{L}_0 + \boldsymbol{S}_0$. We want to exactly identity both $\boldsymbol{L}_0$ and $\boldsymbol{S}_0$ without the knowledge of their magnitude, the subspace or even the rank of $\boldsymbol{L}_0$, the locations or even the number of corrupted entries of $\boldsymbol{S}_0$. In other words, ponent analysis (RPCA) aims to recover a low-rank matrix $\boldsymbol{L}_0$ and a gross but sparse $\boldsymbol{S}_0$.

Even though the rank of the matrix $\boldsymbol{L}_0$ and the number of non-zero entries in $\boldsymbol{S}_0$ are both NP-hard to calculate, their convex relaxations still allow us to exactly recover $\boldsymbol{L}_0$ and $\boldsymbol{S}_0$. More specifically, the principal component pursuit (PCP) solves the following convex optimisation problem

$$\begin{aligned} \min \quad & \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{S}\|_1 \\ \text{s.t.} \quad & \boldsymbol{L} + \boldsymbol{S} = \boldsymbol{M}. \end{aligned} \tag{1.1}$$

Roughly, even if the rank of $\boldsymbol{L}_0$ grows linearly with the dimension of the matrix and there is a constant fraction of errors, PCP can still achieve exact recovery. More formally, let us denote the singular value decomposition of $\boldsymbol{L}_0 \in \mathbb{R}^{n_1 \times n_2}$ by

$$\boldsymbol{L}_0 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*. \tag{1.2}$$

Theorem 1. *Suppose that we are given $\boldsymbol{L}_0 \in \mathbb{R}^{n \times n}$ such that*

$$\max_i \|\boldsymbol{U}^* \boldsymbol{e}_i\|^2 \leq \frac{\mu r}{n}, \quad \max_i \|\boldsymbol{V}^* \boldsymbol{e}_i\|^2 \leq \frac{\mu r}{n}, \tag{1.3}$$

and

$$\|\boldsymbol{U}\boldsymbol{V}^*\|_\infty \leq \sqrt{\frac{\mu r}{n^2}}, \tag{1.4}$$

where μ is a parameter and r is the rank of $\boldsymbol{L}_0$. And it is assumed that the support set of $\boldsymbol{S}_0$ is uniformly distributed among all sets of cardinality m. Then there is a numerical constant c such that with probability at least $1 - cn^{-10}$ (over the choice of support of $\boldsymbol{S}_0$), PCP with $\lambda = 1/\sqrt{n}$ is exact, i.e. $\hat{\boldsymbol{L}} = \boldsymbol{L}_0$ and $\hat{\boldsymbol{S}} = \boldsymbol{S}_0$ for solutions $\hat{\boldsymbol{L}}$ and $\hat{\boldsymbol{S}}$ to (1.1), provided that

$$r \leq \rho_r n \mu^{-1} (\log n)^{-2}, \quad m \leq \rho_s n^2, \tag{1.5}$$

where ρ_r and ρ_s are positive numerical constants.

Proof. See [21]. □

In the case of missing values, suppose that we know the orthogonal projection operator $\boldsymbol{\Pi}_\Omega$ onto the linear space of matrices supported on $\Omega \subset [n_1] \times [n_2]$:

$$\boldsymbol{\Pi}_\Omega \boldsymbol{X} = \begin{cases} \boldsymbol{X}_{ij} & (i,j) \in \Omega \\ \boldsymbol{0} & (i,j) \notin \Omega. \end{cases} \tag{1.6}$$

(1.1) can be modified to solve the following problem with perfect recovery [21]:

$$\begin{aligned} \min \quad & \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 \\ \text{s.t.} \quad & \boldsymbol{\Pi}_\Omega(\boldsymbol{L} + \boldsymbol{S}) = \boldsymbol{M}. \end{aligned} \tag{1.7}$$

To efficiently solve (1.1), the augmented Lagrange multiplier algorithm, a.k.a. ADMM, can be adopted. The associated augmented Lagrangian is given by

$$\mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y}) = \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 + \langle \boldsymbol{Y}, \boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S} \rangle + \frac{\mu}{2} \|\boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S}\|_F^2, \tag{1.8}$$

Instead of iterative updates of $(\boldsymbol{L}_k, \boldsymbol{S}_k) = \operatorname{argmin}_{\boldsymbol{L},\boldsymbol{S}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y}_k)$ and the Lagrange multiplier matrix $\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k + \mu(\boldsymbol{M} - \boldsymbol{L}_k - \boldsymbol{S}_k)$, ADMM first minimises l with respect to $\boldsymbol{L}$ fixing $\boldsymbol{S}$ ($\min_{\boldsymbol{L}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}_k, \boldsymbol{Y}_k)$), then minimises l with respect to $\boldsymbol{S}$ fixing $\boldsymbol{L}$ ($\min_{\boldsymbol{S}} \mathcal{L}(\boldsymbol{L}_{k+1}, \boldsymbol{S}, \boldsymbol{Y}_k)$) and finally updates the Lagrange multiplier matrix $\boldsymbol{Y}$.

Let us define the shrinkage operator that is helpful in solving the subproblems:

$$\mathcal{S}_\tau(x) = \operatorname{sgn}(x)\max(|x|-\tau, 0). \tag{1.9}$$

For a matrix, the shrinkage operator $\mathcal{S}_\tau$ applies to each element separately. For $\min_{\boldsymbol{L}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y})$, we have

$$\operatorname*{argmin}_{\boldsymbol{L}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y}) = \mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{M} - \boldsymbol{L} + \mu^{-1}\boldsymbol{Y}). \tag{1.10}$$

Denoting the singular value thresholding operator by $\mathcal{D}_\tau(\boldsymbol{X}) = \boldsymbol{U}\mathcal{S}_\tau(\boldsymbol{\Sigma})\boldsymbol{V}^*$ for any singluar value decomposition $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$, the problem $\min_{\boldsymbol{S}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y})$ has a solution

$$\operatorname*{argmin}_{\boldsymbol{S}} \mathcal{L}(\boldsymbol{L}, \boldsymbol{S}, \boldsymbol{Y}) = \mathcal{D}_{\mu^{-1}}(\boldsymbol{M} - \boldsymbol{S} + \mu^{-1}\boldsymbol{Y}). \tag{1.11}$$

The overall algorithm is summarised below

Algorithm 3 PCP by ADMM

```
Input: $\mu$
  $\boldsymbol{S}_0 = \boldsymbol{Y}_0 = \boldsymbol{0}$
  while not converged do
    $\boldsymbol{L}_{k+1} = \mathcal{D}_{\mu^{-1}}(\boldsymbol{M} - \boldsymbol{S}_k + \mu^{-1}\boldsymbol{Y}_k)$
    $\boldsymbol{S}_{k+1} = \mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{M} - \boldsymbol{L}_{k+1} + \mu^{-1}\boldsymbol{Y}_k)$
    $\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k + \mu(\boldsymbol{M} - \boldsymbol{L}_{k+1} - \boldsymbol{S}_{k+1})$
  end while
  return $\boldsymbol{L}$, $\boldsymbol{S}$
```

The above algorithm is known to be convergent [17]. The most computationally heavy step is the singular value thresholding operation $\mathcal{D}_{\mu^{-1}}$ which requires the singular value decomposition. Empirically, μ can be set to $\frac{n_1 n_2}{4\|\boldsymbol{M}\|_1}$ and the algorithm is terminated when $\|\boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S}\|_F \leq 10^{-7}\|\boldsymbol{M}\|_F$.

1.7 Robust Principal Component Analysis with Missing Values

In the case of missing data, robust matrix recovery methods [29, 109] enhance PCP to deal with occlusions:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{S} \circ \boldsymbol{W}\|_1 \quad \text{s.t.} \quad \boldsymbol{X} = \boldsymbol{L} + \boldsymbol{S}, \tag{1.12}$$

where $\boldsymbol{W}$ is the matrix of binary occlusion masks. Its Jacobi-type update schemes can be implemented in parallel and hence are attractive for solving large-scale problems. Disgruntled at the unrealistic uniform sampling assumption for missing entries, Liu et al. [69] set out to use the isomeric condition hypothesis to tackle irregular and deterministic missing data.

1.8 Robust Principal Component Analysis with Features

Several machine learning and signal processing tasks involve the separation of a data matrix into a low-rank matrix and a matrix with sparse support (i.e., a sparse matrix) containing entries of arbitrary magnitude. Robust principal component analysis (RPCA) [21, 24] offers a provably correct and convenient way to solve this matrix separation problem, when certain incoherence conditions on the data hold. In fact, RPCA solves a convex relaxation of the natural rank minimization problem which is regularized by the sparsity promoting ℓ_0-(quasi) norm. Nevertheless, prior side information, oftentimes in the form of features, may also be present in practice. For instance, features are available for the following tasks:

- Collaborative filtering: apart from ratings of an item by other users, the profile of the user and the description of the item can also be exploited in making recommendations [31];
- Relationship prediction: user behaviours and message exchanges can assist in finding missing links on social media networks [134];
- Person-specific facial deformable models: an orthonormal subspace learnt from manually annotated data captured in-the-wild, when fed into an image congealing procedure, can help produce more correct fittings [100].

It is thus reasonable to investigate whether it is propitious for RPCA to exploit the available side information by incorporating features. Indeed, recent results [69], indicate that in case of union of multiple subspaces where RPCA degrades due to the increasing row-coherence (when the number of subspaces grows), the use of features as side information allow accurate low-rank recovery by removing its dependency on the row-coherence. Despite the theoretical and practical merits of convex variants of RPCA with features, such as LRR [68] and PCPF [32], convex relaxations of the rank function and l_0-norm result into *algorithm weakening* [25].

If feasible feature dictionaries, $\boldsymbol{X}$ and $\boldsymbol{Y}$, regarding row and column spaces are available, PCPF [32] makes use of these to generalize (1.1) to the below objective:

$$\min_{\boldsymbol{H},\boldsymbol{S}} \ \|\boldsymbol{H}\|_* + \gamma\|\boldsymbol{S}\|_1 \quad \text{subject to} \quad \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\mathsf{T} + \boldsymbol{S} = \boldsymbol{M}, \tag{1.13}$$

for the same γ as in (1.1). Convergence to the RPCA solution has only been established for the random sparsity model.

IRPCA-IHT [83] includes features $\boldsymbol{X}$, $\boldsymbol{Y}$ in an iterative non-convex projection algorithm. Similar to AltProj, at each step, a new sparse estimate is calculated from hard thresholding via a monotonically decreasing threshold. After that, spectral hard thresholding takes place to attain the low-rank estimate. IRPCA-IHT provably converges to the solution of RPCA.

1.9 Robust Principal Component Analysis with Side Information

Principal Component Pursuit (PCP) as proposed in [21,24] and its variants e.g. [4,6,19,109,133,143] are the current methods of choice for recovering a low-rank subspace from a set of grossly corrupted and possibly incomplete high-dimensional data. PCP employs the nuclear norm and the l_1 norm (convex surrogates of the rank and sparsity constraints, respectively) in order to approximate the original l_0 norm regularised rank minimisation problem. In particular, under certain conditions (such as the restricted isometry property [20]), the relaxation gap is zero and rank minimisation is equivalent to nuclear norm minimisation. However, these conditions rarely hold for real-world visual data and PCP thus occasionally yields degenerate or suboptimal solutions. To alleviate this, it is advantageous for PCP to take into account of domain-dependent prior knowledge [55], i.e. side information [129].

The use of side information has been studied in the context of matrix completion [31, 134] and compressed sensing [79]. Recently, side information has been applied to the PCP framework in the *noiseless* case [33,101]. In particular, an error-free orthogonal column space was used to drive a PCP-based deformable image alignment algorithm [101]. More generally, Chiang et al. [33] used both a column and a row space as side information and the algorithm had to recover the weights of their interaction. The main limitation of such methods is that they require a set of clean, noise-free data samples in order to determine the column and/or row spaces of the low-rank component. Clearly, these data are are difficult to find in practice.

Shahid et al. [106, 107] incorporate structural knowledge into RPCA by adding spectral graph regularisation. Given the graph Laplacian $\mathbf{\Phi}$ of each data similarity graph, Robust PCA on Graphs (RPCAG) and Fast Robust PCA on Graphs (FRPCAG) add an additional $\mathrm{tr}(\boldsymbol{L}\Phi\boldsymbol{L}^\intercal)$ term to the PCP objective for the low-rank component $\boldsymbol{L}$. The main drawback of the above mentioned models is that the side information needs to be accurate and noiseless, which is not trivial in practical scenarios.

1.10 Tensors

1.10.1 matrix products

The Kronecker product of $\boldsymbol{A} \in \mathbb{R}^{I\times K}$ and $\boldsymbol{B} \in \mathbb{R}^{J\times L}$ is the $IJ \times KL$ matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}A_{11} & \boldsymbol{B}A_{12} & \dots & \boldsymbol{B}A_{1K} \\ \boldsymbol{B}A_{21} & \boldsymbol{B}A_{22} & \dots & \boldsymbol{B}A_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{B}A_{I1} & \boldsymbol{B}A_{I2} & \dots & \boldsymbol{B}A_{IK} \end{bmatrix}.$$

$\boldsymbol{b}^{\mathsf{T}} \otimes \boldsymbol{a} = \boldsymbol{a}\boldsymbol{b}^{\mathsf{T}}$ and $\text{vec}(\boldsymbol{a}\boldsymbol{b}^{\mathsf{T}}) = \boldsymbol{b} \otimes \boldsymbol{a}$ immediately follow. For $\boldsymbol{A} \in \mathbb{R}^{I\times K}, \boldsymbol{M} \in \mathbb{R}^{K\times L}$ and $\boldsymbol{B} \in \mathbb{R}^{J\times L}$, $\text{vec}(\boldsymbol{A}\boldsymbol{M}\boldsymbol{B}^{\mathsf{T}}) = (\boldsymbol{B} \otimes \boldsymbol{A})\text{vec}(\boldsymbol{M})$. This is useful when dealing with the linear least squares problem

$$\min_{\boldsymbol{M}} \|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{M}\boldsymbol{B}^{\mathsf{T}}\|_F^2 = \min_{\boldsymbol{m}} \|\text{vec}(\boldsymbol{X}) - (\boldsymbol{B} \otimes \boldsymbol{A})\boldsymbol{m}\|_2^2,$$

where $\boldsymbol{m} = \text{vec}(\boldsymbol{M})$.

For two matrices with the same number of columns $\boldsymbol{A} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_l]$ and $\boldsymbol{B} = [\boldsymbol{b}_1, \cdots, \boldsymbol{b}_l]$, the Khatri-Rao product is

$$\boldsymbol{A} \odot \boldsymbol{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1, \cdots, \boldsymbol{a}_l \otimes \boldsymbol{b}_l].$$

If $\boldsymbol{D}$ is a diagonal matrix with diagonal elements in the vector $\boldsymbol{d}$, the property $\text{vec}(\boldsymbol{A}\boldsymbol{D}\boldsymbol{B}^{\mathsf{T}}) = (\boldsymbol{B} \odot \boldsymbol{A})\boldsymbol{d}$ holds. This is useful when dealing with the linear least squares problem

$$\min_{\text{diagnoal } \boldsymbol{D}} \|\boldsymbol{X} - \boldsymbol{A}\boldsymbol{D}\boldsymbol{B}^{\mathsf{T}}\|_F^2 = \min_{\boldsymbol{d}} \|\text{vec}(\boldsymbol{X}) - (\boldsymbol{B} \odot \boldsymbol{A})\boldsymbol{d}\|_2^2.$$

The Hadamard product is the elementwise matrix product. Given matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{I\times J}$, the Hadamard product $\boldsymbol{A} \circ \boldsymbol{B}$ is defined by

$$\begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \dots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \dots & a_{2J}b_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \dots & a_{IJ}b_{IJ} \end{bmatrix}.$$

We also have the following properties:

- $(\boldsymbol{A} \otimes \boldsymbol{B}) \otimes \boldsymbol{C} = \boldsymbol{A} \otimes (\boldsymbol{B} \otimes \boldsymbol{C}) = \boldsymbol{A} \otimes \boldsymbol{B} \otimes \boldsymbol{C}$;

- $(\boldsymbol{A} \otimes \boldsymbol{B})^{\mathsf{T}} = (\boldsymbol{A}^{\mathsf{T}} \otimes \boldsymbol{B}^{\mathsf{T}})$;

- $(\boldsymbol{A} \otimes \boldsymbol{B})(\boldsymbol{E} \otimes \boldsymbol{F}) = (\boldsymbol{A}\boldsymbol{E} \otimes \boldsymbol{B}\boldsymbol{F})$;

- $(\boldsymbol{A} \otimes \boldsymbol{B})^{\dagger} = \boldsymbol{A}^{\dagger} \otimes \boldsymbol{B}^{\dagger}$;
- For $\boldsymbol{A} = \boldsymbol{U}_1 \boldsymbol{\Sigma}_1 \boldsymbol{V}_1^{\mathsf{T}}$ and $\boldsymbol{B} = \boldsymbol{U}_2 \boldsymbol{\Sigma}_2 \boldsymbol{V}_2^{\mathsf{T}}$, we have $\boldsymbol{A} \otimes \boldsymbol{B} = (\boldsymbol{U}_1 \otimes \boldsymbol{U}_2)(\boldsymbol{\Sigma}_2 \otimes \boldsymbol{\Sigma}_2)(\boldsymbol{V}_1 \otimes \boldsymbol{V}_2)^{\mathsf{T}}$;
- $\text{rank}(\boldsymbol{A} \otimes \boldsymbol{B}) = \text{rank}(\boldsymbol{A})\text{rank}(\boldsymbol{B})$;
- For square matrices $\boldsymbol{A}, \boldsymbol{B}$, $\text{tr}(\boldsymbol{A} \otimes \boldsymbol{B}) = \text{tr}(\boldsymbol{A})\text{tr}(\boldsymbol{B})$;
- For square matrices $\boldsymbol{A}, \boldsymbol{B}$, $\det(\boldsymbol{A} \otimes \boldsymbol{B}) = \det(\boldsymbol{A})\det(\boldsymbol{B})$;
- $(\boldsymbol{A} \odot \boldsymbol{B}) \odot \boldsymbol{C} = \boldsymbol{A} \odot (\boldsymbol{B} \odot \boldsymbol{C}) = \boldsymbol{A} \odot \boldsymbol{B} \odot \boldsymbol{C}$;
- $(\boldsymbol{A} \otimes \boldsymbol{B})(\boldsymbol{E} \odot \boldsymbol{F}) = (\boldsymbol{AE}) \odot (\boldsymbol{BF})$;
- $(\boldsymbol{A} \odot \boldsymbol{B})^{\mathsf{T}}(\boldsymbol{A} \odot \boldsymbol{B}) = \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} * \boldsymbol{B}^{\mathsf{T}}\boldsymbol{B}$;
- $(\boldsymbol{A} \odot \boldsymbol{B})^{\dagger} = ((\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}) * (\boldsymbol{B}^{\mathsf{T}}\boldsymbol{B}))^{\dagger}(\boldsymbol{A} \odot \boldsymbol{B})^{\mathsf{T}}$.

The tensor product or outer product of vectors $\boldsymbol{a} \in \mathbb{R}^I$ and $\boldsymbol{b} \in \mathbb{R}^J$ is the $I \times J$ matrix $\boldsymbol{a} \circ \boldsymbol{b}$ with elements $(\boldsymbol{a} \circ \boldsymbol{b})_{ij} = a_i b_j$. The outer product of three vectors is an $I \times J \times K$ three-way array or three-way tensor $\boldsymbol{a} \circ \boldsymbol{b} \circ \boldsymbol{c}$ with elements $(\boldsymbol{a} \circ \boldsymbol{b} \circ \boldsymbol{c})_{ijk} = a_i b_j c_k$.

1.10.2 Tensor Basics

The order of a tensor is the number of dimensions, also known as ways or modes.

A fibre is defined by fixing every index but one. If the n^{th} index is free, then the fibre is called n-mode. Similarly, slices are defined by fixing every index but two.

Matricisation, also known as unfolding or flattening, is the process of reordering the elements of an N-way array into a matrix. The mode-n matricisation of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is denoted by $\boldsymbol{X}_{(n)}$ and arranges the mode-n fibres to be the columns of the resulting matrix. Element $\mathcal{X}_{i_1 i_2 \cdots i_N}$ becomes $X_{i_n j}$ where

$$j = 1 + \sum_{k=1, k \neq n}^{N} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{m=1, m \neq n}^{k-1} i_m.$$

Multiplying a tensor by a matrix in mode n is known as the n-mode product. The n-mode product of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\boldsymbol{U} \in \mathbb{R}^{J \times I_n}$ is denoted by $\boldsymbol{\mathcal{X}} \times_n \boldsymbol{U}$ and is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$. Elementwise,

$$(\boldsymbol{\mathcal{X}} \times_n \boldsymbol{U})_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \cdots i_N} u_{j i_n}.$$

By definition, we have the following properties:

- $\boldsymbol{Y}_{(n)} = (\boldsymbol{\mathcal{X}} \times_n \boldsymbol{U})_{(n)} = \boldsymbol{U}\boldsymbol{X}_{(n)}$;
- $\boldsymbol{\mathcal{X}} \times_m \boldsymbol{A} \times_n \boldsymbol{B} = \boldsymbol{\mathcal{X}} \times_n \boldsymbol{B} \times_m \boldsymbol{A}$ $(m \neq n)$;
- $\boldsymbol{\mathcal{X}} \times_n \boldsymbol{A} \times_n \boldsymbol{B} = \boldsymbol{\mathcal{X}} \times_n (\boldsymbol{B}\boldsymbol{A})$;
- $\boldsymbol{\mathcal{X}} \times_m \boldsymbol{a} \times_n \boldsymbol{b} = (\boldsymbol{\mathcal{X}} \times_m \boldsymbol{a}) \times_{n-1} \boldsymbol{b} = (\boldsymbol{\mathcal{X}} \times_n \boldsymbol{b}) \times_m \boldsymbol{a}$ for $m < n$;
- $\boldsymbol{Y}_{(n)} = (\boldsymbol{\mathcal{X}} \times_1 \boldsymbol{A}^{(1)} \times_2 \boldsymbol{A}^{(2)} \cdots \times_N \boldsymbol{A}^{(N)})_{(n)} = \boldsymbol{A}^{(n)}\boldsymbol{X}_{(n)}(\boldsymbol{A}^{(N)} \otimes \cdots \otimes \boldsymbol{A}^{(n+1)} \otimes \boldsymbol{A}^{(n-1)} \otimes \cdots \otimes \boldsymbol{A}^{(1)})^{\mathsf{T}}$;
- $\text{vec}(\boldsymbol{\mathcal{X}} \times_{i=1}^{N} \boldsymbol{U}_i) = (\otimes_{i=N}^{1}\boldsymbol{U}_i)\text{vec}(\boldsymbol{\mathcal{X}})$.

The Frobenius norm of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is

$$\|\boldsymbol{\mathcal{X}}\|_F = \sqrt{\sum_{i_1}^{I_1}\sum_{i_2}^{I_2} \cdots \sum_{i_N}^{I_N} \mathcal{X}_{i_1 i_2 \dots i_N}^2}.$$

1.10.3 CP decomposition

A rank-one three-way tensor $\boldsymbol{\mathcal{X}}$ of size $I \times J \times K$ is an outer product of three vectors $\boldsymbol{a} \in \mathbb{R}^I, \boldsymbol{b} \in \mathbb{R}^J, \boldsymbol{c} \in \mathbb{R}^K$, i.e. $\boldsymbol{\mathcal{X}} = \boldsymbol{a} \circ \boldsymbol{b} \circ \boldsymbol{c}$. A rank-one N-way tensor $\boldsymbol{\mathcal{X}}$ is likewise an outer product of N vectors: $\boldsymbol{\mathcal{X}} = \boldsymbol{a}_1 \circ \boldsymbol{a}_2 \circ \cdots \circ \boldsymbol{a}_N$.

The rank of tensor $\boldsymbol{\mathcal{X}}$ is the minimum number of rank-one tensors needed to produce $\boldsymbol{\mathcal{X}}$ as their sum. See Fig. 1.1 for a tensor of rank three. Therefore, a three-way tensor of rank F can be written as

$$\boldsymbol{\mathcal{X}} = \sum_{f=1}^{F} \boldsymbol{a}_f \circ \boldsymbol{b}_f \circ \boldsymbol{c}_f.$$

It is common to use the notation $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$, where $\boldsymbol{A} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_F]$, $\boldsymbol{B} = [\boldsymbol{b}_1, \cdots, \boldsymbol{b}_F]$ and $\boldsymbol{C} = [\boldsymbol{c}_1, \cdots, \boldsymbol{c}_F]$.

The following relationships hold

- $\boldsymbol{X}_{(1)} = \boldsymbol{A}(\boldsymbol{C} \odot \boldsymbol{B})^{\mathsf{T}}$;
- $\boldsymbol{X}_{(2)} = \boldsymbol{B}(\boldsymbol{C} \odot \boldsymbol{A})^{\mathsf{T}}$;

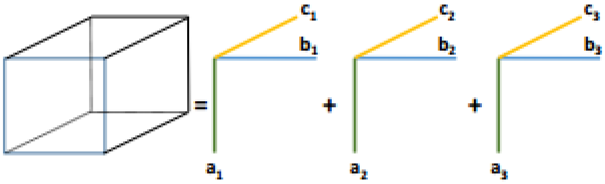


Figure 1.1: Schematic of tensor of rank three

Size	Maximum attainable rank
$I \times J \times 2$	$\min(I,J)+\min(I,J,\lfloor\max(I,J)/2\rfloor)$
$2 \times 2 \times 2$	3
$3 \times 3 \times 3$	5

Table 1.1: Maximum attainable ranks.

- $\boldsymbol{X}_{(3)} = \boldsymbol{C}(\boldsymbol{B} \odot \boldsymbol{A})^{\mathsf{T}}$;

- $\mathrm{vec}(\boldsymbol{X}^{\mathsf{T}}_{(3)}) = (\boldsymbol{C} \odot \boldsymbol{B} \odot \boldsymbol{A})\mathbf{1}$, where $\mathbf{1}$ an $F \times 1$ vector full of 1's.

The rank F of a three-way tensor $\boldsymbol{\mathcal{X}}$ satisfies the following inequality

$$\max(R_1, R_2, R_3) \leq F \leq \min(R_1R_2, R_2R_3, R_1R_3),$$

where R_n is the mode-n rank of $\boldsymbol{\mathcal{X}}$.

However, if $(\boldsymbol{C} \odot \boldsymbol{B})$ is full column rank as well as $\boldsymbol{A}$, then $\mathrm{rank}(\boldsymbol{X}_{(1)}) = F$ =rank($\boldsymbol{\mathcal{X}}$). For this to happen it is necessary that $JK \geq F$ and $I \geq F$. Hence for the matricisation of $\boldsymbol{\mathcal{X}}$ to be rank-revealing, it must be really small relative to the upper bound.

In general, the rank of a tensor for decomposition over $\mathbb{R}$ is a random variable that can take more than one value with positive probability. These values are called typical ranks. When there is only one typical rank, it is called generic rank. Constraints such as symmetry and orthogonality of the factor matrices or the matricisation can strongly affect the tensor rank. In particular, determining the rank of a tensor is NP-hard. Some rank examples are shown in tables 1.1, 1.2 and 1.3.

We can also observe essential uniqueness for such decomposition under mild conditions. Given a tensor $\boldsymbol{\mathcal{X}}$ of rank F, we say that its rank decomposition into F rank-one terms is essentially unique if the rank-one tensors in its decomposition are unique. Note that we have to allow the inherently unresolvable permutation and scaling ambiguity. That is, if $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$, with $\boldsymbol{A} : I \times F, \boldsymbol{B} : J \times F,$

Size	Typical ranks
$I \times I \times 2$	$\{I, I+1\}$
$I \times J \times 2, I > J$	$\min(I, 2J)$
$I \times J \times K, I > JK$	JK
$I \times J \times K, JK - J < I < JK$	I
$I \times J \times K, I = JK - J$	$\{I, I+1\}$

Table 1.2: Typical ranks.

Size	Typical ranks, symmetric frontal	Typical ranks, no symmetry
$I \times I \times 2$	$\{I, I+1\}$	$\{I, I+1\}$
$I \times 2 \times 2, I \geq 4$	3	4
$9 \times 3 \times 3$	6	9

Table 1.3: Ranks with symmetrical constraints.

and $\boldsymbol{C} : K \times F$, then essential uniqueness means that $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ are unique up to a common permutation and scaling/counter-scaling of columns, meaning that if $\boldsymbol{\mathcal{X}} = [[\bar{\boldsymbol{A}}, \bar{\boldsymbol{B}}, \bar{\boldsymbol{C}}]]$, for some $\bar{\boldsymbol{A}} : I \times F, \bar{\boldsymbol{B}} : J \times F$, and $\bar{\boldsymbol{C}} : K \times F$, then there exists a permutation matrix $\boldsymbol{\Pi}$ and diagonal scaling matrices $\boldsymbol{\Lambda}_1, \boldsymbol{\Lambda}_2, \boldsymbol{\Lambda}_3$ such that

$$\bar{\boldsymbol{A}} = \boldsymbol{A}\boldsymbol{\Pi}\boldsymbol{\Lambda}_1, \bar{\boldsymbol{B}} = \boldsymbol{B}\boldsymbol{\Pi}\boldsymbol{\Lambda}_2, \bar{\boldsymbol{C}} = \boldsymbol{C}\boldsymbol{\Pi}\boldsymbol{\Lambda}_3, \boldsymbol{\Lambda}_1\boldsymbol{\Lambda}_2\boldsymbol{\Lambda}_3 = \boldsymbol{I}.$$

The notion of Kruskal rank plays an important role in uniqueness results. The Kruskal rank k_A of an $I \times F$ matrix $\boldsymbol{A}$ is the largest integer k such that any k columns of $\boldsymbol{A}$ are linearly independent. Clearly, $k_A \leq r_A =$rank$(\boldsymbol{A}) \leq \min(I, F)$.

The general result is provided by Kruskal's theorem. Given $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$, with $\boldsymbol{A} : I \times F, \boldsymbol{B} : J \times F$, and $\boldsymbol{C} : K \times F$, if $k_A + k_B + k_C \geq 2F + 2$, then the decomposition of $\boldsymbol{\mathcal{X}}$ in terms of $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ is essentially unique.

A better uniqueness result can be obtained if one of the loading matrices is full column rank. Given $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$, with $\boldsymbol{A} : I \times F, \boldsymbol{B} : J \times F$, and $\boldsymbol{C} : K \times F$, and assuming $r_C = F$, it holds that the decomposition $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$ is essentially unique $\Leftrightarrow$ no linear combination of columns of $\boldsymbol{A} \odot \boldsymbol{B}$ can be written as $\otimes$ product of two vectors.

The full-rank condition on $\boldsymbol{C}$ can be relaxed if one considers higher-order compound matrices. Let $\boldsymbol{\mathcal{X}} = [[\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]]$, with $\boldsymbol{A} : I \times F, \boldsymbol{B} : J \times F$, and $\boldsymbol{C} : K \times F$. Assume $I \leq J \leq K$ without loss of generality. Let $i_{\text{max}} = \max\{i | 2^i \leq I\}$, and $j_{\text{max}} = \max\{j | 2^j \leq J\}$. If $F \leq 2^{i_{\text{max}} + j_{\text{max}} - 2}$, then the

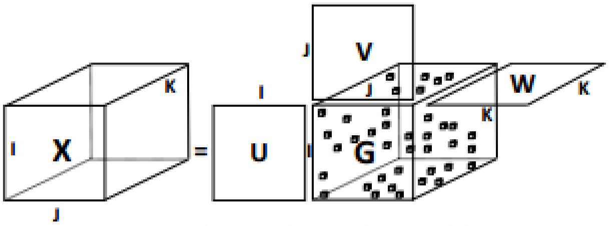


Figure 1.2: Tucker tensor decomposition.

decomposition of $\boldsymbol{\mathcal{X}}$ in terms of $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ is essentially unique, almost surely. In particular, when $I = 2^i, J = 2^j$, then almost sure uniqueness holds for $F \leq \frac{IJ}{4}$. More generally, a sufficient condition is $F \leq \frac{(I+1)(J+1)}{16}$.

The CANDECOMP/PARAFAC (CP) decomposition of a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ is the sum of R component rank-one tensors that best approximates $\boldsymbol{\mathcal{X}}$, i.e. to find

$$\min_{\{\boldsymbol{a}_r, \boldsymbol{b}_r, \boldsymbol{c}_r\}_{r=1}^R} \|\boldsymbol{\mathcal{X}} - \sum_{r=1}^{R} \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r\|_F^2,$$

where R is a positive integer and $\boldsymbol{a}_r \in \mathbb{R}^I, \boldsymbol{b}_r \in \mathbb{R}^J$, and $\boldsymbol{c}_r \in \mathbb{R}^K$ for $r = 1, \ldots, R$. If $R < F$, then the problem becomes the low-rank tensor approximation problem. However, a tensor may be arbitrarily well approximated by a lower rank tensor and this low rank is defined as the border rank. In cases like this, the problem is ill-posed. Normal fixes involve adding application-specific constraints, such as non-negativity or orthogonality of $\boldsymbol{a}_r, \boldsymbol{b}_r, \boldsymbol{c}_r$ or $\lambda(\|\boldsymbol{A}\|_F^2 + \|\boldsymbol{B}\|_F^2 + \|\boldsymbol{C}\|_F^2)$ for Gaussian priors.

1.10.4 Tucker decomposition

To generalise SVD to tensors, we consider a full (possibly dense, but ideally sparse) core tensor $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{P \times Q \times R}$ and matrices $\boldsymbol{U} \in \mathbb{R}^{I \times P}, \boldsymbol{V} \in \mathbb{R}^{J \times Q}$ and $\boldsymbol{W} \in \mathbb{R}^{K \times R}$, such that

$$\mathcal{X}_{ijk} = \sum_{l=1}^{I} \sum_{m=1}^{J} \sum_{n=1}^{K} \mathcal{G}_{lmn} U_{il} V_{jm} W_{kn},$$

where $\boldsymbol{u}_l = \boldsymbol{U}(:, l), \boldsymbol{v}_m = \boldsymbol{V}(:, m), \boldsymbol{w}_n = \boldsymbol{W}(:, n)$ and $\boldsymbol{U}^\intercal \boldsymbol{U} = \boldsymbol{I}, \boldsymbol{V}^\intercal \boldsymbol{V} = \boldsymbol{I}, \boldsymbol{W}^\intercal \boldsymbol{W} = \boldsymbol{I}$. Without fixing P, Q, R, the decomposition is non-unique. An example decomposition is illustrated in Fig. 1.2. Several equivalent forms are summarised below:

- $\boldsymbol{\mathcal{X}} = \sum_{l=1}^{I} \sum_{m=1}^{J} \sum_{n=1}^{K} \mathcal{G}_{lmn} \boldsymbol{u}_l \circ \boldsymbol{v}_m \circ \boldsymbol{w}_n$;
- $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U} \times_2 \boldsymbol{V} \times_3 \boldsymbol{W}$;

- $\text{vec}(\boldsymbol{\mathcal{X}}) = (\boldsymbol{U} \otimes \boldsymbol{V} \otimes \boldsymbol{W})\text{vec}(\boldsymbol{\mathcal{G}})$;
- $\boldsymbol{X}_{(1)} = \boldsymbol{U}\boldsymbol{G}_{(1)}(\boldsymbol{V} \otimes \boldsymbol{W})^{\intercal}$.

Note that each column of $\boldsymbol{U}$ interacts with every column of $\boldsymbol{V}$ and every column of $\boldsymbol{W}$ in this decomposition, and the strength of this interaction is encoded in the corresponding element of $\boldsymbol{\mathcal{G}}$. This is different from the CP decomposition, which only allows interactions between corresponding columns of $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$. A trivial decomposition is given by $\boldsymbol{U} = \boldsymbol{I} \in \mathbb{R}^{I \times I}, \boldsymbol{V} = \boldsymbol{I} \in \mathbb{R}^{J \times J}, \boldsymbol{W} = \boldsymbol{I} \in \mathbb{R}^{K \times K}$ and $\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}}$.

The Tucker rank of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$ is the tuple (k_1, k_2, k_3) with elementwise minimal entries $k_\mu \in \mathbb{N}$ such that there exist columnwise orthonormal matrices $\boldsymbol{U}_\mu \in \mathbb{R}^{n_\mu \times k_\mu}$ and a core tensor $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{k_1 \times k_2 \times k_3}$ with

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \times_3 \boldsymbol{U}_3.$$

This representation is known as the orthogonal Tucker format and the matrices are called mode frames for the Tucker decomposition.

Let R_n be the mode-n rank of $\boldsymbol{\mathcal{X}}$ as before, we can choose $\boldsymbol{U}$ to be an $I \times R_1$ orthonormal basis of the row space of $\boldsymbol{X}_{(1)}$ as $\boldsymbol{U}_{R_1}$ and similarly for $\boldsymbol{V}$ and $\boldsymbol{W}$. Then, we have

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U}_{R_1} \times_2 \boldsymbol{V}_{R_2} \times_3 \boldsymbol{W}_{R_2},$$

where $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$. Such a model is known as the multilinear SVD (MLSVD) or higher-order SVD (HOSVD). We have the following connections with the matrix SVD:

- The slabs of $\boldsymbol{\mathcal{G}}$ along each mode are orthogonal to each other, i.e., $\text{vec}(\boldsymbol{\mathcal{G}}(l, :, :))^{\intercal}\text{vec}(\boldsymbol{\mathcal{G}}(l', :, :)) = 0$ for $l' \neq l$;
- $\|\boldsymbol{\mathcal{G}}(l, :, :)\|_F$ equals the l-th singular value of $\boldsymbol{X}_{(l)}$.

Given fixed general orthonormal $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$, the core tensor is uniquely defined. And it follows that $\|\boldsymbol{\mathcal{X}}\|_F^2 = \|\boldsymbol{\mathcal{G}}\|_F^2$. Besides, if some of the outer products are dropped to form tensor $\tilde{\boldsymbol{\mathcal{X}}}$, we have $\|\boldsymbol{\mathcal{X}} - \tilde{\boldsymbol{\mathcal{X}}}\|_F^2 = \sum_{l,m,n \in \mathcal{D}} |\mathcal{G}_{lmn}|^2$, where $\mathcal{D}$ is the set of dropped core element indices.

In practice, we are interested in a representation where the core tensor has modes e, f, h that best approximates $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I\times J\times K}$. Formally,

$$\begin{aligned}
&\min_{\boldsymbol{\mathcal{G}},\boldsymbol{U},\boldsymbol{V},\boldsymbol{W}} \|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U} \times_2 \boldsymbol{V} \times_3 \boldsymbol{W}\|_F^2 \\
&\text{s.t. } \boldsymbol{\mathcal{G}} \in \mathbb{R}^{e\times f\times h}, \\
&\quad \boldsymbol{U} \in \mathbb{R}^{I\times e}, \boldsymbol{U}^\intercal \boldsymbol{U} = \boldsymbol{I}, \\
&\quad \boldsymbol{V} \in \mathbb{R}^{J\times f}, \boldsymbol{V}^\intercal \boldsymbol{V} = \boldsymbol{I}, \\
&\quad \boldsymbol{W} \in \mathbb{R}^{K\times h}, \boldsymbol{W}^\intercal \boldsymbol{W} = \boldsymbol{I}.
\end{aligned}$$

Such a solution $(\boldsymbol{U}^*, \boldsymbol{V}^*, \boldsymbol{W}^*, \boldsymbol{\mathcal{G}}^*)$ always exists but it is an NP-hard problem. We can compact the energy $|\mathcal{G}_{lmn}|^2$ of the core tensor in its upper-left-front corner by permuting the frontal, lateral slabs, etc. of $\boldsymbol{\mathcal{G}}$ and counter-permute the corresponding columns of $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$. We can obtain an approximation by truncating the core keeping only the upper-left-front $e \times f \times h$ part. The resulting error is bounded as

$$\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}'\|_F^2 \leq \sum_{l=e+1}^{I} \|\boldsymbol{\mathcal{G}}(l,:,:)\|_F^2 + \sum_{m=f+1}^{J} \|\boldsymbol{\mathcal{G}}(:,m,:)\|_F^2 + \sum_{n=h+1}^{K} \|\boldsymbol{\mathcal{G}}(:,:,n)\|_F^2.$$

However, this does not give the best approximation $\boldsymbol{\mathcal{G}}^*$. Alternatively, we can attempt a Tucker truncation. Considering a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1\times n_2\times n_3}$, let $\boldsymbol{X}_{(\mu)} = \boldsymbol{U}_\mu \boldsymbol{\Sigma}_\mu \boldsymbol{V}_\mu^\intercal$, where $\boldsymbol{U}_\mu \in \mathbb{R}^{n_\mu \times n_\mu}$, be an SVD with singular values $\{\sigma_{\mu,1}, \cdots, \sigma_{\mu,n_\mu}\}$. The Tucker truncation of $\boldsymbol{\mathcal{X}}$ to a tensor $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{e\times f\times h}$ is defined by

$$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{X}} \times_1 (\bar{\boldsymbol{U}}_1 \bar{\boldsymbol{U}}_1^\intercal) \times_2 (\bar{\boldsymbol{U}}_2 \bar{\boldsymbol{U}}_2^\intercal) \times_3 (\bar{\boldsymbol{U}}_3 \bar{\boldsymbol{U}}_3^\intercal),$$

where $\bar{\boldsymbol{U}}_\mu$ is the matrix of the first e, f or h columns of $\boldsymbol{U}_\mu$. The error in this case is bounded by

$$\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}''\|_F^2 \leq \sum_{\mu=1}^{3} \sum_{i=e,h,f+1}^{n_\mu} \sigma_{\mu,i}^2 \leq 3\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}^*\|.$$

So this is not the best estimate either. The Eckart-Young theorem does not apply to higher-order tensors. However, the above procedures provide a good warm start. In fact, we can prove the relationship $\boldsymbol{\mathcal{G}}_{(1)}^* = \boldsymbol{U}^{*T} \boldsymbol{X}_{(1)} (\boldsymbol{V}^* \otimes \boldsymbol{W}^*)$. Substituting for $\boldsymbol{\mathcal{G}}$, we can recast the original problem as

$$\max_{\boldsymbol{U},\boldsymbol{V},\boldsymbol{W}} \|\boldsymbol{U}^\intercal \boldsymbol{X}_{(1)} (\boldsymbol{V} \otimes \boldsymbol{W})\|_F^2,$$

with the same conditions on $\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}$ as before. Furthemore, $\boldsymbol{G}_{(1)}^*$ has orthogonal columns and $\boldsymbol{U}^*, \boldsymbol{V}^*, \boldsymbol{W}^*$ are the first e, f or h columns of the left subspaces of $\boldsymbol{X}_{(1)}(\boldsymbol{V}^* \otimes \boldsymbol{W}^*), \boldsymbol{X}_{(2)}(\boldsymbol{U}^* \otimes \boldsymbol{W}^*)$ and $\boldsymbol{X}_{(3)}(\boldsymbol{U}^* \otimes \boldsymbol{V}^*)$ respectively. Finaly, $\{\|\boldsymbol{G}_{(1)}^*(:,m)\|_2^2\}_{m=1}^e$ are the e principal singular values of $\boldsymbol{X}_{(1)}(\boldsymbol{V}^* \otimes \boldsymbol{W}^*)$.

1.11 Rank Sparsity Tensor Decomposition

In many real-world applications, input data are naturally represented by tensors (i.e., multi-dimensional arrays). Traditionally, such data would require vectorising before processing and thus destroy the inherent higher-order interactions. As a result, novel models must be developed to preserve the multilinear structure when extracting the hidden and evolving trends in such data. Typical tensor data are video clips, color images, multi-channel EEG records, etc.

In practice, important information usually lie in a (multi-linear) low-dimensional space whose dimensionality is much lower dimensional space than observations.This is the essence of low-rank modelling. In this chapter, we focus on the problem of recovering a low-dimensional multilinear structure from tensor data corrupted by gross corruptions.

Given a tensor $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, its tensor rank [61] is defined by the smallest r such that $\boldsymbol{\mathcal{L}} = \sum_{i=1}^{r} \boldsymbol{a}_i^{(1)} \circ \cdots \circ \boldsymbol{a}_i^{(N)}$, where $\circ$ denotes outer products among some Nr vectors $\boldsymbol{a}_i^{(1)}, \cdots, \boldsymbol{a}_i^{(N)}, 1 \leq i \leq r$. As such, robust low-rank tensor modelling seeks a decomposition $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}}$ for an N^{th}-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, where $\boldsymbol{\mathcal{L}}$ has a low tensor rank and $\boldsymbol{\mathcal{S}}$ is sparse. However, the tensor rank is usually intractable [49]. A common adjustment [74, 95, 137] is to use a convex combination of the n-ranks of $\boldsymbol{\mathcal{L}}$, that is $\gamma = \sum_{i=1}^{N} \alpha_i \text{rank}_i(\boldsymbol{\mathcal{L}})$, where $\alpha_i \geq 0$, $\sum_{i=1}^{N} \alpha_i = 1$ and $\text{rank}_i(\boldsymbol{\mathcal{L}})$ is the the column rank of the mode-i matricisation [60] of $\boldsymbol{\mathcal{L}}$. It is, therefore, natural to obtain the decomposition by solving optimisation problem (1.14)

$$\min_{\boldsymbol{\mathcal{L}},\boldsymbol{\mathcal{S}}} \gamma + \lambda \|\boldsymbol{\mathcal{E}}\|_0 \quad \text{s.t. } \gamma = \sum_{i=1}^{N} \alpha_i \text{rank}_i(\boldsymbol{\mathcal{L}}),\ \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}}, \tag{1.14}$$

where $\|\boldsymbol{\mathcal{S}}\|_0$ is the l_0 norm of the vectorisation of $\boldsymbol{\mathcal{S}}$ and λ is a weighting parameter.

RSTD [65] is a direct multi-linear extension of matrix principal component pursuit (PCP) [21]. It approximates (1.14) by replacing $\text{rank}_i(\boldsymbol{\mathcal{L}})$ and $\|\boldsymbol{\mathcal{S}}\|_0$ with convex surrogates $\|\boldsymbol{L}_{(i)}\|_*$ and $\|\boldsymbol{\mathcal{S}}\|_1$ respectively, where $\|\boldsymbol{L}_{(i)}\|_*$ is the nuclear norm of the mode-i matricisation of $\boldsymbol{\mathcal{L}}$ and $\|\boldsymbol{\mathcal{S}}\|_1$ is the l_1 norm of the vectorisation of $\boldsymbol{\mathcal{S}}$. As a result, it solves the following alternative objective

$$\min_{\boldsymbol{\mathcal{L}},\boldsymbol{\mathcal{S}}} \sum_{i=1}^{N} \alpha_i \|\boldsymbol{L}_{(i)}\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \quad \text{s.t. } \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}}. \tag{1.15}$$

An ALM solver can be found in [46]. It is also worth noting that under certain conditions RSTD is guaranteed to exactly recover the low-rank component [52].

1.12 Applications of Robust Principal Component Analysis

1.12.1 Image Classification

RPCA can be used to denoise images. Once clean, classification can be performed on those images. The classification results directly reflect the image denoising ability. For a set of correlated images, low-rank algorithms are normally used to remove noise that is sparse. The same classifier is thus able to compare the different low-rank models.

1.12.2 Face Denoising

It is common practice to decompose raw facial images as a low-rank component for faithful face representation and a sparse component for defects. This is because the face is a convex Lambertian surface which under distant and isotropic lighting has an underlying model that spans a 9-D linear subspace [7], but theoretical lighting conditions cannot be realised and there are unavoidable occlusion and albedo variations in real images.

1.12.3 Background Subtraction

In automated video analytics, object detection is instrumental in object tracking, activity recognition and behaviour understanding. Practical applications include surveillance, traffic control, robotic operation, etc, where foreground objects can be people, vehicles, products and so forth. Background subtraction segments moving objects by calculating the pixel-wise difference between each video frame and the background. For a static camera, the background is almost static, while the foreground objects are mostly moving. Consequently, a decomposition into a low-rank component for the background and a sparse component for foreground objects is a valid model for such dynamics. Indeed, if the only change in the background is illumination, then the matrix representation of vectorised backgrounds has a rank of 1. It has been demonstrated that PCP is quite effective for such a low-rank matrix analysis problem [21].

1.12.4 Face and Facial Expression Recognition

Recent research has established that an expressive face can be treated as a neutral face plus a sparse expression component [117], which is identity-independent due to its constituent local non-rigid motions, i.e. action units. This is central to computer vision as it enables human emotion classification from such visual cues. The images of the neutral face form a low-rank model whereas the facial expressions can be treated as sparse noise.

1.12.5 UV Map Completion

As faces form a convex Lambertian surface so do their UV textures. RPCA with missing values can be applied to incomplete UV maps lifted from video frames that are due to self-occlusion of the faces and also regional fitting errors of an imperfect 3D model such as the 3DMM. Once completed, this UV map, combined with the corresponding 3D face, is extremely useful, as it can be used to synthesise 2D faces of arbitrary poses. That way, we can probe image pairs of similar poses to improve recognition performance.

CHAPTER 2

Informed Non-Convex Robust Principal Component Analysis with Features

2.1 Introduction

Recent advances in non-convex optimization algorithms continue to undermine their convex counterparts [43,47,59]. In particular, non-convex RPCA algorithms such as fast RPCA [138] and AltProj [81] exhibit better properties than the convex formulation. Most recently, [83] embedded features into a non-convex RPCA framework known as IRPCA-IHT with faster speed. However, it remains unclear, how to exploit side information in non-convex RPCA and whether it facilitates provably correct, fast, and more accurate algorithms.

In this chapter, we give positive answers to the above questions by proposing a novel, non-convex scheme that fully leverages side information (features) regarding row and column subspaces of the low-rank matrix. Even though the proposed algorithm is inspired by the recently proposed fast RPCA [138], our contributions are by no means trivial, especially from a theoretical perspective. First, fast RPCA cannot be easily extended to consistently take account of features. Second, as we show in this work, incoherence assumptions on the observation matrix and features play a decisive role in determining the corruption bound and the computational complexity of the non-convex algorithm. Third, fast RPCA is limited to a corruption rate of 50% due to their choice of the hard threshold, whereas our algorithm ups this rate to 90%. Fourth, we prove that the costly projection onto factorized spaces is entirely optional when features satisfy certain incoherence conditions. Although our algorithm maintains the same corruption rate of $O(\frac{n}{r^{1.5}})$ and complexity of $O(rn^2 \log(\frac{1}{\epsilon}))$ as fast RPCA, we show empirically that massive gains in accuracy and speed can still be obtained. Besides, the transfer of coherence dependency from observation to features means that our algorithm is capable of dealing with highly incoherent data.

Unavoidably, features adversely affect tolerance to corruption in IRPCA-IHT ($O(\frac{n}{d})$) compared to its predecessor AltProj ($O(\frac{n}{r})$). This is not always true with our algorithm in relation to fast RPCA. And when the underlying rank is low but features are only weakly informative, *i.e.* $r \ll d$, which is often the case, our tolerance to corruption is arguably better. IRPCA-IHT also has a higher complexity of $O((dn^2 + d^2 r)\log(\frac{1}{\epsilon}))$ than that of our algorithm. Although feature-free convex and non-convex algorithms have higher asymptotic error bounds than our algorithm, we show in our experiments that this does not translate as accuracy in reality. Our algorithm still has the best performance in recovering accurately the low-rank part from highly corrupted matrices. This may be attributed to the fact that our bounds are not tight. Besides, PCPF and AltProj have much higher complexity ($O(\frac{n^3}{\sqrt{\epsilon}})$ and $O(r^2 n^2 \log(\frac{1}{\epsilon}))$) than ours. For PCPF, there does not exist any theoretical analysis under the deterministic sparsity model. Nonetheless, we show in our experiments that our algorithm is superior with regard to both recoverability and running time.

The contributions of this work are summarised as follows:

- A novel non-convex algorithm integrating features with informed sparsity is proposed in order to solve the RPCA problem.

- We establish theoretical guarantees of exact recovery under different assumptions regarding the incoherence of features and observation.

- Extensive experimental results on synthetic data indicate that the proposed algorithm is faster and more accurate in low-rank matrix recovery than the compared state-of-the-art convex and non-convex methods for RPCA (with and without features).

- Experiments on two real-world datasets, namely MNIST and Yale B database demonstrate the practical merits of the proposed algorithm.

2.2 Related Work

AltProj [81] addresses RPCA by minimizing an entirely different objective:

$$\begin{aligned} \min_{\boldsymbol{L},\boldsymbol{S}} \quad & \|\boldsymbol{M}-\boldsymbol{L}-\boldsymbol{S}\|_F \\ \text{subject to} \quad & \boldsymbol{L} \in \text{ set of low-rank matrices} \\ & \boldsymbol{S} \in \text{ set of sparse matrices,} \end{aligned} \tag{2.1}$$

where the search consists of alternating non-convex projections. That is, during each cycle, hard-thresholding takes place first to remove large entries and projection of appropriate residuals onto the set of low-rank matrices with increasing ranks is carried out next. Exact recovery has also been established.

Fast RPCA [138] follows yet another non-convex approach to solve RPCA. After an initialization stage, fast RPCA updates bilinear factors $\boldsymbol{U}$, $\boldsymbol{V}$ such that $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{V}^\intercal$ through a series of projected gradient descent and sparse estimations, where $\boldsymbol{U}$, $\boldsymbol{V}$ minimize the following loss:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \frac{1}{2}\|\boldsymbol{U}\boldsymbol{V}^\intercal+\boldsymbol{S}-\boldsymbol{M}\|_F^2+\frac{1}{8}\|\boldsymbol{U}^\intercal\boldsymbol{U}-\boldsymbol{V}^\intercal\boldsymbol{V}\|_F^2, \tag{2.2}$$

for $\boldsymbol{U}$, $\boldsymbol{V}$ properly constrained. Recovery guarantee is ensured.

We also mention here several works of non-convex objectives [85, 108], though exact recovery guarantees are lacking.

2.3 Problem Setup

Suppose that there is a known data matrix $\boldsymbol{M} \in \mathbb{R}^{n_1 \times n_2}$, which can be decomposed into a low-rank component $\boldsymbol{L}^*$ and a sparse error matrix $\boldsymbol{S}^*$ of compatible dimensions. Our aim is to identify these underlying matrices and hence robustly recover the low-rank component with the help of available side information in the form of feature matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$.

Concretely, let $\boldsymbol{L}^* = \boldsymbol{U}^*\boldsymbol{\Sigma}^*\boldsymbol{V}^{*T}$be the singular value decomposition and $\boldsymbol{P}^* = \boldsymbol{X}^\intercal\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}$ and $\boldsymbol{Q}^* = \boldsymbol{Y}^\intercal\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}$. $\boldsymbol{S}^*$ follows the random sparsity model. That is, the support of $\boldsymbol{S}^*$ is chosen uniformly at random from the collection of all support sets of the same size. Furthermore, let us be informed of the proportion of non-zero entries per row and column, denoted by α. Assume that there are also available features $\boldsymbol{X} \in \mathbb{R}^{n_1 \times d_1}$ and $\boldsymbol{Y} \in \mathbb{R}^{n_2 \times d_2}$ such that they are feasible, i.e. col($\boldsymbol{X}$)$\supseteq$col($\boldsymbol{U}^*$) and col($\boldsymbol{Y}$)$\supseteq$col($\boldsymbol{V}^*$) where col($\boldsymbol{A}$) is the column space of $\boldsymbol{A}$ and $\boldsymbol{X}^\intercal\boldsymbol{X} = \boldsymbol{Y}^\intercal\boldsymbol{Y} = \boldsymbol{I}$[1].

[1] This can always achieved via orthogonalisation.

In this work, we discuss robust low-rank recovery using the above mentioned features and three different incoherence conditions: (i) $\|\boldsymbol{U}^*\|_{2,\infty} \leq \sqrt{\frac{\mu_1 r}{n_1}}$ and $\|\boldsymbol{V}^*\|_{2,\infty} \leq \sqrt{\frac{\mu_1 r}{n_2}}$; (ii) $\|\boldsymbol{X}\|_{2,\infty} \leq \sqrt{\frac{\mu_2 d_1}{n_1}}$ and $\|\boldsymbol{Y}\|_{2,\infty} \leq \sqrt{\frac{\mu_2 d_2}{n_2}}$; (iii) both (i) and (ii), where r is the given rank of $\boldsymbol{L}^*$ and μ_1, μ_2 are constants.

2.4 Algorithm

We use a non-convex approach to achieve the above objective. The algorithm consists of an initialization phase followed by a gradient descent phase. At each stage, we keep track of the factors $\boldsymbol{P}$, $\boldsymbol{Q}$ such that $\boldsymbol{L} = \boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal$.

2.4.1 Hard-thresholding

We first introduce the sparse estimator via hard-thresholding which is used in both phases. Given a threshold θ, $\mathcal{T}_\theta(\boldsymbol{A})$ removes elements of $\boldsymbol{A}$ that are not among the largest θ-fraction of elements in their respective rows and columns, breaking ties arbitrarily for equal elements:

$$\mathcal{T}_\theta(\boldsymbol{A})_{ij} = \begin{cases} 0 & \text{if } |\boldsymbol{A}_{ij}| \leq \boldsymbol{A}^\theta_{i\cdot} \text{ or } |\boldsymbol{A}_{ij}| \leq \boldsymbol{A}^\theta_{\cdot j}, \\ \boldsymbol{A}_{ij} & \text{otherwise,} \end{cases} \tag{2.3}$$

where $\boldsymbol{A}^\theta_{i\cdot}, \boldsymbol{A}^\theta_{\cdot j}$ are the $(n_2\theta)^{\text{th}}$ and $(n_1\theta)^{\text{th}}$ largest element in absolute value in row i and column j respectively.

2.4.2 Initialization

$\boldsymbol{S}$ is first initialized as $\boldsymbol{S}_0 = \mathcal{T}_\alpha(\boldsymbol{M})$. Next, we obtain $\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal$ as the r-truncated SVD of $\boldsymbol{L}_0$, which is calculated via $\boldsymbol{L}_0 = \boldsymbol{M} - \boldsymbol{S}_0$. We can then construct $\boldsymbol{P}_0 = \boldsymbol{X}^\intercal\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}$ and $\boldsymbol{Q}_0 = \boldsymbol{Y}^\intercal\boldsymbol{V}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}$. Such an initialization scheme gives $\boldsymbol{P}$, $\boldsymbol{Q}$ the desirable properties for use in the second phase.

2.4.3 Gradient Descent

In case (i), we need the following sets:

$$\mathcal{P} = \{\boldsymbol{A} \in \mathbb{R}^{d_1 \times r} | \|\boldsymbol{X}\boldsymbol{A}\|_{2,\infty} \leq \sqrt{\frac{2\mu_1 r}{n_1}} \|\boldsymbol{P}_0\|_2\}, \tag{2.4}$$

$$\mathcal{Q} = \{\boldsymbol{A} \in \mathbb{R}^{d_2 \times r} | \|\boldsymbol{Y}\boldsymbol{A}\|_{2,\infty} \leq \sqrt{\frac{2\mu_1 r}{n_2}} \|\boldsymbol{Q}_0\|_2\}. \tag{2.5}$$

Otherwise, we can simply take $\mathcal{P}$ as $\mathbb{R}^{d_1 \times r}$ and $\mathcal{Q}$ as $\mathbb{R}^{d_2 \times r}$.

Algorithm 4 Non-convex solver for robust principal component analysis with features

```
Input: Observation $\boldsymbol{M}$, features $\boldsymbol{X}, \boldsymbol{Y}$, rank $r$, corruption approximation $\alpha$ and step size $\eta$.
    Initialization:
    $\boldsymbol{S} = \mathcal{T}_{\alpha}(\boldsymbol{M})$
    $\boldsymbol{U\Sigma V}^{\intercal} = r\text{-SVD}(\boldsymbol{M} - \boldsymbol{S})$
    $\boldsymbol{P} = \boldsymbol{X}^{\intercal}\boldsymbol{U\Sigma}^{\frac{1}{2}}$
    $\boldsymbol{Q} = \boldsymbol{Y}^{\intercal}\boldsymbol{V\Sigma}^{\frac{1}{2}}$
    Gradient descent:
    $\boldsymbol{P} = \boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P})$
    $\boldsymbol{Q} = \boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q})$
    while not converged do
        $\boldsymbol{S} = \mathcal{T}_{\alpha+\min(10\alpha,0.1)}(\boldsymbol{M} - \boldsymbol{XPQ}^{\intercal}\boldsymbol{Y}^{\intercal})$
        $\boldsymbol{P} = \boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P} - \eta\nabla_{\boldsymbol{P}}\mathcal{L})$
        $\boldsymbol{Q} = \boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q} - \eta\nabla_{\boldsymbol{Q}}\mathcal{L})$
    end while
Return: $\boldsymbol{L} = \boldsymbol{XPQ}^{\intercal}\boldsymbol{Y}^{\intercal}, \boldsymbol{S}$
```

To proceed, we first regularise $\boldsymbol{P}_0$ and $\boldsymbol{Q}_0$:

$$\boldsymbol{P} = \boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P}_0),\ \boldsymbol{Q} = \boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q}_0). \tag{2.6}$$

At each iteration, we first update $\boldsymbol{S}$ with the sparse estimator using a threshold of $\alpha+\min(10\alpha, 0.1)$:

$$\boldsymbol{S} = \mathcal{T}_{\alpha+\min(10\alpha,0.1)}(\boldsymbol{M} - \boldsymbol{XPQ}^{\intercal}\boldsymbol{Y}^{\intercal}). \tag{2.7}$$

For $\boldsymbol{P}$, $\boldsymbol{Q}$, we define the following objective function

$$\mathcal{L}(\boldsymbol{P}, \boldsymbol{Q}) = \frac{1}{2}\|\boldsymbol{XPQ}^{\intercal}\boldsymbol{Y}^{\intercal} + \boldsymbol{S} - \boldsymbol{M}\|_F^2 + \frac{1}{64}\|\boldsymbol{P}^{\intercal}\boldsymbol{P} - \boldsymbol{Q}^{\intercal}\boldsymbol{Q}\|_F^2. \tag{2.8}$$

$\boldsymbol{P}$ and $\boldsymbol{Q}$ are updated by minimizing the above function subject to the constraints imposed by the sets $\mathcal{P}$ and $\mathcal{Q}$. That is,

$$\boldsymbol{P} = \boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P} - \eta\nabla_{\boldsymbol{P}}\mathcal{L}), \tag{2.9}$$

$$\boldsymbol{Q} = \boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q} - \eta\nabla_{\boldsymbol{Q}}\mathcal{L}), \tag{2.10}$$

where the step size η is determined analytically below. With properly initialized $\boldsymbol{P}$ and $\boldsymbol{Q}$, such an optimization design converges to $\boldsymbol{P}^*$ and $\boldsymbol{Q}^*$. The procedure is summarized in Algorithm 4.

2.5 Analysis

We first provide theoretical justification of our proposed approach. Then we evaluate its computational complexity. The proofs can be found in the appendix.

The Euclidean metric is not applicable here because of the non-uniqueness of the bi-factorisation $\boldsymbol{L}^* = \boldsymbol{A}^*\boldsymbol{B}^{*T}$, which corresponds to a manifold rather than a point. Hence, we define the following distance between $(\boldsymbol{A}, \boldsymbol{B})$ and any of the optimal pair $(\boldsymbol{A}^*, \boldsymbol{B}^*)$ such that $\boldsymbol{L}^* = \boldsymbol{A}^*\boldsymbol{B}^{*T}$:

$$d(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{A}^*, \boldsymbol{B}^*) = \min_{\boldsymbol{R}} \sqrt{\|\boldsymbol{A} - \boldsymbol{A}^*\boldsymbol{R}\|_F^2 + \|\boldsymbol{B} - \boldsymbol{B}^*\boldsymbol{R}\|_F^2}, \tag{2.11}$$

where $\boldsymbol{R}$ is an $r \times r$ orthogonal matrix.

2.5.1 Convergence

The initialization phase provides us with the following guarantees on $\boldsymbol{P}$ and $\boldsymbol{Q}$.

Theorem 2. *In cases (i) and (iii), if $\alpha \leq \frac{1}{16\kappa r\mu_1}$, we have*

$$d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\alpha r\mu_1\sqrt{r\kappa\sigma_1^*}. \tag{2.12}$$

In case (ii), if $\alpha \leq \frac{1}{16\kappa\mu_2\sqrt{d_1 d_2}}$, we have

$$d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\alpha\mu_2\sqrt{r d_1 d_2 \kappa\sigma_1^*}, \tag{2.13}$$

where κ is the condition number of L^* and d is a distance metric defined in the appendix.

Theorem 3. *For $\eta \leq \frac{1}{192\|\boldsymbol{L}_0\|_2}$, there exist constants $c_1 > 0$, $c_2 > 0$, $c_3 > 0$, $c_4 > 0$, $c_5 > 0$ and $c_6 > 0$ such that, in case (i), when $\alpha \leq \frac{c_1}{\mu_1(\kappa r)^{\frac{3}{2}}}$, we have the following relationship*

$$d(\boldsymbol{P}_t, \boldsymbol{Q}_t, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2 \leq (1 - c_2\eta\sigma_r^*)^t d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2, \tag{2.14}$$

in case (ii), when $\alpha \leq \frac{c_3}{\mu_2 d r^{\frac{1}{2}} \kappa^{\frac{3}{2}}}$, we have

$$d(\boldsymbol{P}_t, \boldsymbol{Q}_t, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2 \leq (1 - c_4\eta\sigma_r^*)^t d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2. \tag{2.15}$$

and in case (iii), when $\alpha \leq c_5 \min(\frac{1}{\mu_2 d\kappa}, \frac{1}{\mu_1(\kappa r)^{\frac{3}{2}}})$, we have

$$d(\boldsymbol{P}_t, \boldsymbol{Q}_t, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2 \leq (1 - c_6\eta\sigma_r^*)^t d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2. \tag{2.16}$$

2.5.2 Complexity

From **Theorem 2**, it follows that our algorithm converges at a linear rate under assumptions (ii) and (iii). To converge below ϵ of the initial error, $O(\log(\frac{1}{\epsilon}))$ iterations are needed. At each iteration, the most costly step is matrix multiplication which takes $O(rn^2)$ time. Overall, our algorithm has total running time of $O(rn^2\log(\frac{1}{\epsilon}))$.

2.6 Experiments

We have found that when the step size is set to 0.5, reasonable results can be obtained. For all algorithms in comparison, we run a total of 3000 iterations or until $\|\boldsymbol{M} - \boldsymbol{L} - \boldsymbol{S}\|_F / \|\boldsymbol{M}\|_F < 10^{-7}$ is met.

2.6.1 Phase transition

Here, we vary the rank and the error sparsity to investigate the behavior of both our algorithm and existing state-of-art algorithms in terms of recoverability. True low-rank matrices are created via $\boldsymbol{L}^* = \boldsymbol{J}\boldsymbol{K}^\intercal$, where $200 \times r$ matrices $\boldsymbol{J}, \boldsymbol{K}$ have independent elements drawn randomly from a Gaussian distribution of mean 0 and variance $5 \cdot 10^{-3}$ so r becomes the rank of $\boldsymbol{L}^*$. Next, we corrupt each column of $\boldsymbol{L}^*$ such that α of the elements are set independently with magnitude $\mathcal{U}(0, \frac{r}{40})$. However, this does not guarantee α row corruption. We thus select only matrices whose maximum row corruption does not exceed $\alpha + 6.5\%$ but we still feed α to the algorithms in order to demonstrate that our algorithm does not need the exact value of corruption ratio. We consider two types of signs for error: Bernoulli ± 1 and sgn$(\boldsymbol{L}^*)$. The resulting $\boldsymbol{M}$ thus becomes the simulated observation. In addition, let $\boldsymbol{L}^* = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\intercal$ be the SVD of $\boldsymbol{L}^*$. Feature $\boldsymbol{X}$ is formed by randomly interweaving column vectors of $\boldsymbol{U}$ with 5 arbitrary orthonormal bases for the null space of $\boldsymbol{U}^\intercal$, while permuting the expanded columns of $\boldsymbol{V}$ with 5 random orthonormal bases for the kernel of $\boldsymbol{V}^\intercal$ forms feature $\boldsymbol{Y}$. Hence, the feasibility conditions are fulfilled: col$(\boldsymbol{X})$ $\supseteq$col$(\boldsymbol{L}_0)$, col$(\boldsymbol{Y})$ $\supseteq$col$(\boldsymbol{L}_0^\intercal)$. For each (r, α) pair, three observations are constructed. The recovery is successful if for all these three problems, $\frac{\|\boldsymbol{L} - \boldsymbol{L}^*\|_F}{\|\boldsymbol{L}^*\|_F} < 10^{-3}$ from the recovered $\boldsymbol{L}$.

Figures 2.1(I) plot results from algorithms incorporating features. Besides, our algorithm contrasts with fast RPCA in Figure 2.1(II). Other feature-free algorithms are investigated in Figure 2.1(III). Figures 2.1(a) illustrate the random sign model and Figures 2.1(b) for the coherent sign model. All previous non-convex attempts fail to outperform their convex equivalents. IRPCA-IHT is unable to deal with even moderate levels of corruption. The frontier of recoverability that has been advanced by our algorithm over PCPF is phenomenal, massively ameliorating fast RPCA. The anomalous asymmetry in the two sign models is no longer observed in non-convex algorithms.

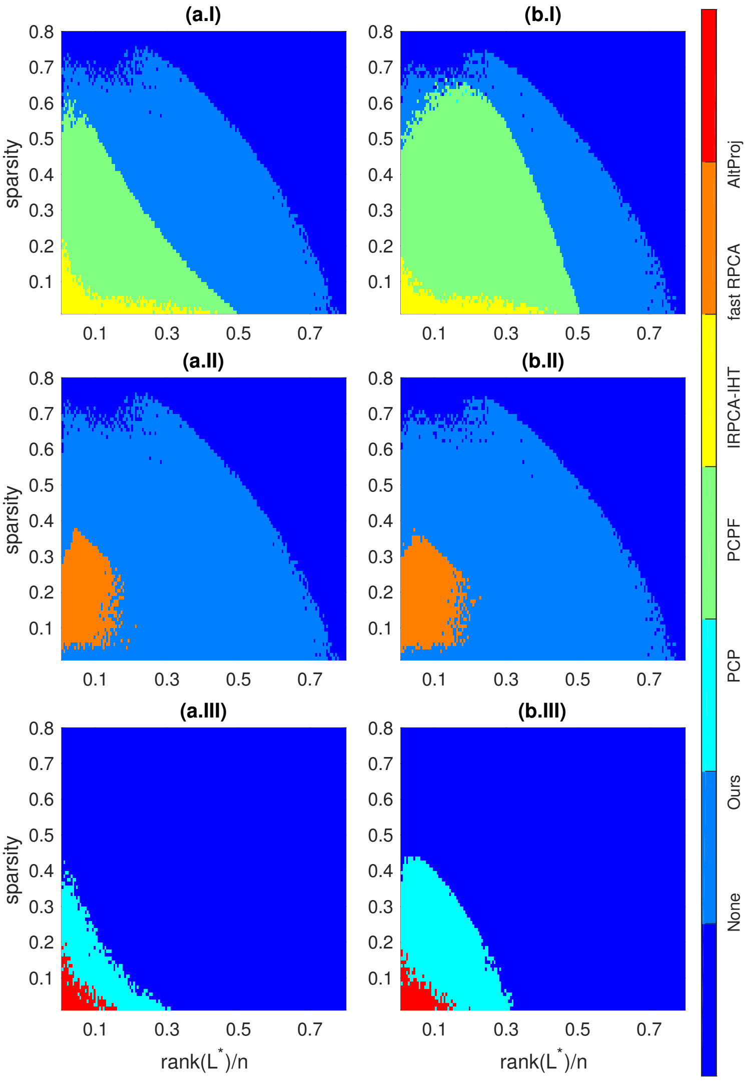


Figure 2.1: Domains of recovery by various algorithms: (a) for random signs and (b) for coherent signs.

2.6.2 Running Time

Next, we highlight the speed of our algorithm for large-scale matrices, typical of video sequences [132]. 1500×1500 to 2500×2500 random observation matrices are generated, where the rank is chosen to be 20% of the column number and random sign error corrupts 11% of the entries, with features $\boldsymbol{X}, \boldsymbol{Y}$ having a dimension of 50% of the column number. The running times of all algorithms except IRPCA-IHT are plotted in 2.2 (i) because IRPCA-IHT is not able to achieve a relative error ($\frac{\|\boldsymbol{L}-\boldsymbol{L}^*\|_F}{\|\boldsymbol{L}^*\|_F}$) less than 1% for larger matrices. For fair comparison, we have relaxed the rank to 0.3% of the column number and error rate to 0.1% to compare our algorithm with IRPCA-IHT for matrices ranging from 2000×2000 to 10000×10000. We have used features $\boldsymbol{X}, \boldsymbol{Y}$ having a dimension of 80% of the column number to speed up the process. The result is shown in Figure 2.2 (ii). All times are averaged over three trials. It is evident that, for large matrices, our algorithm overtakes all existing algorithms in terms of speed. Note that features in PCPF even slow down the recovery process.

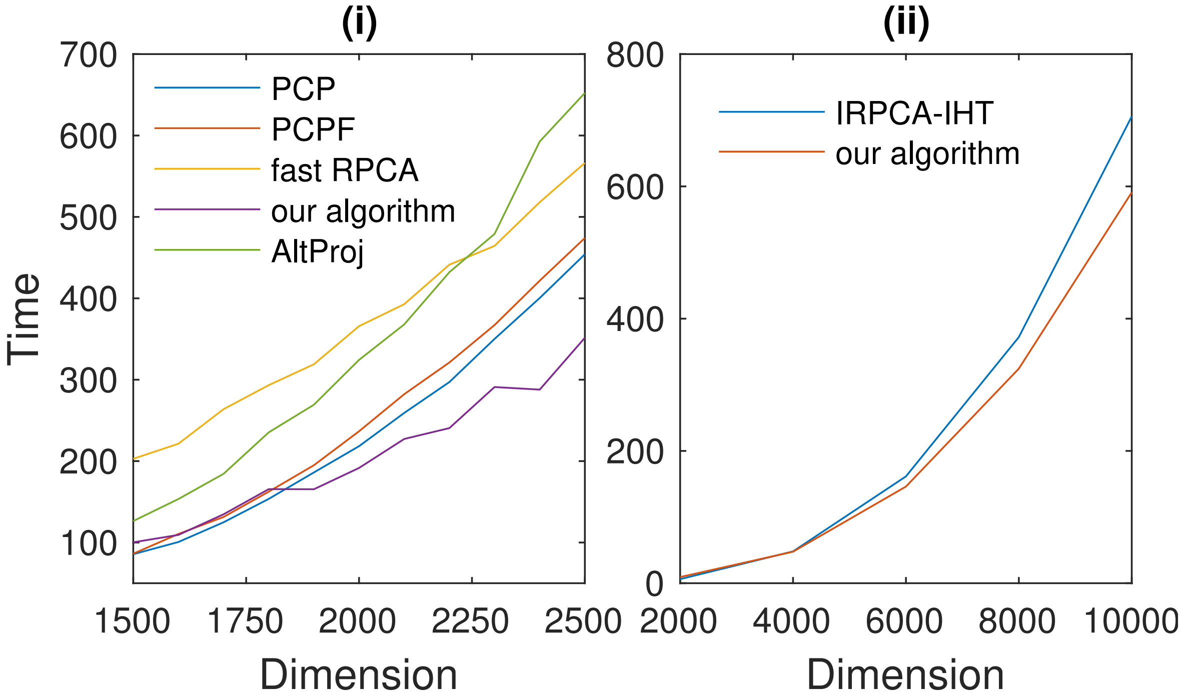


Figure 2.2: (i) Running times for observation matrices of increasing dimensions for (i) PCP, PCPF, fast RPCA, AltProj, our algorithm and (ii) IRPCA-IHT and our algorithm when $\frac{\|\boldsymbol{L}-\boldsymbol{L}^*\|_F}{\|\boldsymbol{L}^*\|_F} \leq 1\%$.

Table 2.1: Classification results obtained by a linear SVM.

α	clean	noisy	PCP	PCPF	AltProj	IRPCA-IHT	fast RPCA	our algorithm
10	89.65	30.45	82.75	83.35	81.4	65.2	81.1	**86.9**
15		25.1	82.95	83.4	81.15	49.65	79.65	**84.8**
20		23.15	83.5	84	79.3	37.8	78.65	**83.8**
25		18.65	81.35	82.65	74.05	30.35	75.3	**83.15**
30		18.6	77.95	79	71.5	24.1	72.9	**82.05**
35		16.95	71.2	73.4	67.75	21.05	71.45	**79.05**

Table 2.2: Classification results obtained by an SVM with RBF kernel.

α	clean	noisy	PCP	PCPF	AltProj	IRPCA-IHT	fast RPCA	our algorithm
10	92.25	87	87.25	87.3	86.45	89.3	89.25	**90.3**
15		75.85	87.15	87.4	86.75	82.85	87.2	**89.8**
20		64.35	87.6	87.55	84.65	71.2	85.55	**88.55**
25		55.85	87	86.95	79.4	62.35	82.65	**87.8**
30		47.15	81.15	81.55	76.75	53.5	78.3	**85.65**
35		40.55	74.8	75.7	71	47.4	76.75	**85.15**

2.6.3 Image Classification

The MNIST dataset contains hand-written digits divided into training and testing sets which can be used for image classification experiments. Let the observation matrix be composed of 2000 vectorized random images from the test set stacked column-wise. In this case, the left feature obtained from the training set is also applicable to the test set because of the Eigendigit nature. This imparts our algorithm to supervised learning where there are clean related training samples available. The right feature does not posses such property and is set to the identity matrix. We add a range of sparse noise to the test set separately where the noise sets the pixel to 255. For PCPF, we take $d = 300$ as in [32] and for IRPCA-IHT and our algorithm we use $d = 150$ instead.

The relative error between the recovered matrix by the competing algorithms and the clean test matrix is plotted in Figure 2.3. Our algorithm is most accurate in removing the added artificial noise. To evaluate how classifiers perform on the recovered matrices, we train the linear and kernel SVM using the training set and test the corresponding models on the recovered images. Table 2.1 tabulates the linear SVM. Table 2.2 tabulates the kernel SVM. Both classifiers confirm the recovery result obtained by various models corroborating our algorithm's pre-eminent accuracy.

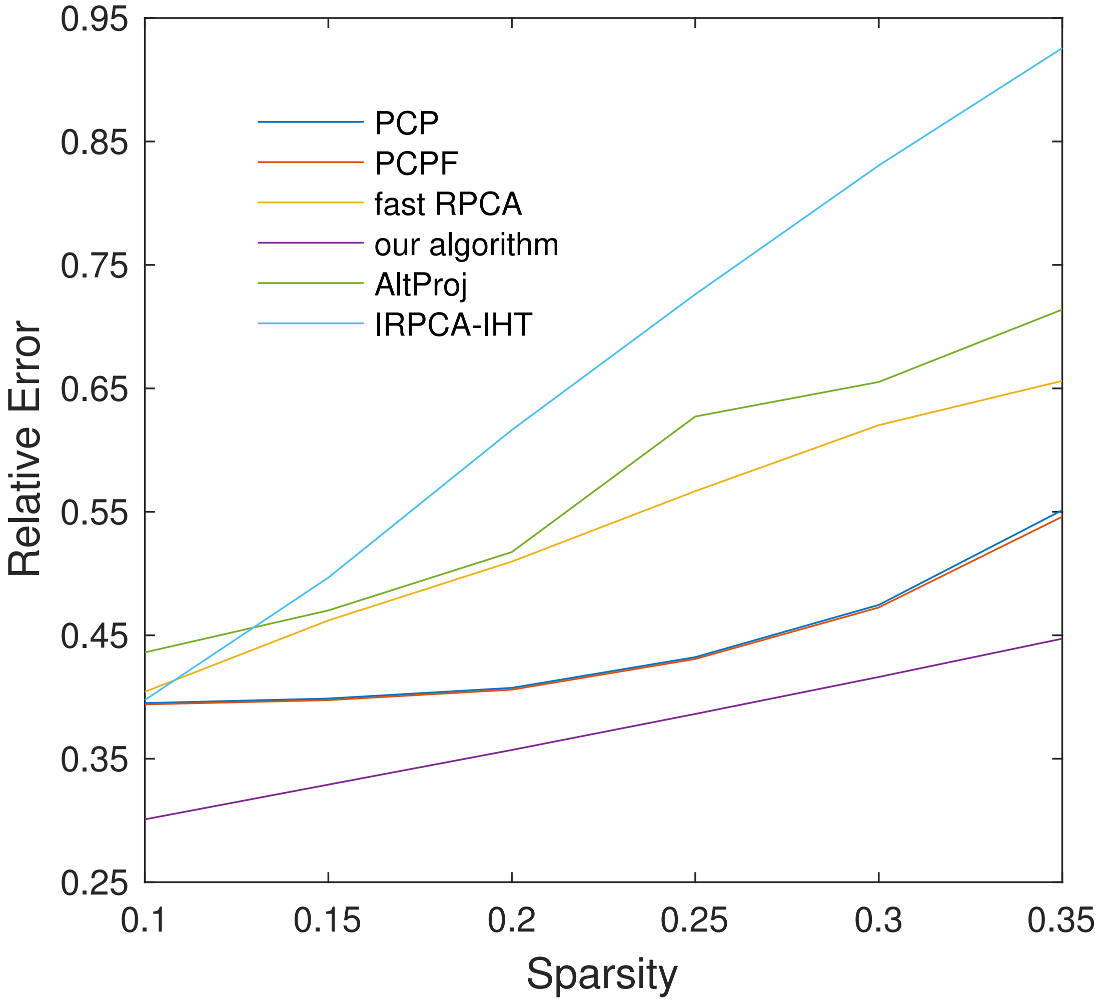


Figure 2.3: Relative error ($\frac{\|\boldsymbol{L}-\boldsymbol{L}^*\|_F}{\|\boldsymbol{L}^*\|_F}$) for sparsity values: 10%, 15%, 20%, 25%, 30%, 35%.

2.6.4 Face denoising

We demonstrate that there can be a substantial boost to the performance of facial denoising by leveraging dictionaries learnt from the images themselves. The extended Yale B database is used as our observation which consists images under different illuminations for a fixed pose. We study all 64 images of a randomly chosen person. A 32556 × 64 observation matrix is formed by vectorizing each 168 × 192 image. For fast RPCA and our algorithm, a sparsity of 0.2 is adopted. We learn the feature dictionary as in [135]. In a nutshell, the feature learning process can be treated as a sparse encoding problem. More specifically, we simultaneously seek a dictionary $\boldsymbol{D} \in \mathbb{R}^{n_1 \times c}$ and a sparse representation $\boldsymbol{B} \in \mathbb{R}^{c \times n_2}$ such that:

$$\begin{aligned} &\underset{\boldsymbol{D},\boldsymbol{B}}{\text{minimize}} \quad \|\boldsymbol{M} - \boldsymbol{D}\boldsymbol{B}\|_F^2 \\ &\text{s. t. to} \quad \gamma_i \leq t \text{ for } i = 1 \ldots n_2, \end{aligned} \tag{2.17}$$

where c is the number of atoms, γ_i's count the number of non-zero elements in each sparsity code and t is the sparsity constraint factor. This can be solved by the K-SVD algorithm. Here, feature $\boldsymbol{X}$ is the dictionary $\boldsymbol{D}$, feature $\boldsymbol{Y}$ corresponds to a similar solution using the transpose of the observation matrix as input. We set c to 40, t to 40 and used 10 iterations.

As a visual illustration, recovered images from all algorithms are exhibited in Figure 2.5. For this challenging scenario, our algorithm totally removed all shadows. PCPF is smoother than PCP but still suffers from shade. AltProj and fast RPCA both introduced extra artefacts. Although IRPCA-IHT managed to remove the shadows but brought back a severely distorted image. To quantitatively verify the improvement made by our proposed method, we examine the structural information contained within the denoised eigenfaces. Singular values of the recovered low-rank matrices from all algorithms are plotted in Figure 2.4. All non-convex algorithms are competent in incorporating the rank information to keep only 9 singular values, vastly outperforming convex approaches. Among them, our algorithm has the most rapid decay that is found naturally [126].

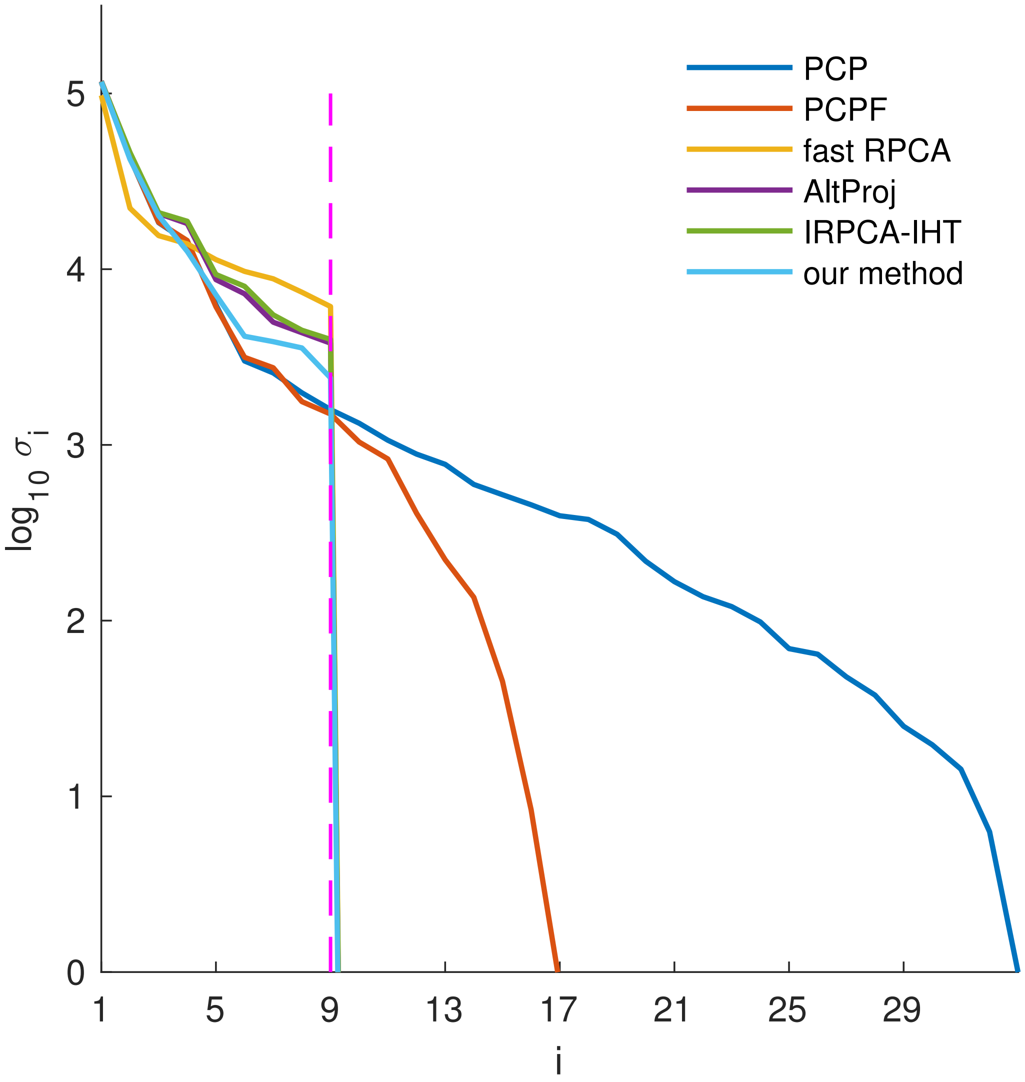


Figure 2.4: Log-scale singular values of the denoised matrices.

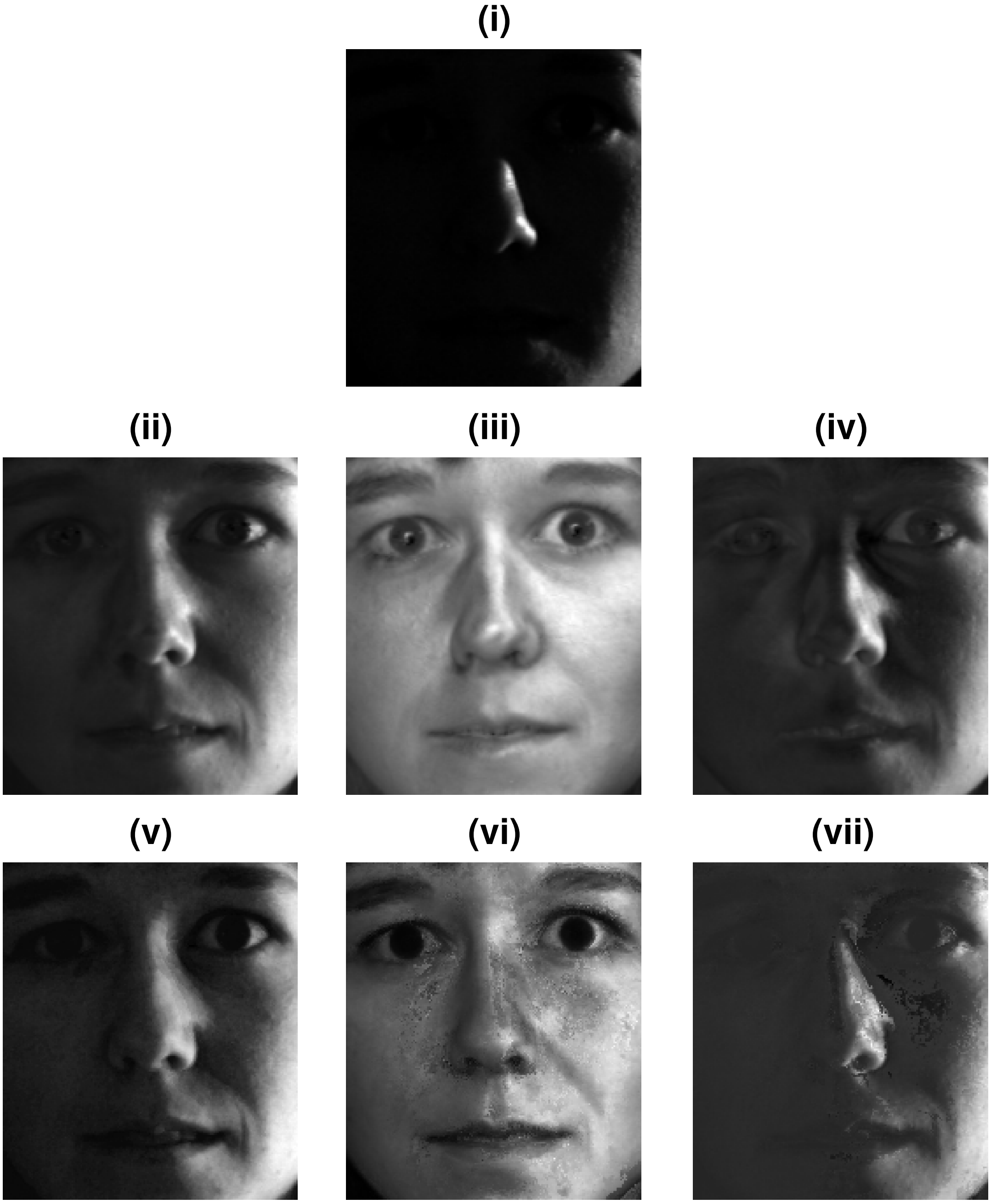


Figure 2.5: (i) original; (ii) PCPF; (iii) our algorithm; (iv) IRPCA-IHT; (v) PCP; (vi) fast RPCA; (vii) AltProj.

2.7 Appendix

2.7.1 Convex Projection

Given $\boldsymbol{P}$, the problem of finding $\boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P})$ can be seen as projection onto the intersection of a series of closed convex sets $\mathcal{P}_i$, that is $\mathcal{P} = \mathcal{P}_1 \bigcap \cdots \bigcap \mathcal{P}_{d_1}$, where $\mathcal{P}_i = \{\boldsymbol{A} \in \mathbb{R}^{d_1 \times r} | |\boldsymbol{X}_{i\cdot}\boldsymbol{A}|_2 \leq \sqrt{\frac{2\mu_1 r}{n_1}} \|\boldsymbol{P}_0\|_2\}$. We have emperically found that the Cyclic Dykstra algorithm [96] has the fastest rate of convergence. Let $\boldsymbol{A}_0 = \boldsymbol{P}$, and $\boldsymbol{B}_{-(d_1-1)} = \boldsymbol{B}_{-(d_1-2)} = \cdots = \boldsymbol{B}_{-1} = \boldsymbol{B}_0 = \boldsymbol{0} \in \mathbb{R}^{d_1 \times r}$, the Cyclic Dykstra algorithm updates, at each iteration, $\boldsymbol{A}_{k+1} = \boldsymbol{\Pi}_{\mathcal{P}_{k+1 \bmod d_1}}(\boldsymbol{A}_k + \boldsymbol{B}_{k+1-d_1})$ and $\boldsymbol{B}_{k+1} = \boldsymbol{A}_k + \boldsymbol{B}_{k+1-d_1} - \boldsymbol{A}_{k+1}$.

For $\boldsymbol{\Pi}_{\mathcal{P}_i}(\boldsymbol{P})$, we formulate the equivalent optimisation problem below

$$\min_{\boldsymbol{A}} \|\boldsymbol{A} - \boldsymbol{P}\|_F^2 \quad \text{s.t.} \quad |\boldsymbol{X}_{i\cdot}\boldsymbol{A}|_2 = \sqrt{\frac{2\mu_1 r}{n_1}} \|\boldsymbol{P}_0\|_2, \tag{2.18}$$

for $|\boldsymbol{X}_{i\cdot}\boldsymbol{P}|_2 > \sqrt{\frac{2\mu_1 r}{n_1}} \|\boldsymbol{P}_0\|_2$. Its solution is given by

$$A = (\boldsymbol{I}_{d_1 \times d_1} + \frac{(\frac{|\boldsymbol{X}_{i\cdot}\boldsymbol{P}|_2}{\sqrt{\frac{2\mu_1 r}{n_1}}\|\boldsymbol{P}_0\|_2} - 1)\boldsymbol{X}_{i\cdot}^\intercal \boldsymbol{X}_{i\cdot}}{|\boldsymbol{X}_{i\cdot}|_2^2})^{-1}\boldsymbol{P}. \tag{2.19}$$

For $\boldsymbol{Q}$, $\boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q})$ follows similarly.

We have also run experiments to see how much improvement can be gained by convex projection. 200×200 high-incoherence matrices are created with ranks from 140 to 155 and corrupted by 10% random sign errors. Our algorithm is applied with projection several times. Each uses a different number of iterative steps ranging from 0 to 2000. Recoverability is plotted against the number of iterative projections in Figure 2.6. There is hardly any noticeable improvement so we do not use convex projection in our comparison experiments. Further analysis is demanded to justify the redundency of convex projection.

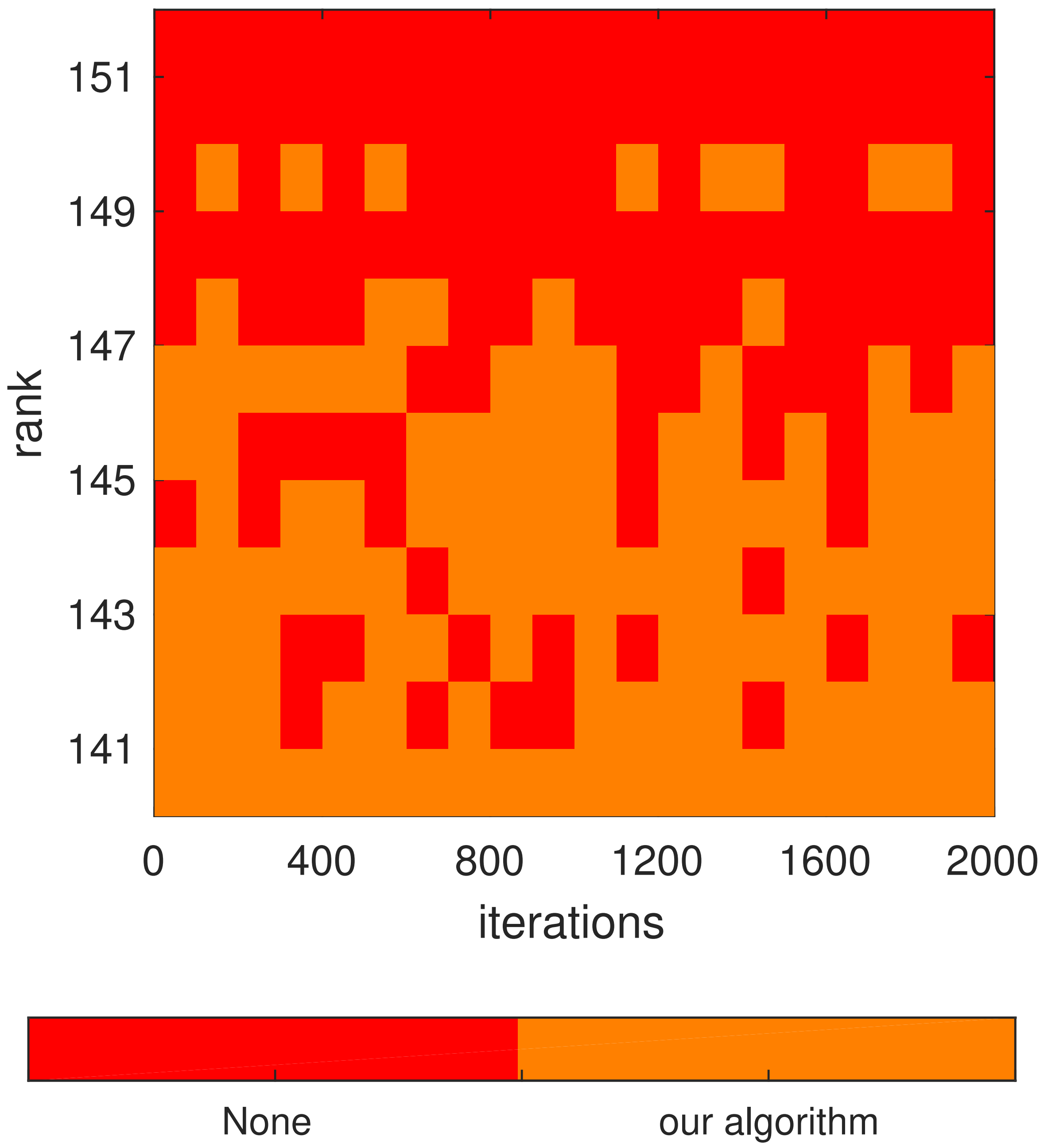


Figure 2.6: Effectiveness of convex projection.

2.7.2 Proofs

For simplicity, we assume that $n_1 = n_2 = n$, $d_1 = d_2 = d$.

2.7.3 Proof of Theorem 1

We first declare some lemmas that will be essential to our result.

Lemma 2.7.1. *Let $\boldsymbol{S}_0$ be obtained from the initialisation phase, we have*

$$\|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_\infty \leq 2\|\boldsymbol{L}^*\|_\infty. \tag{2.20}$$

Proof. See [138] theorem 1. □

Lemma 2.7.2. *For any matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ for which the proportion of non-zero entries per row and column is β, we have*

$$\|\boldsymbol{A}\|_2 \leq \beta n \|\boldsymbol{A}\|_\infty. \tag{2.21}$$

Proof. See [81] lemma 4. □

Lemma 2.7.3. *For two rank r matrices $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ of the same dimension whose compact SVDs are $\boldsymbol{L}_1 = \boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^\intercal$ and $\boldsymbol{L}_2 = \boldsymbol{U}_2\boldsymbol{\Sigma}_2\boldsymbol{V}_2^\intercal$, we have*

$$d(\boldsymbol{U}_1\boldsymbol{\Sigma}_1^{\frac{1}{2}}, \boldsymbol{V}_1\boldsymbol{\Sigma}_1^{\frac{1}{2}}, \boldsymbol{U}_2\boldsymbol{\Sigma}_2^{\frac{1}{2}}, \boldsymbol{V}_2\boldsymbol{\Sigma}_2^{\frac{1}{2}})^2 \leq \frac{2}{\sqrt{2}-1}\frac{\|\boldsymbol{L}_1 - \boldsymbol{L}_2\|_F^2}{\sigma_r(\boldsymbol{L}_2)}, \tag{2.22}$$

provided $\|\boldsymbol{L}_1 - \boldsymbol{L}_2\|_2 \leq \frac{1}{2}\sigma_r(\boldsymbol{L}_2)$.

Proof. See [120] lemma 5.14. □

Lemma 2.7.4. *For any matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of consistent sizes, we have*

$$\|\boldsymbol{A}\boldsymbol{B}\|_{2,\infty} \leq \|\boldsymbol{A}\|_{2,\infty}\|\boldsymbol{B}\|_2. \tag{2.23}$$

Proof. See [69] lemma 4.2. □

Lemma 2.7.5. *For any matrix $\boldsymbol{A}$ with compact SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\intercal$,*

$$\|\boldsymbol{A}\|_\infty \leq \|\boldsymbol{\Sigma}\|_2\|\boldsymbol{U}\|_{2,\infty}\|\boldsymbol{V}\|_{2,\infty}. \tag{2.24}$$

Proof. See [138] theorem 1. □

Lemma 2.7.6. *Let $\boldsymbol{U}_0, \boldsymbol{V}_0, \boldsymbol{\Sigma}_0, \boldsymbol{S}_0$ be obtained from the initialisation phase, we have*

$$\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top - \boldsymbol{M} + \boldsymbol{S}_0\|_2 \leq \|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2 \tag{2.25}$$

Proof. Weyl's theorem tells us that, for $1 \leq i \leq n$, $|\sigma_i(\boldsymbol{L}^*) - \sigma_i(\boldsymbol{M} - \boldsymbol{S}_0)| \leq \|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2$. When $i = r+1$, $\sigma_i(\boldsymbol{L}^*) = 0$ and $\sigma_i(\boldsymbol{M} - \boldsymbol{S}_0) = \|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top - \boldsymbol{M} + \boldsymbol{S}_0\|_2$ because $\boldsymbol{L}^*$ has rank r and $\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top = r\text{-SVD}(\boldsymbol{M} - \boldsymbol{S}_0)$. □

Lemma 2.7.7. *For $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D} \in \mathbb{R}^{d \times r}$*

$$d(\boldsymbol{X}^\top\boldsymbol{A}, \boldsymbol{Y}^\top\boldsymbol{B}, \boldsymbol{X}^\top\boldsymbol{C}, \boldsymbol{Y}^\top\boldsymbol{D}) \leq d(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}). \tag{2.26}$$

Proof.

$$\begin{aligned}
&d(\boldsymbol{X}^\top\boldsymbol{A}, \boldsymbol{Y}^\top\boldsymbol{B}, \boldsymbol{X}^\top\boldsymbol{C}, \boldsymbol{Y}^\top\boldsymbol{D}) \\
&= \min_{\boldsymbol{R}} \sqrt{\|\boldsymbol{X}^\top(\boldsymbol{A} - \boldsymbol{C}\boldsymbol{R})\|_F^2 + \|\boldsymbol{Y}^\top(\boldsymbol{B} - \boldsymbol{D}\boldsymbol{R})\|_F^2} \\
&\leq \min_{\boldsymbol{R}} \sqrt{\|\boldsymbol{X}^\top\|_2^2\|(\boldsymbol{A} - \boldsymbol{C}\boldsymbol{R})\|_F^2 + \|\boldsymbol{Y}^\top\|_2^2\|(\boldsymbol{B} - \boldsymbol{D}\boldsymbol{R})\|_F^2} \\
&= \min_{\boldsymbol{R}} \sqrt{\|(\boldsymbol{A} - \boldsymbol{C}\boldsymbol{R})\|_F^2 + \|(\boldsymbol{B} - \boldsymbol{D}\boldsymbol{R})\|_F^2} \\
&= d(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}).
\end{aligned} \tag{2.27}$$

□

We begin by deriving a bound on $\|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2$,

$$\begin{aligned}
\|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2 \leq 2\alpha n\|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_\infty &\leq 4\alpha n\|\boldsymbol{L}^*\|_\infty \\
&\leq 4\alpha n\|\boldsymbol{\Sigma}^*\|_2\|\boldsymbol{U}^*\|_{2,\infty}\|\boldsymbol{V}^*\|_{2,\infty},
\end{aligned} \tag{2.28}$$

where the first inequality follows from **Lemma 2.7.2** with $\beta = 2\alpha$, the second from **Lemma 2.7.1** and the third from **Lemma 2.7.5**. Next, we look at $\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top - \boldsymbol{L}^*\|_2$:

$$\begin{aligned}
&\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top - \boldsymbol{L}^*\|_2 \\
&\leq \|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\top - \boldsymbol{M} + \boldsymbol{S}_0\|_2 + \|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2 \\
&\leq 2\|\boldsymbol{M} - \boldsymbol{S}_0 - \boldsymbol{L}^*\|_2 \\
&\leq 8\alpha n\|\boldsymbol{\Sigma}^*\|_2\|\boldsymbol{U}^*\|_{2,\infty}\|\boldsymbol{V}^*\|_{2,\infty},
\end{aligned} \tag{2.29}$$

where we have used **Lemma 2.7.6** and (2.28).

In cases (i) and (iii), the condition $\alpha \leq \frac{1}{16\kappa\mu_1 r}$ gives $\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal - \boldsymbol{L}^*\|_2 \leq \frac{1}{2}\sigma_r(\boldsymbol{L}^*)$ and we have

$$\begin{aligned}
& d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*)^2 \\
&= d(\boldsymbol{X}^\intercal\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}, \boldsymbol{Y}^\intercal\boldsymbol{V}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}, \boldsymbol{X}^\intercal\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}, \boldsymbol{Y}^\intercal\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}})^2 \\
&\leq d(\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}, \boldsymbol{V}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}, \boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}, \boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}})^2 \\
&\leq \frac{2}{\sqrt{2}-1}\frac{\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal - \boldsymbol{L}^*\|_F^2}{\sigma_r(\boldsymbol{L}^*)} \\
&\leq \frac{2r}{\sqrt{2}-1}\frac{\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal - \boldsymbol{L}^*\|_2^2}{\sigma_r(\boldsymbol{L}^*)} \leq \frac{128 r^3\alpha^2\kappa\sigma_1^*\mu_1^2}{\sqrt{2}-1},
\end{aligned} \tag{2.30}$$

using **Lemma 2.7.7**, **Lemma 2.7.3** and (2.29). So, we have

$$d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\mu_1\alpha r\sqrt{r\kappa\sigma_1^*}. \tag{2.31}$$

In case (ii), we have

$$\|\boldsymbol{U}^*\|_{2,\infty} = \|\boldsymbol{X}\boldsymbol{X}^\intercal\boldsymbol{U}^*\|_{2,\infty} \leq \|\boldsymbol{X}^\intercal\boldsymbol{U}^*\|_2\|\boldsymbol{X}\|_{2,\infty} \leq \sqrt{\frac{\mu_2 d}{n}}, \tag{2.32}$$

$$\|\boldsymbol{V}^*\|_{2,\infty} = \|\boldsymbol{Y}\boldsymbol{Y}^\intercal\boldsymbol{V}^*\|_{2,\infty} \leq \|\boldsymbol{Y}^\intercal\boldsymbol{V}^*\|_2\|\boldsymbol{Y}\|_{2,\infty} \leq \sqrt{\frac{\mu_2 d}{n}}. \tag{2.33}$$

The condition $\alpha \leq \frac{1}{16\kappa\mu_2 d}$ gives $\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal - \boldsymbol{L}^*\|_2 \leq \frac{1}{2}\sigma_r(\boldsymbol{L}^*)$ and we have similar to (2.31)

$$d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\mu_2\alpha d\sqrt{r\kappa\sigma_1^*}. \tag{2.34}$$

2.7.4 Proof of Theorem 2

To ease our exposition, we define the following auxiliary quantities.

Let the solution set be

$$\mathcal{E} = \{(\boldsymbol{A}, \boldsymbol{B}) \in \mathbb{R}^{d\times r} \times \mathbb{R}^{d\times r} | d(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{P}^*, \boldsymbol{Q}^*) = 0\}. \tag{2.35}$$

For any $(\boldsymbol{P}, \boldsymbol{Q}) \in \mathbb{R}^{d\times r} \times \mathbb{R}^{d\times r}$, the corresponding solution is given by

$$(\boldsymbol{P}^\dagger, \boldsymbol{Q}^\dagger) \in \arg\min_{(\boldsymbol{A},\boldsymbol{B})\in\mathcal{E}} \|\boldsymbol{P} - \boldsymbol{A}\|_F^2 + \|\boldsymbol{Q} - \boldsymbol{B}\|_F^2. \tag{2.36}$$

Let $\Delta\boldsymbol{P} = \boldsymbol{P} - \boldsymbol{P}^{\dagger}$, $\Delta\boldsymbol{Q} = \boldsymbol{Q} - \boldsymbol{Q}^{\dagger}$ and $\delta = \|\Delta\boldsymbol{P}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F^2$, from which we have

$$\begin{aligned}
2\|\Delta\boldsymbol{P}\|_F\|\Delta\boldsymbol{Q}\|_F &\leq \|\Delta\boldsymbol{P}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F^2,\\
\|\Delta\boldsymbol{P}\|_F + \|\Delta\boldsymbol{Q}\|_F &\leq \sqrt{2\delta},\\
\|\Delta\boldsymbol{P}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F\|\Delta\boldsymbol{P}\|_F &\leq \sqrt{2\delta}\|\Delta\boldsymbol{P}\|_F,\\
\|\Delta\boldsymbol{Q}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F\|\Delta\boldsymbol{P}\|_F &\leq \sqrt{2\delta}\|\Delta\boldsymbol{Q}\|_F,\\
4\|\Delta\boldsymbol{Q}\|_F\|\Delta\boldsymbol{P}\|_F &\leq \|\Delta\boldsymbol{P}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F^2 + 2\|\Delta\boldsymbol{Q}\|_F\|\Delta\boldsymbol{P}\|_F\\
&\leq \sqrt{2\delta}(\|\Delta\boldsymbol{Q}\|_F + \|\Delta\boldsymbol{P}\|_F).
\end{aligned} \tag{2.37}$$

Let $\mathcal{H} = \frac{1}{2}\|\boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^{\intercal}\boldsymbol{Y}^{\intercal} + \boldsymbol{S} - \boldsymbol{M}\|_F^2$ and $\Delta\boldsymbol{M} = \nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})$, we have

$$\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q}) = \boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^{\intercal}\boldsymbol{Y}^{\intercal} + \boldsymbol{S} - \boldsymbol{M} = \boldsymbol{L} + \boldsymbol{S} - \boldsymbol{L}^* - \boldsymbol{S}^*. \tag{2.38}$$

We also have

$$\nabla_{\boldsymbol{P}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q}) = \boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}\boldsymbol{Y}\boldsymbol{Q}, \tag{2.39}$$

$$\nabla_{\boldsymbol{Q}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q}) = (\boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}\boldsymbol{Y})^{\intercal}\boldsymbol{P}. \tag{2.40}$$

Let $\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}) = \frac{1}{64}\|\boldsymbol{P}^{\intercal}\boldsymbol{P} - \boldsymbol{Q}^{\intercal}\boldsymbol{Q}\|_F^2$, we have

$$\nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}) = \frac{1}{16}\boldsymbol{P}(\boldsymbol{P}^{\intercal}\boldsymbol{P} - \boldsymbol{Q}^{\intercal}\boldsymbol{Q}), \tag{2.41}$$

$$\nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}) = \frac{1}{16}\boldsymbol{Q}(\boldsymbol{Q}^{\intercal}\boldsymbol{Q} - \boldsymbol{P}^{\intercal}\boldsymbol{P}). \tag{2.42}$$

Let $\boldsymbol{F} = \begin{bmatrix}\boldsymbol{P}\\ \boldsymbol{Q}\end{bmatrix}$, $\boldsymbol{F}^{\dagger} = \begin{bmatrix}\boldsymbol{P}^{\dagger}\\ \boldsymbol{Q}^{\dagger}\end{bmatrix}$ and $\Delta\boldsymbol{F} = \boldsymbol{F} - \boldsymbol{F}^{\dagger}$, then we have $\delta = \|\Delta\boldsymbol{F}\|_F^2$.

We now state several lemmas that will help us construct the proof.

Lemma 2.7.8. *For any $\boldsymbol{P} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{Q} \in \mathbb{R}^{d\times r}$, we have*

$$\|\boldsymbol{L} - \boldsymbol{L}^*\|_F^2 \leq 2\delta(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta}}{4})^2. \tag{2.43}$$

Proof.

$$\begin{aligned}
\|\boldsymbol{L}-\boldsymbol{L}^*\|_F &= \|\boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal - \boldsymbol{X}\boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T}\boldsymbol{Y}^\intercal\|_F \\
&= \|\boldsymbol{X}(\boldsymbol{P}^\dagger\Delta\boldsymbol{Q}^\intercal + \Delta\boldsymbol{P}\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal)\boldsymbol{Y}^\intercal\|_F \\
&\le \|\boldsymbol{P}^\dagger\Delta\boldsymbol{Q}^\intercal\|_F + \|\Delta\boldsymbol{P}\boldsymbol{Q}^{\dagger T}\|_F + \|\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\|_F \\
&\le \|\boldsymbol{P}^\dagger\|_2\|\Delta\boldsymbol{Q}\|_F + \|\Delta\boldsymbol{P}\|_F\|\boldsymbol{Q}^\dagger\|_2 + \|\Delta\boldsymbol{P}\|_F\|\Delta\boldsymbol{Q}\|_F \\
&\le \sqrt{\sigma_1^*}\|\Delta\boldsymbol{Q}\|_F + \sqrt{\sigma_1^*}\|\Delta\boldsymbol{P}\|_F + \frac{\sqrt{2\delta}}{4}(\|\Delta\boldsymbol{Q}\|_F + \|\Delta\boldsymbol{P}\|_F) \\
&\le (\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta}}{4})(\|\Delta\boldsymbol{Q}\|_F + \|\Delta\boldsymbol{P}\|_F) \le (\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta}}{4})\sqrt{2\delta}.
\end{aligned} \tag{2.44}$$

□

Lemma 2.7.9. *For $1 \le i,j \le n$, in case (i), if $\|\boldsymbol{X}\boldsymbol{P}\|_{2,\infty} \le \sqrt{\frac{3\mu_1 r\sigma_1^*}{2n}}$ and $\|\boldsymbol{Y}\boldsymbol{Q}\|_{2,\infty} \le \sqrt{\frac{3\mu_1 r\sigma_1^*}{2n}}$, then*

$$|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}| \le \frac{1}{2}\sqrt{\frac{\mu_1 r\sigma_1^*}{n}}(3+\sqrt{\frac{3}{2}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2 + \|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2) \tag{2.45}$$

and in cases (ii) and (iii), if $\|\boldsymbol{X}\boldsymbol{P}\|_{2,\infty} \le \sqrt{\frac{3\mu_2 d\sigma_1^*}{2n}}$ and $\|\boldsymbol{Y}\boldsymbol{Q}\|_{2,\infty} \le \sqrt{\frac{3\mu_2 d\sigma_1^*}{2n}}$, then

$$|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}| \le \frac{1}{2}\sqrt{\frac{\mu_2 d\sigma_1^*}{n}}(3+\sqrt{\frac{3}{2}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2 + \|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2). \tag{2.46}$$

Proof.

$$\begin{aligned}
&|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}| \\
&= |(\boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal - \boldsymbol{X}\boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T}\boldsymbol{Y}^\intercal)_{ij}| \\
&\le |(\boldsymbol{X}\boldsymbol{P}^\dagger)_{i\cdot}(\boldsymbol{Y}\Delta\boldsymbol{Q})^\intercal_{\cdot j}| + |(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}(\boldsymbol{Y}\boldsymbol{Q}^\dagger)^\intercal_{\cdot j}| + |(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}(\boldsymbol{Y}\Delta\boldsymbol{Q})^\intercal_{\cdot j}| \\
&\le \|(\boldsymbol{X}\boldsymbol{P}^\dagger)_{i\cdot}\|_2\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2 + \|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2\|(\boldsymbol{Y}\boldsymbol{Q}^\dagger)_{j\cdot}\|_2 + \|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2 \\
&\le \|\boldsymbol{X}\boldsymbol{P}^\dagger\|_{2,\infty}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2 + \|\boldsymbol{Y}\boldsymbol{Q}^\dagger\|_{2,\infty}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2 \\
&\quad + \frac{1}{2}\|\boldsymbol{X}\Delta\boldsymbol{P}\|_{2,\infty}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2 + \frac{1}{2}\|\boldsymbol{Y}\Delta\boldsymbol{Q}\|_{2,\infty}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2 \\
&\le \frac{1}{2}((3\|\boldsymbol{X}\boldsymbol{P}^\dagger\|_{2,\infty} + \|\boldsymbol{X}\boldsymbol{P}\|_{2,\infty})\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2 \\
&\quad + (3\|\boldsymbol{Y}\boldsymbol{Q}^\dagger\|_{2,\infty} + \|\boldsymbol{Y}\boldsymbol{Q}\|_{2,\infty})\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2),
\end{aligned} \tag{2.47}$$

where we have used $\|\boldsymbol{X}\Delta\boldsymbol{P}\|_{2,\infty} \le \|\boldsymbol{X}\boldsymbol{P}^\dagger\|_{2,\infty} + \|\boldsymbol{X}\boldsymbol{P}\|_{2,\infty}$ and $\|\boldsymbol{Y}\Delta\boldsymbol{Q}\|_{2,\infty} \le \|\boldsymbol{Y}\boldsymbol{Q}^\dagger\|_{2,\infty} + \|\boldsymbol{Y}\boldsymbol{Q}\|_{2,\infty}$.

In case (i),

$$
\begin{aligned}
|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}| &\leq \frac{1}{2}(3\sqrt{\frac{\mu_1 r\sigma_1^*}{n}}+\sqrt{\frac{3\mu_1 r\sigma_1^*}{2n}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2) \\
&= \frac{1}{2}\sqrt{\frac{\mu_1 r\sigma_1^*}{n}}(3+\sqrt{\frac{3}{2}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2).
\end{aligned}
\tag{2.48}
$$

In cases (ii) and (iii),

$$
\begin{aligned}
|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}| &\leq \frac{1}{2}(3\sqrt{\frac{\mu_2 d\sigma_1^*}{n}}+\sqrt{\frac{3\mu_2 d\sigma_1^*}{2n}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2) \\
&= \frac{1}{2}\sqrt{\frac{\mu_2 d\sigma_1^*}{n}}(3+\sqrt{\frac{3}{2}})(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2).
\end{aligned}
\tag{2.49}
$$

□

Lemma 2.7.10. *For any $\alpha \in (0,1)$, suppose the support index set $\Omega \subseteq [n]\times[n]$ satisfies $|\Omega_{i\cdot}| \leq \alpha n$ for all $i \in [n]$ and $|\Omega_{\cdot j}| \leq \alpha n$ for all $j \in [n]$ where $\Omega_{i\cdot} = \{(i,j) \in \Omega | j \in [n]\}$ and $\Omega_{\cdot j} = \{(i,j) \in \Omega | i \in [n]\}$. In case (i), we have*

$$
\|\Pi_\Omega(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 \leq \frac{\alpha\mu_1 r\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2),
\tag{2.50}
$$

and in cases (ii) and (iii), we have

$$
\|\Pi_\Omega(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 \leq \frac{\alpha\mu_2 d\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2).
\tag{2.51}
$$

Proof.

$$
\|\Pi_\Omega(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 = \sum_{i,j\in\boldsymbol{\Omega}} |(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}|^2.
\tag{2.52}
$$

Using Lemma 2.7.9, in case (i),

$$\begin{aligned}
&\sum_{i,j\in\boldsymbol{\Omega}}|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}|^2\\
\leq&\sum_{i,j\in\boldsymbol{\Omega}}\frac{\mu_1 r\sigma_1^*}{4n}(3+\sqrt{\frac{3}{2}})^2(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2)^2\\
\leq&\sum_{i,j\in\boldsymbol{\Omega}}\frac{\mu_1 r\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2)\\
\leq&\frac{\mu_1 r\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\sum_{i,j\in\boldsymbol{\Omega}}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\sum_{i,j\in\boldsymbol{\Omega}}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2) \qquad (2.53)\\
\leq&\frac{\mu_1 r\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\sum_{i}\sum_{j\in\boldsymbol{\Omega}_{i\cdot}}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\sum_{j}\sum_{i\in\boldsymbol{\Omega}_{\cdot j}}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2)\\
\leq&\frac{\alpha\mu_1 r\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\boldsymbol{X}\Delta\boldsymbol{P}\|_F^2+\|\boldsymbol{Y}\Delta\boldsymbol{Q}\|_F^2)\\
\leq&\frac{\alpha\mu_1 r\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2).
\end{aligned}$$

and in cases (ii) and (iii),

$$\begin{aligned}
&\sum_{i,j\in\boldsymbol{\Omega}}|(\boldsymbol{L}-\boldsymbol{L}^*)_{ij}|^2\\
\leq&\sum_{i,j\in\boldsymbol{\Omega}}\frac{\mu_2 d\sigma_1^*}{4n}(3+\sqrt{\frac{3}{2}})^2(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2)^2\\
\leq&\sum_{i,j\in\boldsymbol{\Omega}}\frac{\mu_2 d\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2)\\
\leq&\frac{\mu_2 d\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\sum_{i,j\in\boldsymbol{\Omega}}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\sum_{i,j\in\boldsymbol{\Omega}}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2) \qquad (2.54)\\
\leq&\frac{\mu_2 d\sigma_1^*}{2n}(3+\sqrt{\frac{3}{2}})^2(\sum_{i}\sum_{j\in\boldsymbol{\Omega}_{i\cdot}}\|(\boldsymbol{X}\Delta\boldsymbol{P})_{i\cdot}\|_2^2+\sum_{j}\sum_{i\in\boldsymbol{\Omega}_{\cdot j}}\|(\boldsymbol{Y}\Delta\boldsymbol{Q})_{j\cdot}\|_2^2)\\
\leq&\frac{\alpha\mu_2 d\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\boldsymbol{X}\Delta\boldsymbol{P}\|_F^2+\|\boldsymbol{Y}\Delta\boldsymbol{Q}\|_F^2)\\
\leq&\frac{\alpha\mu_2 d\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2).
\end{aligned}$$

□

Lemma 2.7.11. *Given that $\boldsymbol{S} = \mathcal{T}_{\alpha+\min(10\alpha,0.1)}(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal)$, we have in case (i)*

$$\begin{aligned}\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle &\geq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 \\ - \frac{\mu_1 r\sigma_1^*\delta}{4}((4+\beta)\alpha + 2\min(10\alpha,0.1))(3+\sqrt{\frac{3}{2}})^2 &- \frac{2\alpha\delta}{\beta\min(10\alpha,0.1)}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 \\ &- \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}),\end{aligned} \tag{2.55}$$

and in cases (ii) and (iii)

$$\begin{aligned}\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle &\geq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 \\ - \frac{\mu_2 d\sigma_1^*\delta}{4}((4+\beta)\alpha + 2\min(10\alpha,0.1))(3+\sqrt{\frac{3}{2}})^2 &- \frac{2\alpha\delta}{\beta\min(10\alpha,0.1)}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 \\ &- \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}).\end{aligned} \tag{2.56}$$

Proof.

$$\begin{aligned}&\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle \\ &= \langle \boldsymbol{L}+\boldsymbol{S}-\boldsymbol{L}^*-\boldsymbol{S}^*, \boldsymbol{L}-\boldsymbol{L}^* + \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle \\ &\geq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - |\langle \boldsymbol{S}-\boldsymbol{S}^*, \boldsymbol{L}-\boldsymbol{L}^*\rangle| - |\langle \boldsymbol{L}+\boldsymbol{S}-\boldsymbol{L}^*-\boldsymbol{S}^*, \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle|.\end{aligned} \tag{2.57}$$

Following [138] lemma 2, we have

$$|\langle \boldsymbol{S}-\boldsymbol{S}^*, \boldsymbol{L}-\boldsymbol{L}^*\rangle| \leq \|\boldsymbol{\Pi}_{\boldsymbol{\Omega}}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 + (1+\frac{\beta}{2})\|\boldsymbol{\Pi}_{\boldsymbol{\Omega}}^*\backslash\boldsymbol{\Omega}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 + \frac{\alpha}{\beta\min(10\alpha,0.1)}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2, \tag{2.58}$$

where $\beta > 0$, $\boldsymbol{\Omega}$ and $\boldsymbol{\Omega}^*$ are supports of $\boldsymbol{S}$ and $\boldsymbol{S}^*$ respectively.

On the other hand,

$$\begin{aligned}|\langle \boldsymbol{L}+\boldsymbol{S}-\boldsymbol{L}^*-\boldsymbol{S}^*, \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle| \leq |\langle \boldsymbol{\Pi}_{\boldsymbol{\Omega}}^{*c}\cap\boldsymbol{\Omega}^c(\boldsymbol{L}-\boldsymbol{L}^*), \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle| \\ + |\langle \boldsymbol{\Pi}_{\boldsymbol{\Omega}}^{*}\cap\boldsymbol{\Omega}^c(\Delta\boldsymbol{M}), \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle|,\end{aligned} \tag{2.59}$$

because $\Delta\boldsymbol{M}$ has support $\boldsymbol{\Omega}^c$. From Cauchy-Swartz inequality, we have

$$\begin{aligned}&|\langle \boldsymbol{\Pi}_{\boldsymbol{\Omega}}^{*c}\cap\boldsymbol{\Omega}^c(\boldsymbol{L}-\boldsymbol{L}^*), \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle| \\ &\leq \|\boldsymbol{\Pi}_{\boldsymbol{\Omega}}^{*c}\cap\boldsymbol{\Omega}^c(\boldsymbol{L}-\boldsymbol{L}^*)\|_F\|\boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\|_F \\ &\leq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F\|\boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\|_F \\ &\leq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F\|\Delta\boldsymbol{P}\|_F\|\Delta\boldsymbol{Q}\|_F \\ &\leq \frac{\delta}{2}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F.\end{aligned} \tag{2.60}$$

From [138] lemma 2, we have

$$\begin{aligned}|\langle \mathbf{\Pi}^*_{\mathbf{\Omega}} \cap \mathbf{\Omega}^c(\Delta \boldsymbol{M}), \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle| &\leq \sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F\|\boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\|_F \\ &\leq \delta\sqrt{\frac{\alpha}{2\min(10\alpha,0.1)}}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F.\end{aligned} \tag{2.61}$$

So,

$$\begin{aligned}|\langle \boldsymbol{L}+\boldsymbol{S}-\boldsymbol{L}^*-\boldsymbol{S}^*, \boldsymbol{X}\Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal\rangle| &\leq \frac{\delta}{2}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F + \delta\sqrt{\frac{\alpha}{2\min(10\alpha,0.1)}}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F \\ &\leq \frac{\delta}{2}(1+\sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}})\|\boldsymbol{L}-\boldsymbol{L}^*\|_F.\end{aligned} \tag{2.62}$$

Together, we have

$$\begin{aligned}&\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle \geq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \|\mathbf{\Pi}_{\mathbf{\Omega}}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 \\ &-(1+\frac{\beta}{2})\|\mathbf{\Pi}^*_{\mathbf{\Omega}}\backslash\mathbf{\Omega}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 - \frac{\alpha}{\beta\min(10\alpha,0.1)}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\delta}{2}(1+\sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}})\|\boldsymbol{L}-\boldsymbol{L}^*\|_F.\end{aligned} \tag{2.63}$$

From **Lemma 2.7.8**, we have

$$\begin{aligned}&\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle \geq \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \|\mathbf{\Pi}_{\mathbf{\Omega}}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 \\ &-(1+\frac{\beta}{2})\|\mathbf{\Pi}^*_{\mathbf{\Omega}}\backslash\mathbf{\Omega}(\boldsymbol{L}-\boldsymbol{L}^*)\|_F^2 - \frac{2\alpha\delta}{\beta\min(10\alpha,0.1)}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 - \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}).\end{aligned} \tag{2.64}$$

Since $\mathbf{\Pi}_{\mathbf{\Omega}}(\boldsymbol{L}-\boldsymbol{L}^*)$ and $\mathbf{\Pi}^*_{\mathbf{\Omega}}\backslash\mathbf{\Omega}(\boldsymbol{L}-\boldsymbol{L}^*)$ have at most $\alpha+\min(10\alpha,0.1)$-fraction and α-fraction non-zero entries per row and column respectively, from **Lemma 2.7.10**, we have in case (i)

$$\begin{aligned}&\langle \boldsymbol{X}^\intercal\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\intercal - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\intercal\rangle \\ \geq\ & \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\alpha\mu_1 r\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\ &- \frac{\min(10\alpha,0.1)\mu_1 r\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\ &- \frac{\alpha\mu_1 r\sigma_1^*}{2}(1+\frac{\beta}{2})(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\ &- \frac{2\alpha\delta}{\beta\min(10\alpha,0.1)}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 - \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}) \\ \geq\ & \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\mu_1 r\sigma_1^*\delta}{4}((4+\beta)\alpha+2\min(10\alpha,0.1))(3+\sqrt{\frac{3}{2}})^2 \\ &- \frac{2\alpha\delta}{\beta\min(10\alpha,0.1)}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 - \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}),\end{aligned} \tag{2.65}$$

and in cases (ii) and (iii)

$$\begin{aligned}
&\langle \boldsymbol{X}^\top \nabla_{\boldsymbol{L}} \mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\boldsymbol{Y}, \boldsymbol{P}\boldsymbol{Q}^\top - \boldsymbol{P}^\dagger \boldsymbol{Q}^{\dagger T} + \Delta\boldsymbol{P}\Delta\boldsymbol{Q}^\top\rangle \\
\geq{}& \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\alpha\mu_2 d\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\
&- \frac{\min(10\alpha,0.1)\mu_2 d\sigma_1^*}{2}(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\
&- \frac{\alpha\mu_2 d\sigma_1^*}{2}(1+\frac{\beta}{2})(3+\sqrt{\frac{3}{2}})^2(\|\Delta\boldsymbol{P}\|_F^2+\|\Delta\boldsymbol{Q}\|_F^2) \\
&- \frac{2\min(10\alpha,0.1)\alpha\delta}{\beta}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 - \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}) \\
\geq{}& \|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\mu_2 d\sigma_1^*\delta}{4}((4+\beta)\alpha+2\min(10\alpha,0.1))(3+\sqrt{\frac{3}{2}})^2 \\
&- \frac{2\min(10\alpha,0.1)\alpha\delta}{\beta}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4})^2 - \frac{\sqrt{2}+2\sqrt{\frac{\alpha}{\min(10\alpha,0.1)}}}{2}\sqrt{\delta^3}(\sqrt{\sigma_1^*}+\frac{\sqrt{2\delta}}{4}).
\end{aligned} \tag{2.66}$$

□

Lemma 2.7.12. *When $\|\boldsymbol{F}-\boldsymbol{F}^\dagger\|_2 \leq \sqrt{2\sigma_r^*}$, given that $\|\boldsymbol{P}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}$ and $\|\boldsymbol{Q}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}$ we have*

$$\begin{aligned}
\langle \nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{P}-\boldsymbol{P}^\dagger\rangle + \langle \nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{Q}-\boldsymbol{Q}^\dagger\rangle \geq \frac{1}{64}\|\boldsymbol{P}^\top\boldsymbol{P}-\boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 \\
+\frac{1}{64}(2\sqrt{\sigma_r^*\delta}-\delta)^2 - \frac{1}{16}\|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2 - \frac{\sqrt{2}+\sqrt{3}}{32}\sqrt{\sigma_1^*\delta^3}.
\end{aligned} \tag{2.67}$$

Proof.

$$\begin{aligned}
\boldsymbol{P}^{\dagger T}\boldsymbol{P}^\dagger &= (\boldsymbol{X}^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R})^\top(\boldsymbol{X}^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R}) \\
&= \boldsymbol{R}^\top\boldsymbol{\Sigma}^{*\frac{1}{2}T}\boldsymbol{U}^{*T}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R} \\
&= \boldsymbol{R}^\top\boldsymbol{\Sigma}^{*\frac{1}{2}T}\boldsymbol{U}^{*T}\boldsymbol{U}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R} \\
&= \boldsymbol{R}^\top\boldsymbol{\Sigma}^{*\frac{1}{2}T}\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R} \\
&= \boldsymbol{R}^\top\boldsymbol{\Sigma}^{*\frac{1}{2}T}\boldsymbol{V}^{*T}\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R} \\
&= \boldsymbol{R}^\top\boldsymbol{\Sigma}^{*\frac{1}{2}T}\boldsymbol{V}^{*T}\boldsymbol{Y}\boldsymbol{Y}^\top\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R} \\
&= (\boldsymbol{Y}^\top\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R})^\top(\boldsymbol{Y}^\top\boldsymbol{V}^*\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R}) \\
&= \boldsymbol{Q}^{\dagger T}\boldsymbol{Q}^\dagger.
\end{aligned} \tag{2.68}$$

Then, following [138] lemma 3, we have

$$\begin{aligned}
&\langle \nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{P}-\boldsymbol{P}^\dagger\rangle + \langle \nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{Q}-\boldsymbol{Q}^\dagger\rangle \\
&\qquad = \frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P}-\boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 + \frac{1}{32}\langle \boldsymbol{P}^\top\boldsymbol{P}-\boldsymbol{Q}^\top\boldsymbol{Q}, \Delta\boldsymbol{P}^\top\Delta\boldsymbol{P}-\Delta\boldsymbol{Q}^\top\Delta\boldsymbol{Q}\rangle.
\end{aligned} \tag{2.69}$$

$$
\begin{aligned}
&\frac{1}{32}\langle \boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}, \Delta\boldsymbol{P}^\top\Delta\boldsymbol{P} - \Delta\boldsymbol{Q}^\top\Delta\boldsymbol{Q}\rangle \\
\leq\ & \frac{1}{32}|\langle \boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}, \Delta\boldsymbol{P}^\top\Delta\boldsymbol{P} - \Delta\boldsymbol{Q}^\top\Delta\boldsymbol{Q}\rangle| \\
\leq\ & \frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F\|\Delta\boldsymbol{P}^\top\Delta\boldsymbol{P} - \Delta\boldsymbol{Q}^\top\Delta\boldsymbol{Q}\|_F \\
\leq\ & \frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F(\|\Delta\boldsymbol{P}\|_F^2 + \|\Delta\boldsymbol{Q}\|_F^2) \\
\leq\ & \frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{P}^{\dagger T}\boldsymbol{P}^\dagger + \boldsymbol{Q}^{\dagger T}\boldsymbol{Q}^\dagger - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F\delta \\
\leq\ & \frac{1}{32}(\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{P}^{\dagger T}\boldsymbol{P}^\dagger\|_F + \|\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{Q}^{\dagger T}\boldsymbol{Q}^\dagger\|_F)\delta \qquad (2.70) \\
\leq\ & \frac{1}{32}(\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{P}^\top\boldsymbol{P}^\dagger + \boldsymbol{P}^\top\boldsymbol{P}^\dagger - \boldsymbol{P}^{\dagger T}\boldsymbol{P}^\dagger\|_F + \|\boldsymbol{Q}^\top\boldsymbol{Q} - \boldsymbol{Q}^\top\boldsymbol{Q}^\dagger + \boldsymbol{Q}^\top\boldsymbol{Q}^\dagger - \boldsymbol{Q}^{\dagger T}\boldsymbol{Q}^\dagger\|_F)\delta \\
\leq\ & \frac{1}{32}(\|\boldsymbol{P}^\top\Delta\boldsymbol{P} + \Delta\boldsymbol{P}^\top\boldsymbol{P}^\dagger\|_F + \|\boldsymbol{Q}^\top\Delta\boldsymbol{Q} + \Delta\boldsymbol{Q}^\top\boldsymbol{Q}^\dagger\|_F)\delta \\
\leq\ & \frac{1}{32}((\|\boldsymbol{P}\|_2 + \|\boldsymbol{P}^\dagger\|_2)\|\Delta\boldsymbol{P}\|_F + (\|\boldsymbol{Q}\|_2 + \|\boldsymbol{Q}^\dagger\|_2)\|\Delta\boldsymbol{Q}\|_F)\delta \\
\leq\ & \frac{1}{32}(\sqrt{\sigma_1^*} + \sqrt{\frac{3\sigma_1^*}{2}})(\|\Delta\boldsymbol{P}\|_F + \|\Delta\boldsymbol{Q}\|_F)\delta \\
\leq\ & \frac{\sqrt{2}+\sqrt{3}}{32}\sqrt{\sigma_1^*\delta^3}.
\end{aligned}
$$

Following [138] lemma 3, we have

$$
\frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 \geq \frac{1}{64}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 + \frac{1}{64}(\sqrt{2}\|\Delta\boldsymbol{F}\boldsymbol{F}^{\dagger T}\|_F - \delta)^2 - \frac{1}{16}\|\boldsymbol{L} - \boldsymbol{L}^*\|_F^2, \tag{2.71}
$$

where we have used the fact that $-\|\boldsymbol{P}\boldsymbol{Q}^\top - \boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T}\|_F^2 \geq -\|\boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\top\boldsymbol{Y}^\top - \boldsymbol{X}\boldsymbol{P}^\dagger\boldsymbol{Q}^{\dagger T}\boldsymbol{Y}^\top\|_F^2$. We know that $\boldsymbol{F}^\dagger = \begin{bmatrix}\boldsymbol{P}^\dagger \\ \boldsymbol{Q}^\dagger\end{bmatrix} = \begin{bmatrix}\boldsymbol{X}^\top\boldsymbol{U}^* \\ \boldsymbol{Y}^\top\boldsymbol{V}^*\end{bmatrix}\boldsymbol{\Sigma}^{*\frac{1}{2}}\boldsymbol{R}$. If we let $\boldsymbol{E} = \begin{bmatrix}\boldsymbol{X}^\top\boldsymbol{U}^* \\ \boldsymbol{Y}^\top\boldsymbol{V}^*\end{bmatrix}$, then $\boldsymbol{E}^\top\boldsymbol{E} = [\boldsymbol{U}^{*T}X \ \ \boldsymbol{V}^{*T}Y]\begin{bmatrix}\boldsymbol{X}^\top\boldsymbol{U}^* \\ \boldsymbol{Y}^\top\boldsymbol{V}^*\end{bmatrix} = 2\boldsymbol{I} \in \mathbb{R}^{r\times r}$. So

$$
\boldsymbol{F}^\dagger = (\frac{\sqrt{2}}{2}\begin{bmatrix}\boldsymbol{X}^\top\boldsymbol{U}^* \\ \boldsymbol{Y}^\top\boldsymbol{V}^*\end{bmatrix})(\sqrt{2}\boldsymbol{\Sigma}^{*\frac{1}{2}})\boldsymbol{R}, \tag{2.72}
$$

is the SVD of $\boldsymbol{F}^\dagger$. Therefore,

$$
\frac{1}{32}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 \geq \frac{1}{64}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 + \frac{1}{64}(2\sqrt{\sigma_r^*\delta} - \delta)^2 - \frac{1}{16}\|\boldsymbol{L} - \boldsymbol{L}^*\|_F^2. \tag{2.73}
$$

Thus, altogether we have

$$
\begin{aligned}
\langle\nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{P} - \boldsymbol{P}^\dagger\rangle + \langle\nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q}), \boldsymbol{Q} - \boldsymbol{Q}^\dagger\rangle &\geq \frac{1}{64}\|\boldsymbol{P}^\top\boldsymbol{P} - \boldsymbol{Q}^\top\boldsymbol{Q}\|_F^2 \\
&+ \frac{1}{64}(2\sqrt{\sigma_r^*\delta} - \delta)^2 - \frac{1}{16}\|\boldsymbol{L} - \boldsymbol{L}^*\|_F^2 - \frac{\sqrt{2}+\sqrt{3}}{32}\sqrt{\sigma_1^*\delta^3}.
\end{aligned} \tag{2.74}
$$

□

Lemma 2.7.13. *When* $\boldsymbol{S} = \mathcal{T}_{\alpha+\min(10\alpha,0.1)}(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{P}\boldsymbol{Q}^\intercal\boldsymbol{Y}^\intercal)$*, given that* $\|\boldsymbol{P}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}$ *and* $\|\boldsymbol{Q}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}$*, we have*

$$\|\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\|_F^2 \leq (1+\sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}})^2\|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2, \tag{2.75}$$

$$\|\nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q})\|_F^2 + \|\nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q})\|_F^2 \leq \frac{3\sigma_1^*}{256}\|\boldsymbol{P}^\intercal\boldsymbol{P}-\boldsymbol{Q}^\intercal\boldsymbol{Q}\|_F^2. \tag{2.76}$$

Proof.

$$\begin{aligned}
&\|\nabla_{\boldsymbol{P}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q})\|_F^2 + \|\nabla_{\boldsymbol{Q}}\mathcal{G}(\boldsymbol{P},\boldsymbol{Q})\|_F^2\\
=&\|\frac{1}{16}\boldsymbol{P}(\boldsymbol{P}^\intercal\boldsymbol{P}-\boldsymbol{Q}^\intercal\boldsymbol{Q})\|_F^2 + \|\frac{1}{16}\boldsymbol{Q}(\boldsymbol{Q}^\intercal\boldsymbol{Q}-\boldsymbol{P}^\intercal\boldsymbol{P})\|_F^2\\
\leq&\frac{1}{256}(\|\boldsymbol{P}\|_2^2+\|\boldsymbol{Q}\|_2^2)\|\boldsymbol{P}^\intercal\boldsymbol{P}-\boldsymbol{Q}^\intercal\boldsymbol{Q}\|_F^2\\
\leq&\frac{1}{256}(\frac{3\sigma_1^*}{2}+\frac{3\sigma_1^*}{2})\|\boldsymbol{P}^\intercal\boldsymbol{P}-\boldsymbol{Q}^\intercal\boldsymbol{Q}\|_F^2\\
\leq&\frac{3\sigma_1^*}{256}\|\boldsymbol{P}^\intercal\boldsymbol{P}-\boldsymbol{Q}^\intercal\boldsymbol{Q}\|_F^2.
\end{aligned} \tag{2.77}$$

From **Lemma 2.7.11**, we have

$$\|\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\|_F \leq (1+\sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}})\|\boldsymbol{L}-\boldsymbol{L}^*\|_F, \tag{2.78}$$

so

$$\|\nabla_{\boldsymbol{L}}\mathcal{H}(\boldsymbol{P},\boldsymbol{Q})\|_F^2 \leq (1+\sqrt{\frac{2\alpha}{\min(10\alpha,0.1)}})^2\|\boldsymbol{L}-\boldsymbol{L}^*\|_F^2. \tag{2.79}$$

□

We conduct the proof of **Theorem 2** by induction.

If α is small, then from **Theorem 1** we have $\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0\boldsymbol{V}_0^\intercal - \boldsymbol{L}^*\|_2 \leq \frac{1}{2}\sigma_1^*$. By Weyl's theorem, we have

$$\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}, \tag{2.80}$$

$$\|\boldsymbol{V}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}, \tag{2.81}$$

and

$$\|\boldsymbol{P}_0\|_2 \leq \|\boldsymbol{X}^\intercal\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \|\boldsymbol{X}\|_2\|\boldsymbol{U}_0\boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}, \tag{2.82}$$

$$\|\boldsymbol{Q}_0\|_2 \leq \|\boldsymbol{Y}^\intercal \boldsymbol{V}_0 \boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \|\boldsymbol{Y}\|_2 \|\boldsymbol{V}_0 \boldsymbol{\Sigma}_0^{\frac{1}{2}}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}. \tag{2.83}$$

In case (i), we thus have

$$\|\boldsymbol{X}\boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P}_0)\|_{2,\infty} \leq \sqrt{\frac{2\mu_1 r}{n}} \|\boldsymbol{P}_0\|_2 \leq \sqrt{\frac{3\sigma_1^* \mu_1 r}{n}}, \tag{2.84}$$

$$\|\boldsymbol{Y}\boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q}_0)\|_{2,\infty} \leq \sqrt{\frac{2\mu_1 r}{n}} \|\boldsymbol{Q}_0\|_2 \leq \sqrt{\frac{3\sigma_1^* \mu_1 r}{n}}. \tag{2.85}$$

And it also follows that $d(\boldsymbol{\Pi}_{\mathcal{P}}(\boldsymbol{P}_t), \boldsymbol{\Pi}_{\mathcal{Q}}(\boldsymbol{Q}_t), \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq d(\boldsymbol{P}_t, \boldsymbol{Q}_t, \boldsymbol{P}^*, \boldsymbol{Q}^*)$.

By definition,

$$\begin{aligned} \|\boldsymbol{P} - \boldsymbol{P}^\dagger\|_F^2 &\leq \delta, \\ \|\boldsymbol{P} - \boldsymbol{P}^\dagger\|_2 \leq \|\boldsymbol{P} - \boldsymbol{P}^\dagger\|_F &\leq \delta^{\frac{1}{2}}. \end{aligned} \tag{2.86}$$

And from Weyl's theorem, if $\delta^{\frac{1}{2}} \leq (\sqrt{\frac{3}{2}} - 1)\sqrt{\sigma_1^*}$, we have

$$\|\boldsymbol{P}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}. \tag{2.87}$$

Similarly, we also have

$$\|\boldsymbol{Q}\|_2 \leq \sqrt{\frac{3\sigma_1^*}{2}}. \tag{2.88}$$

In cases (ii) and (iii), we have

$$\|\boldsymbol{X}\boldsymbol{P}\|_{2,\infty} \leq \|\boldsymbol{P}\|_2 \|\boldsymbol{X}\|_{2,\infty} \leq \sqrt{\frac{3\sigma_1^*}{2}} \times \sqrt{\frac{\mu_2 d}{n}} \leq \sqrt{\frac{3d\mu\sigma_1^*}{2n}}, \tag{2.89}$$

$$\|\boldsymbol{Y}\boldsymbol{Q}\|_{2,\infty} \leq \|\boldsymbol{Q}\|_2 \|\boldsymbol{Y}\|_{2,\infty} \leq \sqrt{\frac{3\sigma_1^*}{2}} \times \sqrt{\frac{\mu_2 d}{n}} \leq \sqrt{\frac{3d\mu\sigma_1^*}{2n}}. \tag{2.90}$$

Now, we verify that δ diminishes:

$$
\begin{aligned}
\delta_{t+1} &= \|\boldsymbol{P}_{t+1} - \boldsymbol{P}^{\dagger}_{t+1}\|_F^2 + \|\boldsymbol{Q}_{t+1} - \boldsymbol{Q}^{\dagger}_{t+1}\|_F^2 \\
&\leq \|\boldsymbol{P}_{t+1} - \boldsymbol{P}^{\dagger}_{t}\|_F^2 + \|\boldsymbol{Q}_{t+1} - \boldsymbol{Q}^{\dagger}_{t}\|_F^2 \\
&= \|\boldsymbol{P}_t - \eta\nabla_{\boldsymbol{P}}\mathcal{H}_t - \eta\nabla_{\boldsymbol{P}}\mathcal{G}_t - \boldsymbol{P}^{\dagger}_{t}\|_F^2 + \|\boldsymbol{Q}_t - \eta\nabla_{\boldsymbol{Q}}\mathcal{H}_t - \eta\nabla_{\boldsymbol{Q}}\mathcal{G}_t - \boldsymbol{Q}^{\dagger}_{t}\|_F^2 \\
&= \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{H}_t + \nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{H}_t + \nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad + \eta^2\|\nabla_{\boldsymbol{P}}\mathcal{H}_t + \nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2 + \eta^2\|\nabla_{\boldsymbol{Q}}\mathcal{H}_t + \nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 \\
&= \delta_t + \eta^2\|\nabla_{\boldsymbol{P}}\mathcal{H}_t + \nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2 + \eta^2\|\nabla_{\boldsymbol{Q}}\mathcal{H}_t + \nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + \eta^2(\|\nabla_{\boldsymbol{P}}\mathcal{H}_t\|_F + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F)^2 + \eta^2(\|\nabla_{\boldsymbol{Q}}\mathcal{H}_t\|_F + \|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F)^2 \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + 2\eta^2(\|\boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}_t\boldsymbol{Y}\boldsymbol{Q}\|_F^2 + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2 + \|(\boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}_t\boldsymbol{Y})^{\intercal}\boldsymbol{P}\|_F^2 + \|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2) \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + 2\eta^2(\|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2) \\
&\quad + 2\eta^2(\|\boldsymbol{Q}\|_2^2\|\boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}_t\boldsymbol{Y}\|_F^2 + \|\boldsymbol{P}\|_2^2\|(\boldsymbol{X}^{\intercal}\nabla_{\boldsymbol{L}}\mathcal{H}_t\boldsymbol{Y})^{\intercal}\|_F^2) \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + 2\eta^2(\|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2) \\
&\quad + 2\eta^2(\|\boldsymbol{Q}\|_2^2\|\boldsymbol{X}\|_2^2\|\boldsymbol{Y}\|_2^2\|\nabla_{\boldsymbol{L}}\mathcal{H}_t\|_F^2 + \|\boldsymbol{P}\|_2^2\|\boldsymbol{X}\|_2^2\|\boldsymbol{Y}\|_2^2\|\nabla_{\boldsymbol{L}}\mathcal{H}_t\|_F^2) \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + 2\eta^2(\|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2) \\
&\quad + 2\eta^2(\frac{3\sigma_1^*}{2}\|\nabla_{\boldsymbol{L}}\mathcal{H}_t\|_F^2 + \frac{3\sigma_1^*}{2}\|\nabla_{\boldsymbol{L}}\mathcal{H}_t\|_F^2) \\
&\leq \delta_t - 2\eta\langle\nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}^{\dagger}_t\rangle - 2\eta\langle\nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}^{\dagger}_t\rangle \\
&\quad - 2\eta\langle\nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\
&\quad + 2\eta^2(\|\nabla_{\boldsymbol{Q}}\mathcal{G}_t\|_F^2 + \|\nabla_{\boldsymbol{P}}\mathcal{G}_t\|_F^2 + 3\sigma_1^*\|\nabla_{\boldsymbol{L}}\mathcal{H}_t\|_F^2).
\end{aligned}
\tag{2.91}
$$

Applying **Lemma 2.7.13**, we get

$$\begin{aligned}\delta_{t+1} \leq \delta_t &- 2\eta\langle \nabla_{\boldsymbol{P}}\mathcal{G}_t, \boldsymbol{P}_t - \boldsymbol{P}_t^{\dagger}\rangle - 2\eta\langle \nabla_{\boldsymbol{Q}}\mathcal{G}_t, \boldsymbol{Q}_t - \boldsymbol{Q}_t^{\dagger}\rangle \\ &- 2\eta\langle \nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\ &+ \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{\frac{2\alpha}{\min(10\alpha, 0.1)}})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).\end{aligned} \tag{2.92}$$

Applying **Lemma 2.7.12**, we get

$$\begin{aligned}\delta_{t+1} \leq \delta_t &+ \eta(\frac{1}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2 - \frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2) \\ &+ \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2) \\ &- 2\eta\langle \nabla_{\boldsymbol{L}}\mathcal{H}_t, \boldsymbol{X}(\boldsymbol{P}_t\boldsymbol{Q}_t^{\intercal} - \boldsymbol{P}_t^{\dagger}\boldsymbol{Q}_t^{\dagger T} + \Delta\boldsymbol{P}_t\Delta\boldsymbol{Q}_t^{\intercal})\boldsymbol{Y}^{\intercal}\rangle \\ &+ \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{\frac{2\alpha}{\min(10\alpha, 0.1)}})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).\end{aligned} \tag{2.93}$$

Applying **Lemma 2.7.11**, we have in case (i)

$$\begin{aligned}&\delta_{t+1} \\ &\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2) \\ &\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + 2\sqrt{\frac{\alpha}{\min(10\alpha, 0.1)}})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \\ &\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 2\min(10\alpha, 0.1))(3+\sqrt{\frac{3}{2}})^2 + \frac{4\alpha\delta_t}{\beta\min(10\alpha, 0.1)}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\ &\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{\frac{2\alpha}{\min(10\alpha, 0.1)}})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2),\end{aligned} \tag{2.94}$$

and in cases (ii) and (iii)

$$\begin{aligned}&\delta_{t+1} \\ &\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2) \\ &\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + 2\sqrt{\frac{\alpha}{\min(10\alpha, 0.1)}})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \\ &\quad + \eta(\frac{\mu_2 d\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 2\min(10\alpha, 0.1))(3+\sqrt{\frac{3}{2}})^2 + \frac{4\alpha\delta_t}{\beta\min(10\alpha, 0.1)}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\ &\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{\frac{2\alpha}{\min(10\alpha, 0.1)}})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2),\end{aligned} \tag{2.95}$$

If $10\alpha < 0.1$, then $\min(10\alpha, 0.1) = 10\alpha$.

Therefore, we have in case (i)

$$\begin{aligned}
&\delta_{t+1}\\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2)\\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + \sqrt{\frac{2}{5}})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4}))\\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 20\alpha)(3+\sqrt{\frac{3}{2}})^2 + \frac{2\delta_t}{5\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2)\\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{\frac{1}{5}})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2) \qquad (2.96)\\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2)\\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + 2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4}))\\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t}{2}((24+\beta)\alpha)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2)\\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{20})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2),
\end{aligned}$$

and in cases (ii) and (iii)

$$\begin{aligned}
&\delta_{t+1}\\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2)\\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + 2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \qquad (2.97)\\
&\quad + \eta(\frac{\mu_2 d\sigma_1^*\delta_t}{2}((24+\beta)\alpha)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2)\\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{20})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).
\end{aligned}$$

On the other hand, we have $\min(10\alpha, 0.1) = 0.1$ if $10\alpha \geq 0.1$.

Then, we have in case (i)

$$\begin{aligned}
&\delta_{t+1}\\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2)\\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2} + 2\sqrt{10\alpha})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \qquad (2.98)\\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 0.2)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\alpha\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2)\\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\intercal}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\intercal}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{20\alpha})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).
\end{aligned}$$

But $\alpha \leq 1$, so

$$\begin{aligned}
&\delta_{t+1} \\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\mathsf{T}}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\mathsf{T}}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2) \\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2}+2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 20\alpha)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\mathsf{T}}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\mathsf{T}}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{20})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).
\end{aligned} \tag{2.99}$$

And, similarly, we have in cases (ii) and (iii)

$$\begin{aligned}
&\delta_{t+1} \\
&\leq \delta_t - \eta(\frac{1}{32}\|\boldsymbol{P}_t^{\mathsf{T}}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\mathsf{T}}\boldsymbol{Q}_t\|_F^2 + \frac{15}{8}\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2) \\
&\quad + \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2}+2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})) \\
&\quad + \eta(\frac{\mu_2 d\sigma_1^*\delta_t}{2}((4+\beta)\alpha + 20\alpha)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\
&\quad + \eta^2(\frac{3\sigma_1^*}{128}\|\boldsymbol{P}_t^{\mathsf{T}}\boldsymbol{P}_t - \boldsymbol{Q}_t^{\mathsf{T}}\boldsymbol{Q}_t\|_F^2 + 6(1+\sqrt{20})^2\sigma_1^*\|\boldsymbol{L}_t - \boldsymbol{L}_t^*\|_F^2).
\end{aligned} \tag{2.100}$$

If $\eta \leq \frac{5}{16(1+\sqrt{20})^2\sigma_1^*}$, we have in case (i)

$$\begin{aligned}
\delta_{t+1} \leq \delta_t &+ \eta(\frac{\mu_1 r\alpha\sigma_1^*\delta_t}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\
&+ \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2}+2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})).
\end{aligned} \tag{2.101}$$

and in cases (ii) and (iii)

$$\begin{aligned}
\delta_{t+1} \leq \delta_t &+ \eta(\frac{\mu_2 d\alpha\sigma_1^*\delta_t}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{40\delta_t}{\beta}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})^2) \\
&+ \eta(\frac{\sqrt{2}+\sqrt{3}}{16}\sqrt{\sigma_1^*\delta_t^3} - \frac{1}{32}(2\sqrt{\sigma_r^*\delta_t} - \delta_t)^2 + (\sqrt{2}+2\sqrt{10})\sqrt{\delta_t^3}(\sqrt{\sigma_1^*} + \frac{\sqrt{2\delta_t}}{4})).
\end{aligned} \tag{2.102}$$

If $\delta_t \leq 2\sigma_r^*$, we have in case (i)

$$\begin{aligned}
\delta_{t+1} &\leq \delta_t + \eta(\frac{25\sqrt{2}+\sqrt{3}+48\sqrt{10}}{16}\delta_t\sqrt{\sigma_1^*\delta_t} - \frac{3-\sqrt{2}}{16}\sigma_r^*\delta_t) \\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\delta_t\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{90\delta_t\sigma_1^*}{\beta}) \\
&\leq \delta_t(1 + \eta(\frac{25\sqrt{2}+\sqrt{3}+48\sqrt{10}}{16}\sqrt{\sigma_1^*\delta_t} - \frac{3-\sqrt{2}}{16}\sigma_r^*) \\
&\quad + \eta(\frac{\mu_1 r\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{90\sigma_1^*}{\beta})),
\end{aligned} \tag{2.103}$$

and in cases (ii) and (iii)

$$\begin{aligned}\delta_{t+1} \leq \delta_t(1+\eta(\frac{25\sqrt{2}+\sqrt{3}+48\sqrt{10}}{16}\sqrt{\sigma_1^*\delta_t}-\frac{3-\sqrt{2}}{16}\sigma_r^*)\\ +\eta(\frac{\mu_2 d\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2+\frac{90\sigma_1^*}{\beta})),\end{aligned} \tag{2.104}$$

In case (i), if $\alpha \leq \frac{1}{16\kappa r\mu_1}$, we have

$$\delta^{\frac{1}{2}} = d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\alpha r\mu_1\sqrt{r\kappa\sigma_1^*}, \tag{2.105}$$

which leads to

$$\begin{aligned}\delta_{t+1} \leq \delta_t(1+\eta(\frac{\mu_1 r\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2-\frac{3-\sqrt{2}}{16}\sigma_r^*)\\ +\eta(\frac{90\sigma_1^*}{\beta}+\frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha r\mu_1\sigma_1^*\sqrt{r\kappa})),\end{aligned} \tag{2.106}$$

In case (ii), if $\alpha \leq \frac{1}{16\kappa d\mu_2}$, we have

$$\delta^{\frac{1}{2}} = d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\alpha d\mu_2\sqrt{r\kappa\sigma_1^*}, \tag{2.107}$$

which leads to

$$\begin{aligned}\delta_{t+1} \leq \delta_t(1+\eta(\frac{\mu_2 d\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2-\frac{3-\sqrt{2}}{16}\sigma_r^*)\\ +\eta(\frac{90\sigma_1^*}{\beta}+\frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha d\mu_2\sigma_1^*\sqrt{r\kappa})),\end{aligned} \tag{2.108}$$

In case (iii), if $\alpha \leq \frac{1}{16\kappa r\mu_1}$, we have

$$\delta^{\frac{1}{2}} = d(\boldsymbol{P}_0, \boldsymbol{Q}_0, \boldsymbol{P}^*, \boldsymbol{Q}^*) \leq 18\alpha r\mu_1\sqrt{r\kappa\sigma_1^*}, \tag{2.109}$$

which leads to

$$\begin{aligned}\delta_{t+1} \leq \delta_t(1+\eta(\frac{\mu_2 d\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2-\frac{3-\sqrt{2}}{16}\sigma_r^*)\\ +\eta(\frac{90\sigma_1^*}{\beta}+\frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha r\mu_1\sigma_1^*\sqrt{r\kappa})),\end{aligned} \tag{2.110}$$

In case (i), we require that

$$\begin{aligned}\frac{\mu_1 r\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2-\frac{3-\sqrt{2}}{16}\sigma_r^*+\frac{90\sigma_1^*}{\beta}\\ +\frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha r\mu_1\sigma_1^*\sqrt{r\kappa} \leq 0,\end{aligned} \tag{2.111}$$

which leads to

$$\alpha \leq \frac{\frac{3-\sqrt{2}}{16} + \frac{90\kappa}{\beta}}{\frac{\mu_1 r\kappa}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\kappa r\mu_1\sqrt{r\kappa}}. \tag{2.112}$$

Since other constraints on α are milder, for β large enough, there exist c_1 and c_2 such that if $\alpha \leq \frac{c_1}{\mu_1(\kappa r)^{\frac{3}{2}}}$,

$$\delta_t \leq (1 - c_2\eta\sigma_r^*)^t\delta_0. \tag{2.113}$$

In case (ii), we require that

$$\frac{\mu_2 d\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 - \frac{3-\sqrt{2}}{16}\sigma_r^* + \frac{90\sigma_1^*}{\beta} + \frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha d\mu_2\sigma_1^*\sqrt{r\kappa} \leq 0, \tag{2.114}$$

which leads to

$$\alpha \leq \frac{\frac{3-\sqrt{2}}{16} + \frac{90\kappa}{\beta}}{\frac{\mu_2 d\kappa}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\kappa d\mu_2\sqrt{r\kappa}}. \tag{2.115}$$

Since other constraints on α are milder, for β large enough, there exist c_1 and c_2 such that if $\alpha \leq \frac{c_3}{\mu_2 d r^{\frac{1}{2}}\kappa^{\frac{3}{2}}}$,

$$\delta_t \leq (1 - c_4\eta\sigma_r^*)^t\delta_0. \tag{2.116}$$

In case (iii), we require that

$$\frac{\mu_2 d\sigma_1^*\alpha}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 - \frac{3-\sqrt{2}}{16}\sigma_r^* + \frac{90\sigma_1^*}{\beta} + \frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\alpha r\mu_1\sigma_1^*\sqrt{r\kappa} \leq 0, \tag{2.117}$$

which leads to

$$\alpha \leq \frac{\frac{3-\sqrt{2}}{16} + \frac{90\kappa}{\beta}}{\frac{\mu_2 d\kappa}{2}(24+\beta)(3+\sqrt{\frac{3}{2}})^2 + \frac{225\sqrt{2}+9\sqrt{3}+432\sqrt{10}}{8}\kappa r\mu_1\sqrt{r\kappa}}. \tag{2.118}$$

Since other constraints on α are milder, for β large enough, there exist c_5 and c_6 such that if $\alpha \leq c_5\min(\frac{1}{\mu_2 d\kappa}, \frac{1}{\mu_1(\kappa r)^{\frac{3}{2}}})$,

$$\delta_t \leq (1 - c_6\eta\sigma_r^*)^t\delta_0. \tag{2.119}$$

CHAPTER 3

Side Information in Robust Principal Component Analysis: Algorithms and Applications

3.1 Introduction

In this chapter, we investigate the idea of using a *noisy* approximation of the low-rank component to guide PCP. Knowledge regarding the low-rank component, albeit noisy, is available in many applications. In background subtraction, we may find some frames of the video that do not contain changes and therefore may be used to accurately estimate the background. Another example concerns the problem of disentangling identity and expression components in expressive faces, where the low-rank component is roughly similar to the neutral face. Note that side information which has the same form as the source is already subject to wide-spread usage. Watermark detection methods require a reference image to identify ownership [35]. Automated photo tagging explores visually similar social images [127]. Locality preserving projection can be enhanced by exploiting similar pairs of patterns [3]. Spatial and temporal correlation can improve signal recovery algorithms in compressive imaging [114]. In content-based image retrieval, historical feedback log data can help retrieve semantically relevant images [140]. Low-resolution images can help adapt a high-resolution compressive sensing system [121]. Near-accurate fingerprint or DNA can be used as side information to hack a biometric authentication system [57].

Our contributions are summarised as follows:

- A novel convex program is proposed to use side information, which is a noisy approximation of the low-rank component, within the PCP framework with a provably convergent solver.

- Furthermore, we extend our proposed PCP model using side information to exploit prior knowledge regarding the column and row spaces of the low-rank component in a more general algorithmic framework.
- We demonstrate the applicability and effectiveness of the proposed approaches in several applications, namely background subtraction, facial image denoising as well as face recognition and facial expression classification.
- We also show that our proposed methods can mitigate the transductive constraint of RPCA. With side information, training can be performed on fewer samples and hence reducing the computational cost.

3.2 Models

In this section, the proposed RPCA models with side information are introduced. In particular, we propose to incorporate the side information into PCP by using the trace distance of the difference between the low-rank component and the noisy estimate, which is reasonable if their difference is of low rank. However, we show empirically (Section 3.3) that it also works if the difference is full-rank. This may be attributed to the fact that the trace distance is a natural distance metric between two dissimilar distributions from Kolmogorov–Smirnov statistics [82]. Besides that, this is a generalisation of the compressed sensing with side information where the l_1 norm has been used in order to measure the distance of the target signal with prior information [79].

3.2.1 The PCPS model

Assuming that a noisy estimate of the low-rank component of the data $\boldsymbol{W} \in \mathbb{R}^{n_1 \times n_2}$ is available, we propose the following model of PCP using side information (PCPS):

$$\begin{aligned} &\underset{\boldsymbol{L},\boldsymbol{S}}{\text{minimize}} \quad \|\boldsymbol{L}\|_* + \kappa\|\boldsymbol{L} - \boldsymbol{W}\|_* + \lambda\|\boldsymbol{S}\|_1 \\ &\text{s. t. to} \quad \boldsymbol{L} + \boldsymbol{S} = \boldsymbol{M}, \end{aligned} \tag{3.1}$$

where $\kappa > 0, \lambda > 0$ are parameters that weigh the effects of side information and noise sparsity.

The proposed PCPS can be revamped to generalise the previous attempt of PCPF by the following objective of PCPS with features (PCPSF):

$$\begin{aligned} &\underset{\boldsymbol{H},\boldsymbol{S}}{\text{minimize}} \quad \|\boldsymbol{H}\|_* + \kappa\|\boldsymbol{H} - \boldsymbol{D}\|_* + \lambda\|\boldsymbol{S}\|_1 \\ &\text{s. t. to} \quad \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \boldsymbol{S} = \boldsymbol{M}, \quad \boldsymbol{X}\boldsymbol{W}\boldsymbol{Y}^\intercal = \boldsymbol{D}, \end{aligned} \tag{3.2}$$

where $\boldsymbol{H} \in \mathbb{R}^{d_1 \times d_2}, \boldsymbol{D} \in \mathbb{R}^{d_1 \times d_2}$ are bilinear mappings for the recovered low-rank matrix $\boldsymbol{L}$ and side information $\boldsymbol{W}$ respectively. If $\boldsymbol{W} = \boldsymbol{L} + \boldsymbol{C}$, then $\boldsymbol{D} = \boldsymbol{X}^\intercal(\boldsymbol{L} + \boldsymbol{C})\boldsymbol{Y} = \boldsymbol{L} + \boldsymbol{X}^\intercal\boldsymbol{C}\boldsymbol{Y}$. That is, we have reduced the noise onto a smaller region $\mathbb{R}^{d \times d}$ rather than $\mathbb{R}^{n \times n}$ which has made the problem easier to solve. Note that the low-rank matrix $\boldsymbol{L}$ is recovered from the optimal solution $(\boldsymbol{H}^*, \boldsymbol{S}^*)$ to objective (3.2) via $\boldsymbol{L} = \boldsymbol{X}\boldsymbol{H}^*\boldsymbol{Y}^\intercal$. If side information $\boldsymbol{W}$ is not available, PCPSF reduces to PCPF by setting κ to zero. If the features $\boldsymbol{X}, \boldsymbol{Y}$ are not present either, PCP can be restored by fixing both of them at identity. However, when only the side information $\boldsymbol{W}$ is accessible, objective (3.2) is transformed back into PCPS.

3.2.2 The algorithm

If we substitute $\boldsymbol{E}$ for $\boldsymbol{H} - \boldsymbol{D}$ and orthogonalise $\boldsymbol{X}$ and $\boldsymbol{Y}$, the optimisation problem (3.2) is identical to the following convex but non-smooth problem:

$$\begin{aligned} &\underset{\boldsymbol{H},\boldsymbol{S}}{\text{minimize}} \quad \|\boldsymbol{H}\|_* + \kappa\|\boldsymbol{E}\|_* + \lambda\|\boldsymbol{S}\|_1 \\ &\text{s.t. to} \quad \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \boldsymbol{S} = \boldsymbol{M}, \quad \boldsymbol{E} - \boldsymbol{H} = -\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}, \end{aligned} \tag{3.3}$$

which is amenable to the multi-block alternating direction method of multipliers (ADMM).

The corresponding augmented Lagrangian of (3.3) is:

$$\begin{aligned} \mathcal{L}(\boldsymbol{H}, \boldsymbol{E}, \boldsymbol{S}, \boldsymbol{Z}, \boldsymbol{N}) &= \|\boldsymbol{H}\|_* + \kappa\|\boldsymbol{E}\|_* + \lambda\|\boldsymbol{S}\|_1 \\ &+ \langle \boldsymbol{Z}, \boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal \rangle + \frac{\mu}{2}\|\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal\|_F^2 \\ &+ \langle \boldsymbol{N}, \boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y} \rangle + \frac{\mu}{2}\|\boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}\|_F^2, \end{aligned} \tag{3.4}$$

where $\boldsymbol{Z} \in \mathbb{R}^{n_1 \times n_2}$ and $\boldsymbol{N} \in \mathbb{R}^{d_1 \times d_2}$ are Lagrange multipliers and μ is the learning rate.

The ADMM operates by carrying out repeated cycles of updates till convergence. During each cycle, $\boldsymbol{H}, \boldsymbol{E}, \boldsymbol{S}$ are updated serially by minimising (3.4) with other variables fixed. Afterwards, Lagrange multipliers $\boldsymbol{Z}, \boldsymbol{N}$ are updated at the end of each iteration. Direct solutions to the single variable minimisation subproblems rely on the shrinkage and the singular value thresholding operators [21]. Let $\mathcal{S}_\tau(a) \equiv \operatorname{sgn}(a)\max(|a| - \tau, 0)$ serve as the shrinkage operator, which naturally extends to matrices, $\mathcal{S}_\tau(\boldsymbol{A})$, by applying it to matrix $\boldsymbol{A}$ element-wise. Similarly, let $\mathcal{D}_\tau(\boldsymbol{A}) \equiv \boldsymbol{U}\mathcal{S}_\tau(\boldsymbol{\Sigma})\boldsymbol{V}^\intercal$ be the singular value thresholding operator on real matrix $\boldsymbol{A}$, with $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\intercal$ being the singular value decomposition (SVD) of $\boldsymbol{A}$.

Minimising (3.4) w.r.t. $\boldsymbol{H}$ at fixed $\boldsymbol{E}, \boldsymbol{S}, \boldsymbol{Z}, \boldsymbol{N}$ is equivalent to the following:

$$\arg\min_{\boldsymbol{H}} \ \|\boldsymbol{H}\|_* + \mu\|\boldsymbol{P} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal\|_F^2, \tag{3.5}$$

Algorithm 5 ADMM solver for PCPSF

```
Input: Observation $\boldsymbol{M}$, side information $\boldsymbol{W}$, features $\boldsymbol{X}, \boldsymbol{Y}$, parameters $\kappa, \lambda > 0$, scaling ratio $\alpha > 1$.
1: Initialize: $\boldsymbol{Z} = \boldsymbol{0}$, $\boldsymbol{N} = \boldsymbol{E} = \boldsymbol{H} = \boldsymbol{0}$, $\mu = \frac{1}{\|\boldsymbol{M}\|_2}$.
2: while not converged do
3:     $\boldsymbol{S} = \mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \frac{1}{\mu}\boldsymbol{Z})$
4:     $\boldsymbol{H} = \boldsymbol{X}^\intercal \mathcal{D}_{\frac{1}{2\mu}}(\frac{1}{2}(\boldsymbol{M} - \boldsymbol{S} + \boldsymbol{W} + \frac{1}{\mu}\boldsymbol{Z} + \boldsymbol{X}(\boldsymbol{E} - \frac{1}{\mu}\boldsymbol{N})\boldsymbol{Y}^\intercal))\boldsymbol{Y}$
5:     $\boldsymbol{E} = \mathcal{D}_{\kappa\mu^{-1}}(\boldsymbol{H} - \boldsymbol{X}^\intercal \boldsymbol{W}\boldsymbol{Y} + \frac{1}{\mu}\boldsymbol{N})$
6:     $\boldsymbol{Z} = \boldsymbol{Z} + \mu(\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)$
7:     $\boldsymbol{N} = \boldsymbol{N} + \mu(\boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal \boldsymbol{W}\boldsymbol{Y})$
8:     $\mu = \mu \times \alpha$
9: end while
Return: $\boldsymbol{L} = \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal$, $\boldsymbol{S}$
```

where $\boldsymbol{P} = \frac{1}{2}(\boldsymbol{M} - \boldsymbol{S} + \boldsymbol{W} + \frac{1}{\mu}\boldsymbol{Z} + \boldsymbol{X}(\boldsymbol{E} - \frac{1}{\mu}\boldsymbol{N})\boldsymbol{Y}^\intercal)$. Its solution is shown to be $\boldsymbol{X}^\intercal \mathcal{D}_{\frac{1}{2\mu}}(\boldsymbol{P})Y$. Furthermore, for $\boldsymbol{E}$,

$$\arg\min_{\boldsymbol{E}} \mathcal{L} = \arg\min_{\boldsymbol{E}} \kappa\|\boldsymbol{E}\|_* + \frac{\mu}{2}\|\boldsymbol{Q} - \boldsymbol{E}\|_F^2, \tag{3.6}$$

where $\boldsymbol{Q} = \boldsymbol{H} - \boldsymbol{X}^\intercal \boldsymbol{W}\boldsymbol{Y} + \frac{1}{\mu}N$, whose update rule is $\mathcal{D}_{\frac{\kappa}{\mu}}(\boldsymbol{Q})$, and for $\boldsymbol{S}$,

$$\arg\min_{\boldsymbol{S}} \mathcal{L} = \arg\min_{\boldsymbol{S}} \lambda\|\boldsymbol{S}\|_1 + \frac{\mu}{2}\|\boldsymbol{R} - \boldsymbol{S}\|_F^2, \tag{3.7}$$

where $\boldsymbol{R} = \boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \frac{1}{\mu}\boldsymbol{Z}$ with a closed-form solution $\mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{R})$. Finally, Lagrange multipliers are updated as usual:

$$\boldsymbol{Z} = \boldsymbol{Z} + \mu(\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal), \tag{3.8}$$

$$\boldsymbol{N} = \boldsymbol{N} + \mu(\boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal \boldsymbol{W}\boldsymbol{Y}). \tag{3.9}$$

The overall algorithm is summarised in Algorithm 5.

3.2.3 Complexity and convergence

Orthogonalisation of the features $\boldsymbol{X}, \boldsymbol{Y}$ via the Gram-Schmidt process has an operation count of $O(n_1 d_1^2)$ and $O(n_2 d_2^2)$ respectively. The $\boldsymbol{H}$ update in Step 4 is the most costly step of each iteration in Algorithm 5. Specifically, the SVD required in the singular value thresholding action dominates with $O(\min(n_1 n_2^2, n_1^2 n_2))$ complexity.

It has been recently established that for a 3-block separable convex minimisation problem, the direct extension of the ADMM achieves global convergence with linear convergence rate if one block in the objective is sub-strongly monotonic [115]. In our case, it can be shown that $\|\boldsymbol{S}\|_1$ possesses such sub-strong monotonicity. We have also used the fast continuation technique to increase μ incrementally for accelerated superlinear performance. The cold start initialisation strategies for variables $\boldsymbol{H}, \boldsymbol{E}$ and Lagrange multipliers $\boldsymbol{Z}, \boldsymbol{N}$ are described in [17]. Besides, we have scheduled $\boldsymbol{S}$ to be updated first. As for stopping criteria, we have employed the Karush-Kuhn-Tucker (KKT) feasibility conditions. Namely, within a maximum number of 1000 iterations, when the maximum of $\|\boldsymbol{M} - \boldsymbol{S}_k - \boldsymbol{X}\boldsymbol{H}_k\boldsymbol{Y}^\intercal\|_F / \|\boldsymbol{M}\|_F$ and $\|\boldsymbol{H}_k - \boldsymbol{E}_k - \boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}\|_F / \|\boldsymbol{M}\|_F$ dwindles from a pre-defined threshold ϵ, the algorithm is terminated, where k signifies values at the k^{th} iteration.

3.3 Experiments

In this section, we illustrate the enhancement made by side information through both numerical simulations and real-world applications. First, we compare the recoverability of our proposed algorithms with state-of-the-art methods for incorporating features or dictionaries, i.e. PCPF [33] and RAPS [101] on synthetic data as well as the baseline PCP [21] when there are no features available. Second, we show how powerful side information can be for the task of object segmentation in video pre-processing. Third, we demonstrate that side information is instructive in the low-dimensionality face modeling from images of different illuminations. Last, we reveal that the more accurately reconstructed expressions in the light of side information lead to better emotion classification.

For RAPS, clean subspace $\boldsymbol{X}$ is used instead of the observation $\boldsymbol{M}$ itself as the dictionary in LRR [67]. PCP is solved via the inexact ALM and the heuristics for predicting the dimension of principal singular space is not adopted here due to its lack of validity on uncharted real data. We also include Partial Sum of Singular Values (PSSV) [84] in our comparison for its stated advantage in view of the limited number of expression observations available.

3.3.1 Parameter calibration

The process of tuning the algorithmic parameters for various models is described in the appendix. Although theoretical determination of κ and λ is beyond the scope of this research, we nevertheless provide empirical guidance based on extensive experiments. $\lambda = 1/\sqrt{\max(n_1, n_2)}$ for a general matrix of dimension $n_1 \times n_2$ from PCP works well for both of our proposed models. κ depends on the quality of the side information. When the side information is accurate, a large κ should be selected to capitalise upon the side information as much as possible, whereas when the side information is improper, a small κ should be picked to sidestep the dissonance caused by the side information. Here, we have discovered that a κ value of 0.2 works best with synthetic data and a value of 0.5 is suited for public video sequences. It is worth emphasising again that prior knowledge of the structural information about the data yields more appropriate values for κ and λ.

3.3.2 Phase transition on synthetic datasets

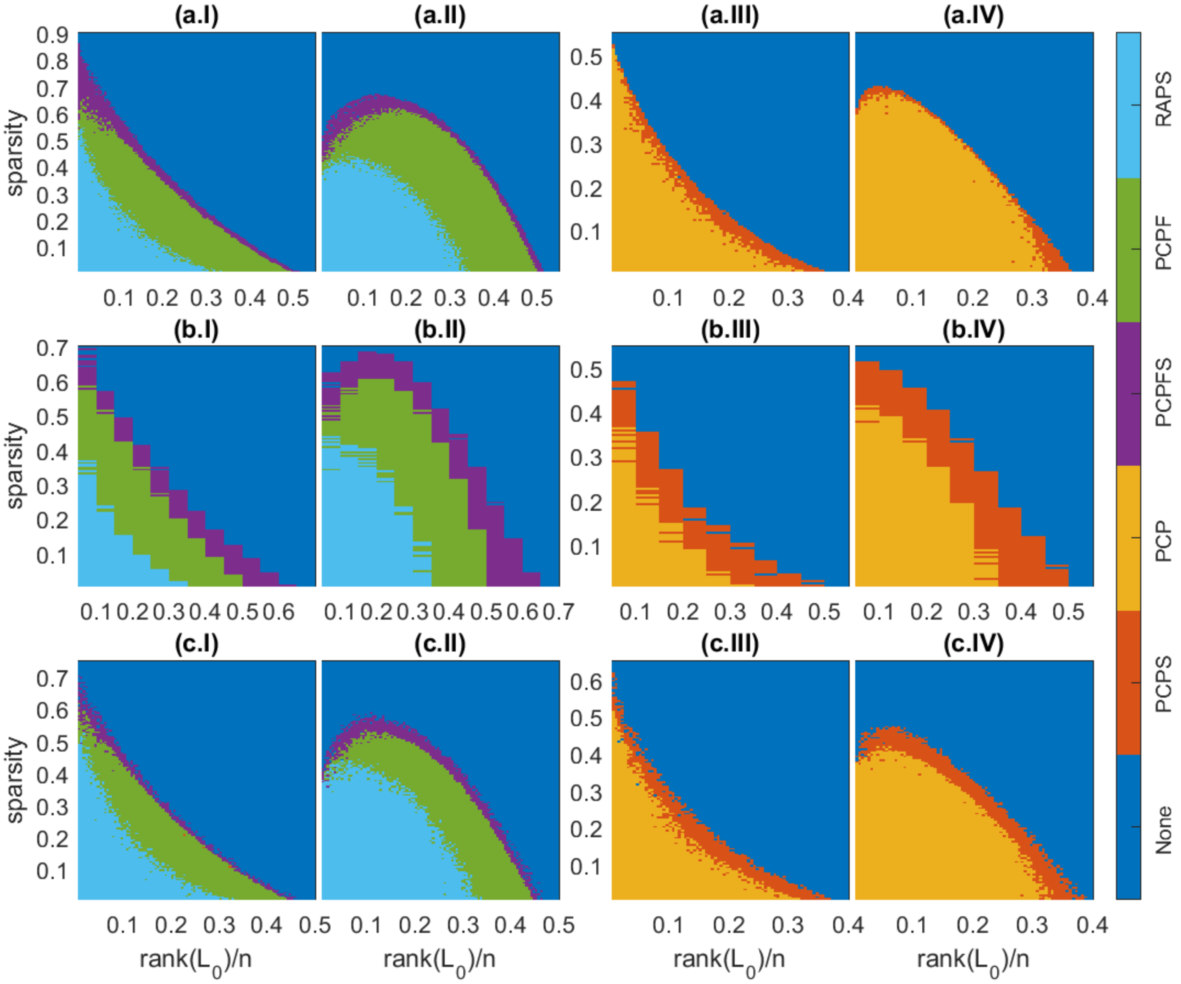


Figure 3.1: Domains of recovery by various algorithms: **(I,III)** for random signs and **(II,IV)** for coherent signs. **(a)** for entry-wise corruptions, **(b)** for deficient ranks and **(c)** for distorted singular values.

We now focus on the recoverability problem, i.e. recovering matrices of varying ranks from errors of varying sparsity. True low-rank matrices are created via $\boldsymbol{L}_0 = \boldsymbol{J}\boldsymbol{K}^\intercal$, where $200 \times r$ matrices $\boldsymbol{J}, \boldsymbol{K}$ have independent elements drawn randomly from a Gaussian distribution of mean 0 and variance $5 \cdot 10^{-3}$ so r is the rank of $\boldsymbol{L}_0$. Next, we generate 200×200 error matrices $\boldsymbol{S}_0$, which possess $\rho_s \cdot 200^2$ non-zero elements located randomly within the matrix. We consider two types of entries for $\boldsymbol{S}_0$: Bernoulli ± 1 and $\mathcal{P}_\Omega(\text{sgn}(\boldsymbol{L}_0))$, where $\mathcal{P}$ is the projection operator and Ω is the support set of $\boldsymbol{S}_0$. $\boldsymbol{M} = \boldsymbol{L}_0 + \boldsymbol{S}_0$ thus becomes the simulated observation. For each (r, ρ_s) pair, three observations are constructed. The recovery is successful if for all these three problems,

$$\frac{\|\boldsymbol{L} - \boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F} < 10^{-3} \tag{3.10}$$

from the recovered $\boldsymbol{L}$. In addition, let $\boldsymbol{L}_0 = \boldsymbol{U\Sigma V}^\intercal$ be the SVD of $\boldsymbol{L}_0$. Feature $\boldsymbol{X}$ is formed by randomly interweaving column vectors of $\boldsymbol{U}$ with d arbitrary orthonormal bases for the null space of $\boldsymbol{U}^\intercal$, while permuting the expanded columns of $\boldsymbol{V}$ with d random orthonormal bases for the kernel of $\boldsymbol{V}^\intercal$ forms feature $\boldsymbol{Y}$. Hence, the feasibility conditions are fulfilled: $\mathbb{C}(\boldsymbol{X}) \supseteq \mathbb{C}(\boldsymbol{L}_0)$, $\mathbb{C}(\boldsymbol{Y}) \supseteq \mathbb{C}(\boldsymbol{L}_0^\intercal)$, where $\mathbb{C}$ is the column space operator.

Entry-wise corruptions. For these trials, we construct the side information by directly adding small Gaussian noise to each element of $\boldsymbol{L}_0$: $L_{ij} \to L_{ij} + \mathcal{N}(0, 2.5r \cdot 10^{-9})$, $i, j = 1, 2, \cdots, 200$. As a result, the standard deviation of the error in each element is 1% of that among the elements themselves. On average, the Frobenius percent error, $\|\boldsymbol{W} - \boldsymbol{L}_0\|_F / \|\boldsymbol{L}_0\|_F$, is 1%. Such side information is genuine in regard to the fact that classical PCA with accurate rank is not able to eliminate the noise [105]. For $d = 10$, Figures 3.1(a.I) and 3.1(a.II) plot results from PCPF, RAPS and PCPSF. On the other hand, the situation with no available features is investigated in Figures 3.1(a.III) and 3.1(a.IV) for PCP and PCPS. The frontier of PCPF has been advanced by PCPSF everywhere for both sign types. Especially at low ranks, errors with much higher density can be removed. Without features, PCPS surpasses PCP by and large with significant expansion at small sparsity for both cases.

Deficient ranks. Now we first make a new matrix $\boldsymbol{\Sigma}'$ by retaining only the singular values from σ_1 to $\sigma_{90\%}$ in $\boldsymbol{\Sigma}$. Then, side information is constructed according to $\boldsymbol{W} = \boldsymbol{U\Sigma}'\boldsymbol{V}^\intercal$, aka hard thresholding. As rank increases, Frobenius percent error of $\boldsymbol{W}$ decreases from 23.3% to 5.8% sublinearly. Figures 3.1(b.I) and 3.1(b.II) show results from PCPF, RAPS and PCPSF where d is again kept at 10. The corresponding cases with no features are presented in Figures 3.1(b.III) and 3.1(b.IV) for PCP and PCPS. Notwithstanding the most spurious side information, PCPSF and PCPS have reclaimed the largest region unattainable by PCPF and PCP respectively for the two signs.

Distorted singular values. Here, we produce the matrix $\boldsymbol{\Sigma}'$ by adding Gaussian noise to singular values in $\boldsymbol{\Sigma}$: $\sigma_i \to \sigma_i + 0.01 \cdot \mathcal{N}(0, \sigma_i^2)$ for all i. Next, side information is formed by $\boldsymbol{W} = \boldsymbol{U\Sigma}'\boldsymbol{V}^\intercal$. The mean Frobenius percent error in $\boldsymbol{W}$ is 1%. With d relaxed to 50, recoverability diagrams for PCPF, RAPS, PCPSF and PCP, PCPS are drawn in Figures (c.I), (c.II) and (c.III), (c.IV). We observe substantial growth of recoverability for PCPS over PCP across the full range of ranks. And with features, there is still omniscient gain in recoverablity for PCPSF against PCPF, which is marked at low ranks.

We remark that in unrecoverable areas, PCPS and PCPSF still obtain much smaller values of $\|\boldsymbol{L} -$

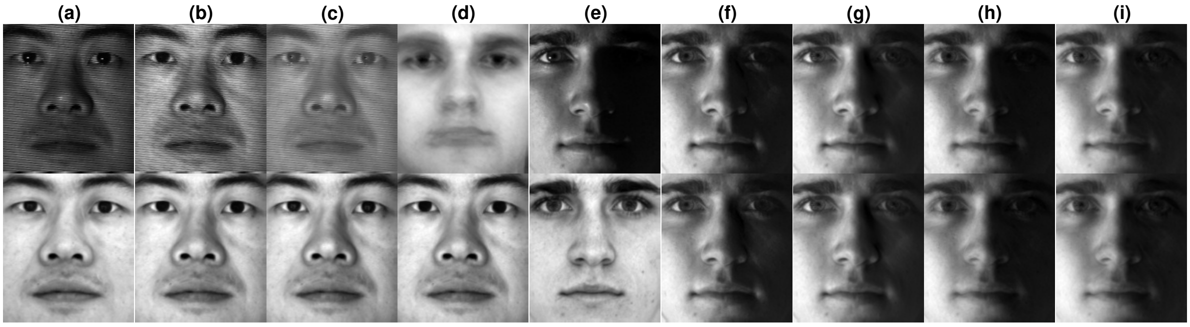


Figure 3.2: Comparison of face denoising ability: In row I, **(a, e)** sample frames from subjects 2 and 33; **(b, f)** single-person PCP; **(c, g)** single-person PCPF; **(h, i)** multi-person PCP and PCPF; **(d)** average of other subjects. In row II, **(a, e)** average of a single subject; **(b, f)** single-person PCPS; **(c, g)** single-person PCPSF; **(h, i)** multi-person PCPS and PCPSF; **(d)** PCPS using the side information above.

$\boldsymbol{L}_0\|_F$. In view of the marginal improvement of RAPS contrasted with PCPF and PCPSF, we will not consider it any longer. Results from RPCAG and PSSV are worse than PCP (see the appendix). FRPCAG fails to recover anything at all.

3.3.3 Face denoising under variable illumination

It has been previously proved that a convex Lambertian surface under distant and isotropic lighting has an underlying model that spans a 9-D linear subspace. Albeit faces can be described as Lambertian, it is only approximate and harmonic planes are not real images due to negative pixels. In addition, theoretical lighting conditions cannot be realised and there are unavoidable occlusion and albedo variations. It is thus more natural to decompose facial image formation as a low-rank component for face description and a sparse component for defects. What is more, we suggest that further boost to the performance of facial characterisation can be gained by leveraging an image which faithfully represents the subject.

We consider images of a fixed pose under different illuminations from the extended Yale B database for testing. Ten subjects were randomly chosen and all 64 images were studied for each person. For single-person experiments, 32556×64 observation matrices were formed by vectorising each 168×192 image and the side information was chosen to be the average of all images, tiled to the same size as the observation matrix for each subject. For the multiperson experiment, both single-person observation and side information matrices were concatenated into 32556×640 matrices respectively.

For PCPF and PCPSF to run, we learn the feature dictionary following an approach by Vishal et al [88]. In a nutshell, the feature learning process can be treated as a sparse encoding problem. More specifically, we simultaneously seek a dictionary $\boldsymbol{D} \in \mathbb{R}^{n_1 \times c}$ and a sparse representation $\boldsymbol{B} \in \mathbb{R}^{c \times n_2}$ such that:

$$\begin{aligned} &\underset{\boldsymbol{D},\boldsymbol{B}}{\text{minimize}} \quad \|\boldsymbol{M} - \boldsymbol{D}\boldsymbol{B}\|_F^2 \\ &\text{s.t. to} \quad \gamma_i \leq t \text{ for } i = 1 \dots n_2, \end{aligned} \tag{3.11}$$

where c is the number of atoms, γ_i's count the number of non-zero elements in each sparsity code and t is the sparsity constraint factor. This can be solved by the K-SVD algorithm. Here, feature $\boldsymbol{X}$ is the dictionary $\boldsymbol{D}$, feature $\boldsymbol{Y}$ corresponds to a similar solution using the transpose of the observation matrix as input and the sparse codes are irrelevant. For implementation details, we set c to 40, t to 40 and used 10 iterations. Because K-SVD could not converge in reasonable time for the multiperson experiment, we resorted to classical PCA applied to the observation matrix to obtain features $\boldsymbol{X}, \boldsymbol{Y}$ of dimension 400.

As a visual illustration, two challenging cases are exhibited in Figure 3.2 (PSSV, RPCAG, FRPCAG do not improve upon PCP and are shown in the appendix). For subject 2, it is clearly evident that PCPS and PCPSF outperform the best existing methods through the complete elimination of acquisition faults. More surprisingly, PCPSF even manages to restore the flash in the pupils that is not present in the side information. For subject 33, PCPS indubitably reconstructs a more vivid left eye than that from PCP which is only discernible. With that said, PCPSF still prevails by uncovering more shadows, especially around the medial canthus of the left eye, and revealing a more distinct crease in the upper eyelid as well a more translucent iris. We also notice that results from the single-person experiment outdo their counterparts from the multiperson experiment. Thence, we will focus on a single subject alone.

To quantitatively verify the improvement made by our proposed approaches, we examine the structural information contained within the deionised eigenfaces. Singular values of the recovered low-rank matrices from all algorithms are plotted in Figure 3.3. Singular values decease most sharply for PCPSF followed by PCPS. By the theoretical limit, they are orders of magnitude smaller than those values from other methods. This validates our proposed approaches.

We further unmask the strength of PCPS by considering the stringent side information made of the average of other subjects. Surprisingly, PCPS still manages to remove the noise recovering an authentic image (see Figure 3.2 **(d)**).

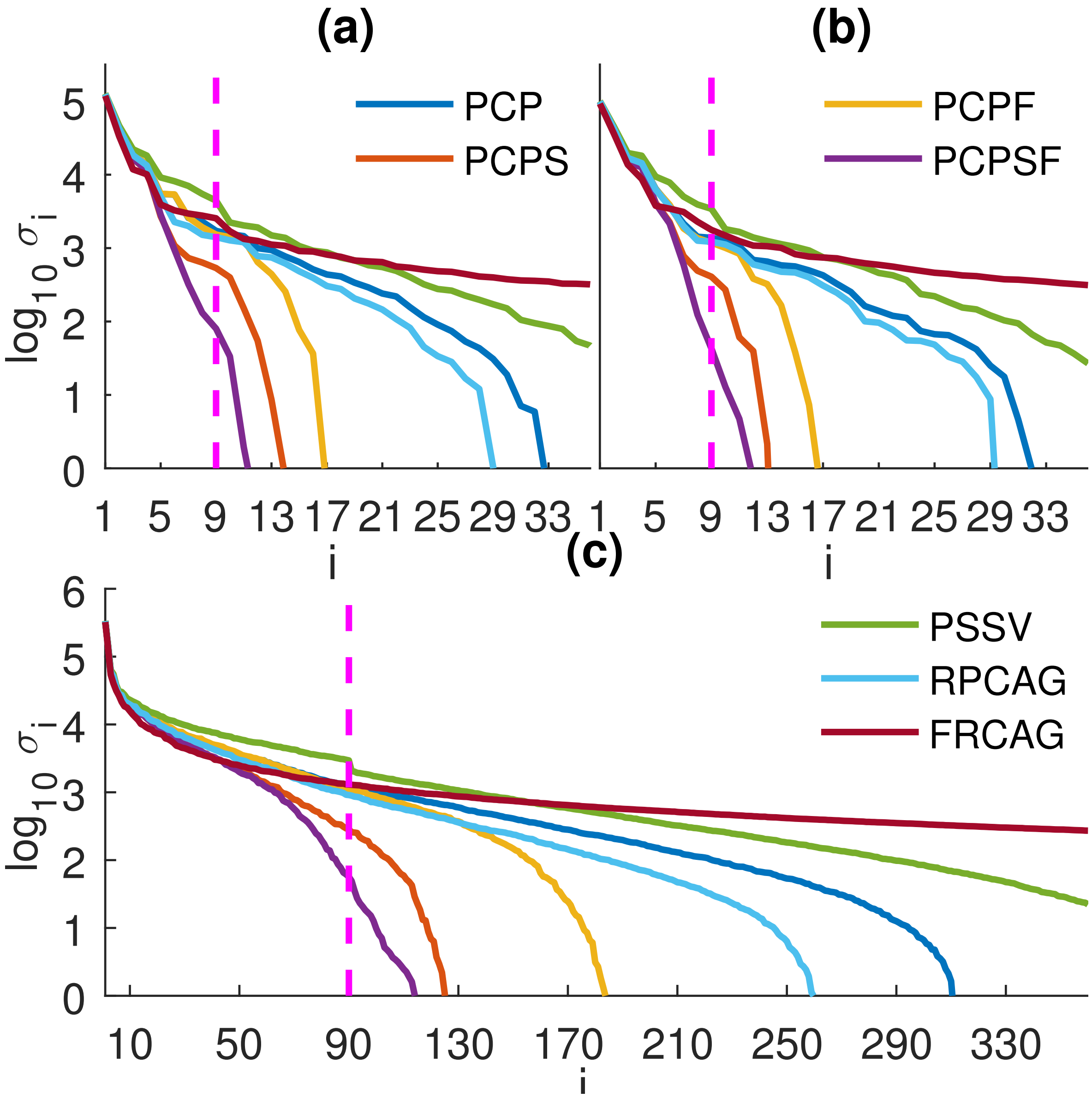


Figure 3.3: Log-scale singular values of the denoised matrices: **(a)** subject 2; **(b)** subject 33; **(c)** all subjects.

3.3.4 Background subtraction from surveillance video

We show that useful side information can help achieve better background restoration through the application of our proposed algorithm to a background-foreground separation scenario. One video sequence from the PETS 2006 dataset and one from the I2R dataset were utilised for evaluation. Each consists of scenes at a hall where people walk intermittently. 200 consecutive frames of 720×576

resolution grayscale images were stacked by columns into a 414720×200 observation matrix from the first video and 200 frames of 176×144 images from the second video were stacked into another 25344×200 observation matrix. Two side information arrays comprised columns that are copies of a vectorised photo which contains an empty hallway. To commence object detection, PCP and PCPS were first run to extract the backgrounds. Then objects were recovered by calculating the absolute values of the difference between the original frame and the estimated background. Since parameters for dictionary learning need exhaustive search, we will not be comparing PCPF and PCPSF for what follows.

We quantitatively compare the performance of the competing methods according to the weighted F-measure [77] against manually annotated bounding boxes provided as the ground truth. The resulting scores for each frame are presented in Figure 3.5. From the consistently higher precision statistics, the merit of PCPS over PCP is confirmed. For qualitative reference, representative images of the recovered background and foreground from all methods are listed in Figure 3.4 (For space reasons, we have only included the most noticeable sector. See the appendix for whole images.). PCP and its variants only partially detect infrequent moving objects, people who stop moving for extended periods of time, leaving ghost artifacts in the background. In contrast, PCPS segments a fairly sharp silhouette of slowly moving objects to produce a much cleaner background, promoting its novelty.

To further unravel of the robustness of our propositions, shortened videos from PETS and Airport consisting of 60 frames are analysed via PCPS. Figures 3.4 **(c,d)** & **(o,p)** show that PCPS with less input can achieve comparative or better results than PCP with more input. This suggests that the transductive constraint of RPCA no longer applies because with the help of side information we can run PCPS on fewer frames rather than the entire collection every time new observation arrives. The speed-ups for PETS and Airport are $2.44\times$ and $2.62\times$ respectively.

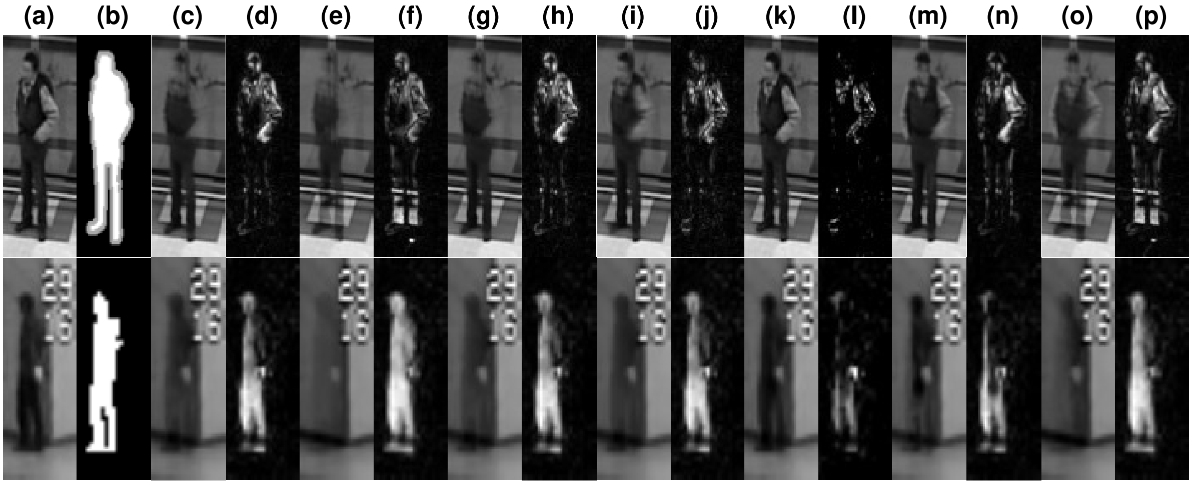


Figure 3.4: Background subtraction results for two sample frames, PETS in row I and Airport in row II: **(a)** original images; **(b)** ground truth; **(c,d)** PCP; **(e,f)** PCPS; **(g,h)** PSSV; **(i,j)** RPCAG; **(k,l)** FRPCAG; **(m,n)** PCP (60 frames); **(o,p)** PCPS (60 frames).

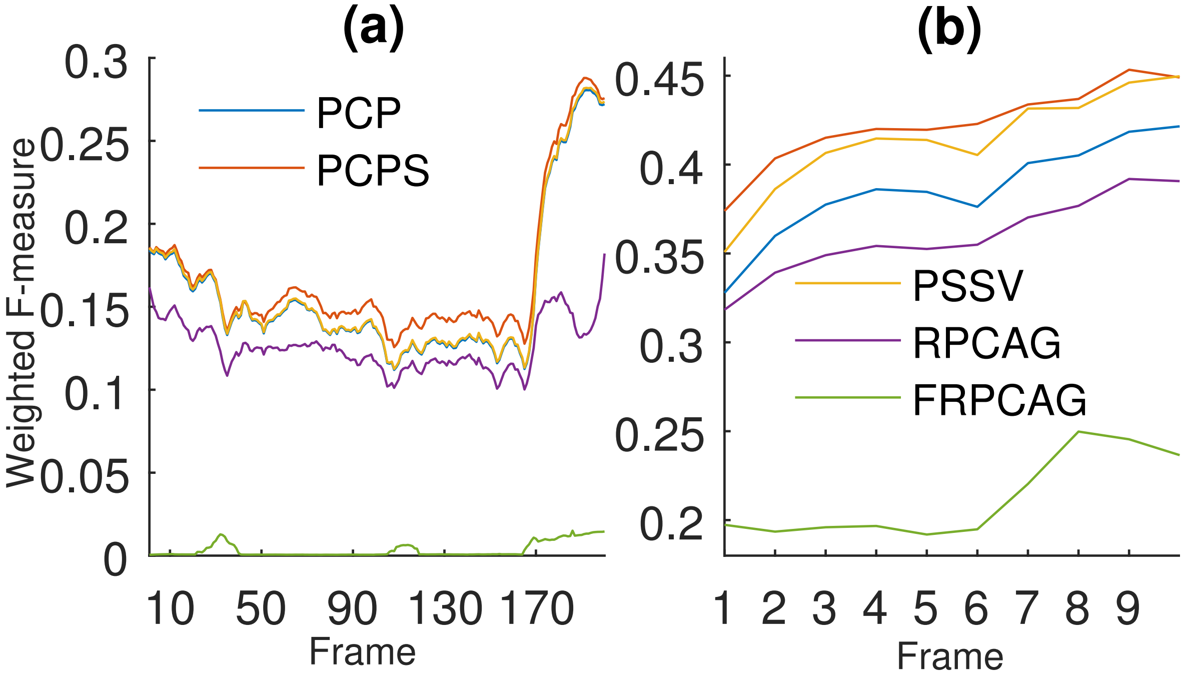


Figure 3.5: Weighted F-measure scores: **(a)** PETS; **(b)** Airport.

3.3.5 Face and facial expression recognition

We demonstrate how the accurate reconstruction of facial expressions guided by side information ameliorates classification analysis. To begin with, evaluation was effected on the CMU Multi-PIE dataset. Aligned and cropped 165×172 images of frontal pose and normal lighting from 54 subjects were used. We batch-processed each subject forming a 28380×6 observation matrix to extract expressions: Neutral, Smile, Surprise, Disgust, Scream and Squint. For each subject, side information was offered by a sextet of neutral face repetitions. Archetypal expressions recovered by PCP, PCPS, PSSV, RPCAG are laid out in Figure 3.6 (the restricted number of expressions disallows FRPCAG). It is noteworthy that local appearance changes separated by PCPS are the most salient which paves the way for better classification. We avail ourselves of the multi-class RBF-kernel SVM and the SRC [125] to map expressions to emotions. 9-fold cross-validation results are reported in Table 3.1. PCPS leads PCP by a fair margin with PSSV, RPCAG underperforming PCP.

Table 3.1: Classification accuracy (%) on the Multi-PIE dataset for PCP, PSSV, PCPS and RPCAG by means of non-linear SVM and SRC learning.

Algorithm	PCP	PSSV	PCPS	RPCAG
Non-linear SVM	78.40	74.69	**79.94**	77.16
SRC	79.01	74.38	**82.72**	79.01

Lastly, the CK+ dataset was incorporated to assess the joint face and expression recognition capabilities of various algorithms. Each test image is sparsely coded via a dictionary of both identities and universal expressions (Anger, Disgust, Fear, Happiness, Sadness and Surprise). The least resulting reconstruction residual thereupon determines its identity or expression. We refer readers to [45] for the exact problem set-up and implementation details. Table 3.2 collects the computed recognition rates. Although RPCAG and FRPCAG are superior than PCP as expected, PCPS performs distinctly better than all others.

Table 3.2: Recognition rates (%) for joint identity & expression recognition averaged over 10 trials on CK+

Algorithm	PCP	PSSV	PCPS	RPCAG	FRPCAG
Identity	87.35	87.05	**95.23**	89.77	90.98
Expression	49.24	45.30	**67.50**	58.26	57.73

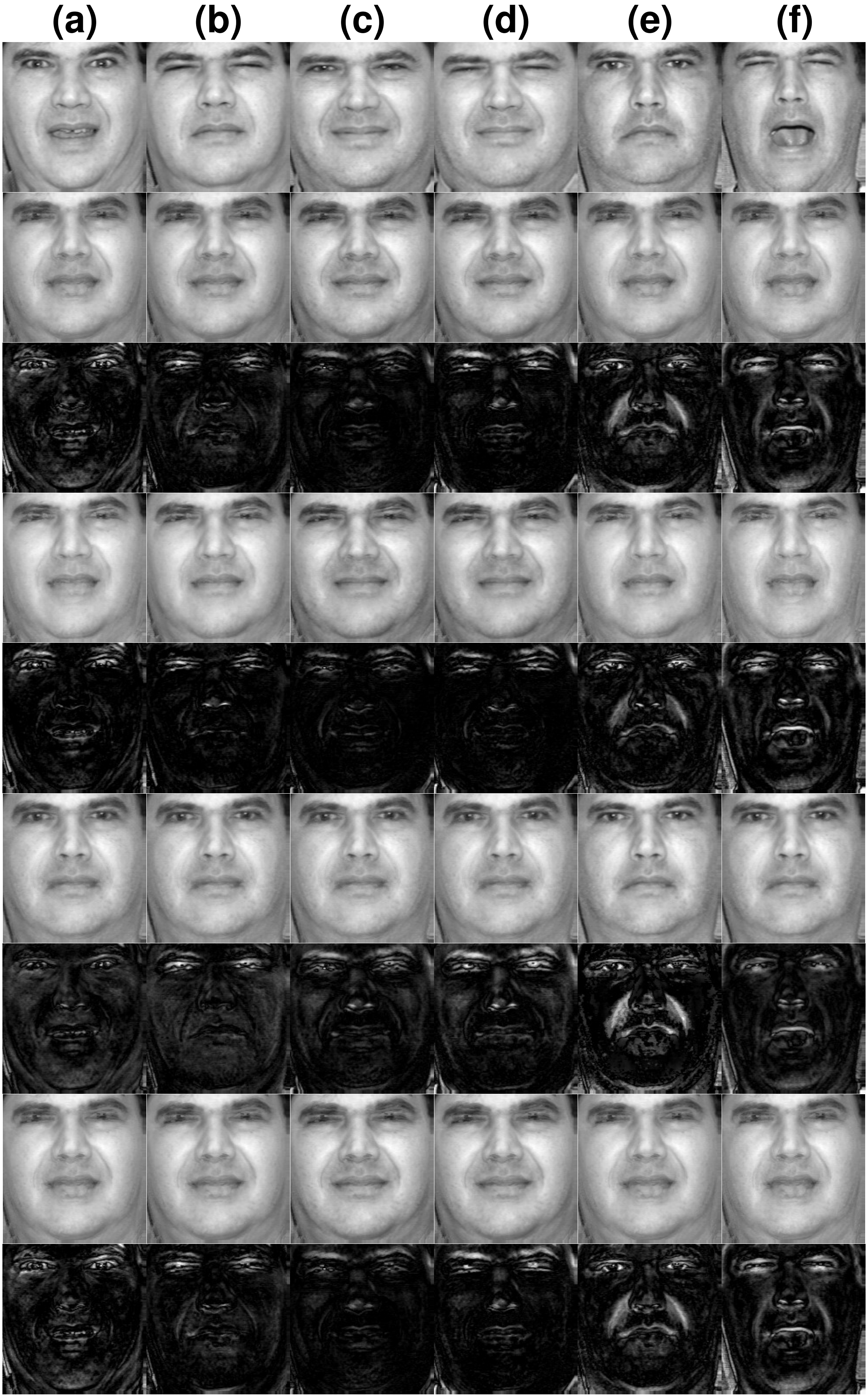


Figure 3.6: Expression extraction for a single subject: Expressive faces reside in row I. Identity classes produced by PCP, PSSV, PCPS, RPCAG are in rows II, IV, VI, VIII. The complementary expression components are depicted in rows III, V, VII, IX.

3.4 Appendix

3.4.1 Parameter calibration

In order to tune the algorithmic parameters, we first conduct a benchmark experiment as follows: a low-rank matrix $\boldsymbol{L}_0$ is generated from $\boldsymbol{L}_0 = \boldsymbol{J}\boldsymbol{K}^\intercal$, where $\boldsymbol{J}, \boldsymbol{K} \in \mathbb{R}^{200\times 10}$ have entries from a $\mathcal{N}(0, 0.005)$ distribution; a 200×200 sparse matrix $\boldsymbol{S}_0$ is generated by randomly setting $38,000$ entries to zero with others taking values of ± 1 with equal probability.

If $\boldsymbol{X}$ is set as the left-singular vectors of $\boldsymbol{L}_0$ and $\boldsymbol{Y}$ is set as the right-singular vectors of $\boldsymbol{L}_0$, then a scaling ratio $\alpha = 1.1$, a tolerance threshold $\epsilon = 10^{-7}$ and a maximum step size $\mu = 10^{18}$ to avoid ill-conditioning can bring PCP, RAPS, PCPF to convergence with a recovered $\boldsymbol{L}$ of rank 10, a recovered $\boldsymbol{S}$ of sparsity 5% and an accuracy $\|\boldsymbol{L} - \boldsymbol{L}_0\|_F / \|\boldsymbol{L}_0\|_F$ on the order of 10^{-6}. Hereafter, we will adopt these parameter settings for PCP, RAPS, PCPF and will apply them to PCPS and PCPSF as well. PSSV also uses these parameter settings as done similarly in [19].

For RPCAG and FRPCAG, the graphs are built using k-nearest neighbors. Using Euclidean distances, each sample is connected to 10 nearest neighbors with weight $e^{-\frac{s^2}{\sigma^2}}$, where s is the Euclidean distance between the two samples and σ is the average of s. Weight between unconnected samples is set to 0. Having obtained such weight matrix $\boldsymbol{A}$, we can calculate the normalised graph Laplacian $\boldsymbol{\Phi} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the diagonal degree matrix. The tolerance threshold for RPCAG and FRPCAG are all set to $\epsilon = 10^{-7}$ for reasons of consistency. We choose $\lambda = 1/\sqrt{\max(n_1, n_2)}$ for a general matrix of dimension $n_1 \times n_2$ as suggested in [23,24]. For simulation experiments, γ in RPCAG is given by the minimiser (at $\gamma = 0.2$) of $\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$ on the benchmark problem (Figure 3.7). And for real-world datasets, γ is set to 10 following [23]. For FRPCAG, we take $\gamma = \gamma_1 = \gamma_2$ which is searched over $[0.01, 10]$ on the benchmark problem (Figure 3.8). The resulting minimiser (at $\gamma = 7.3$) of $\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$ is used in both simulation and real-world experiments.

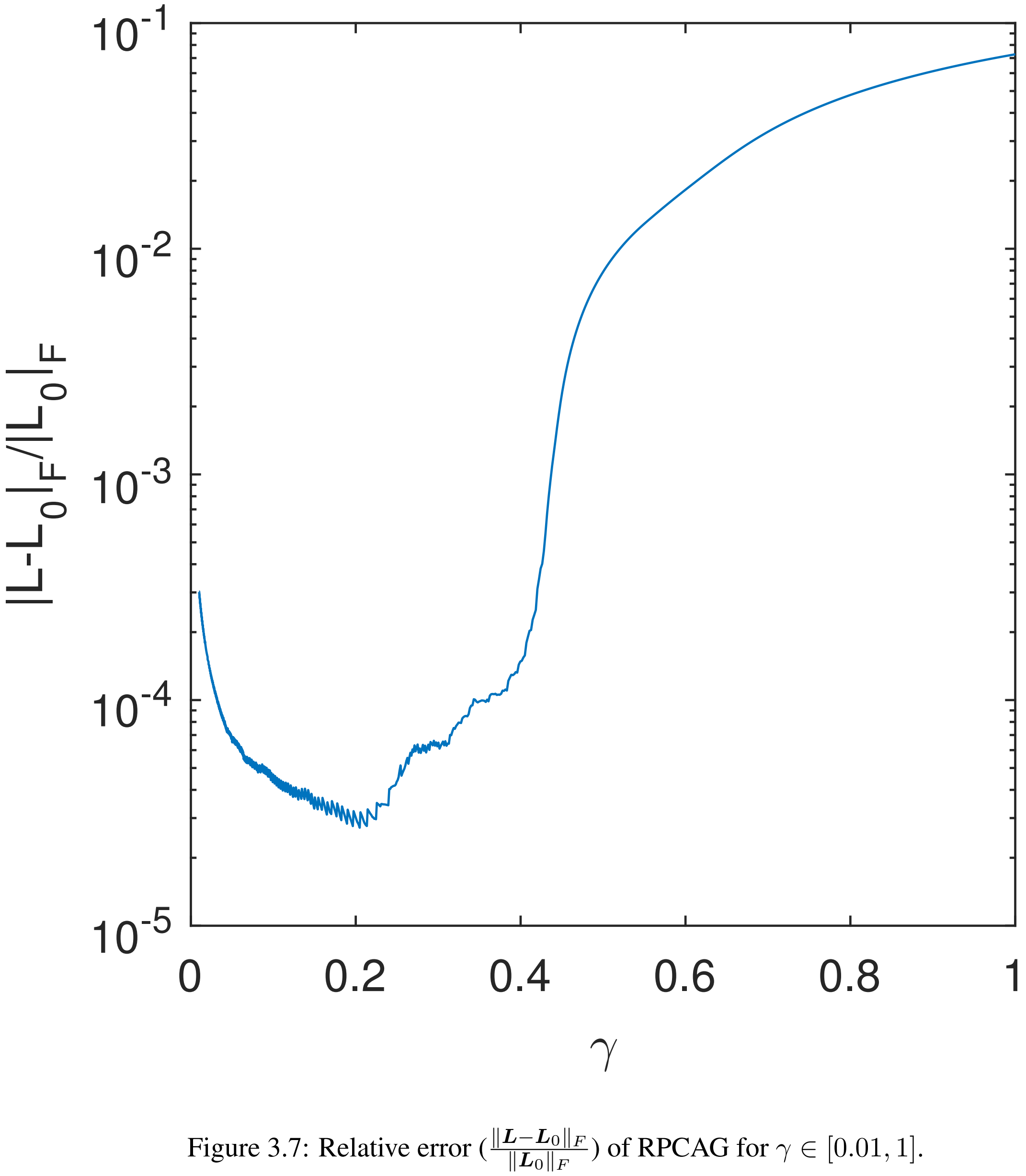


Figure 3.7: Relative error ($\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$) of RPCAG for $\gamma \in [0.01, 1]$.

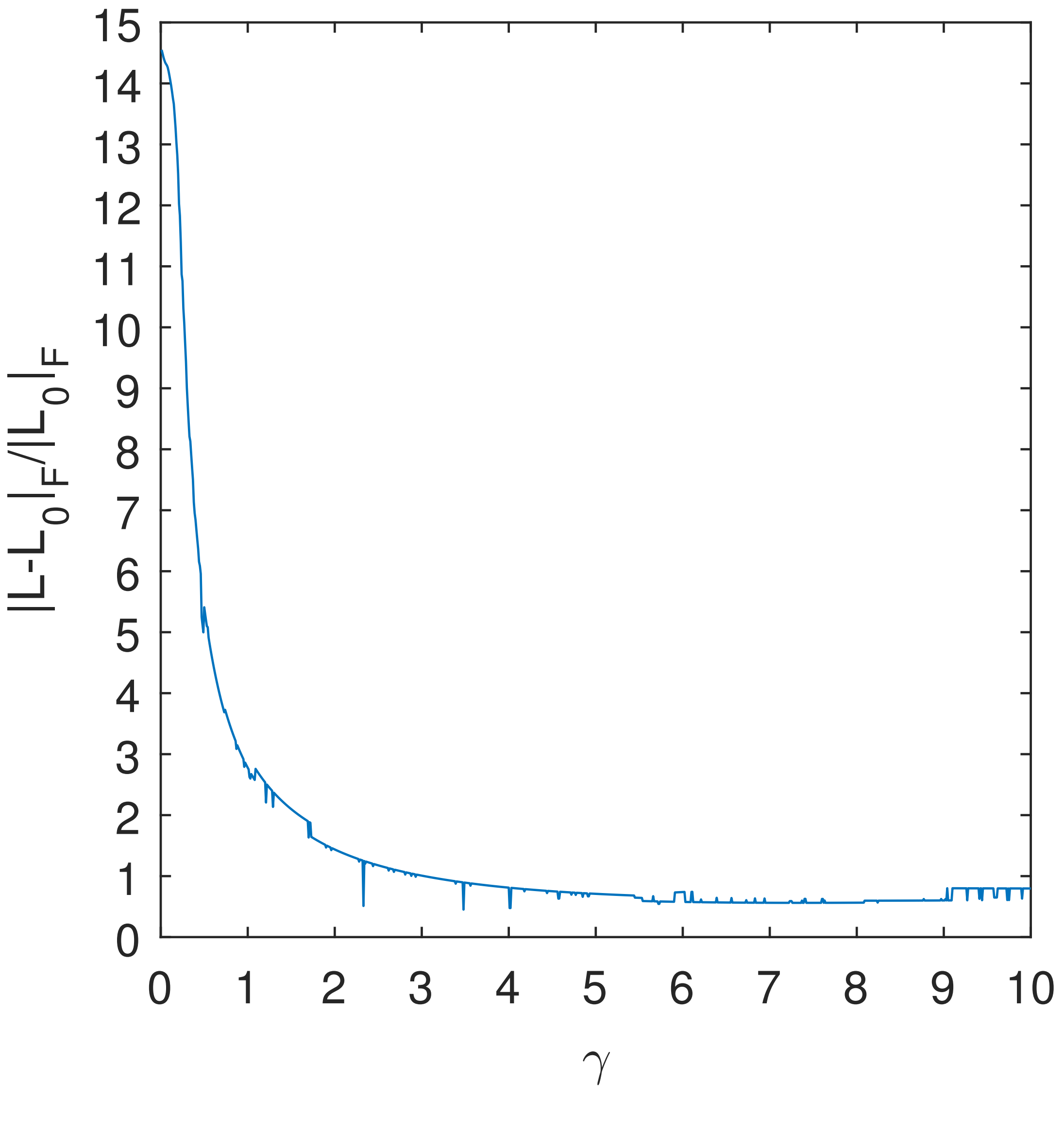


Figure 3.8: Relative error ($\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$) of FRPCAG for $\gamma \in [0.01, 10]$.

To find λ and κ in PCPS, a parameter sweep in the $\kappa - \lambda$ space for perfect side information ($\boldsymbol{W} = \boldsymbol{L}_0$) is shown in Figure 3.9 (a) and for observation as side information ($\boldsymbol{W} = \boldsymbol{M}$) in Figure 3.9 (b) to impart a lower bound and a upper bound respectively. It can be easily seen that $\lambda = 1/\sqrt{200}$ from PCP works well in both cases. Conversely, κ depends on the quality of the side information. At $\lambda = 1/\sqrt{200}$, the minimiser of $\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$ occurs at $\kappa = 0.2$ for noisy side information. This value of κ together with $\lambda = 1/\sqrt{200}$ is used in simulation experiments for both PCPS and PCPSF. For public video sequences, increasing the value of κ to 0.5 can produce visual results that are noticeable to the naked eye.

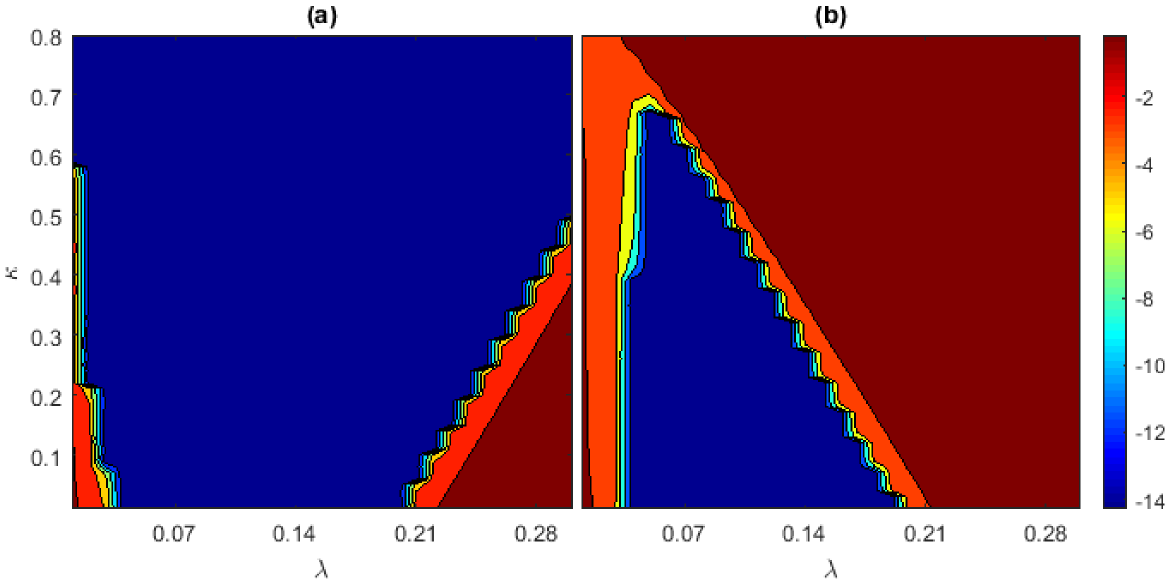


Figure 3.9: Relative error ($\frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$) of PCPS: **(a)** when side information is perfect; **(b)** when side information is the observation.

3.4.2 Simulation Results

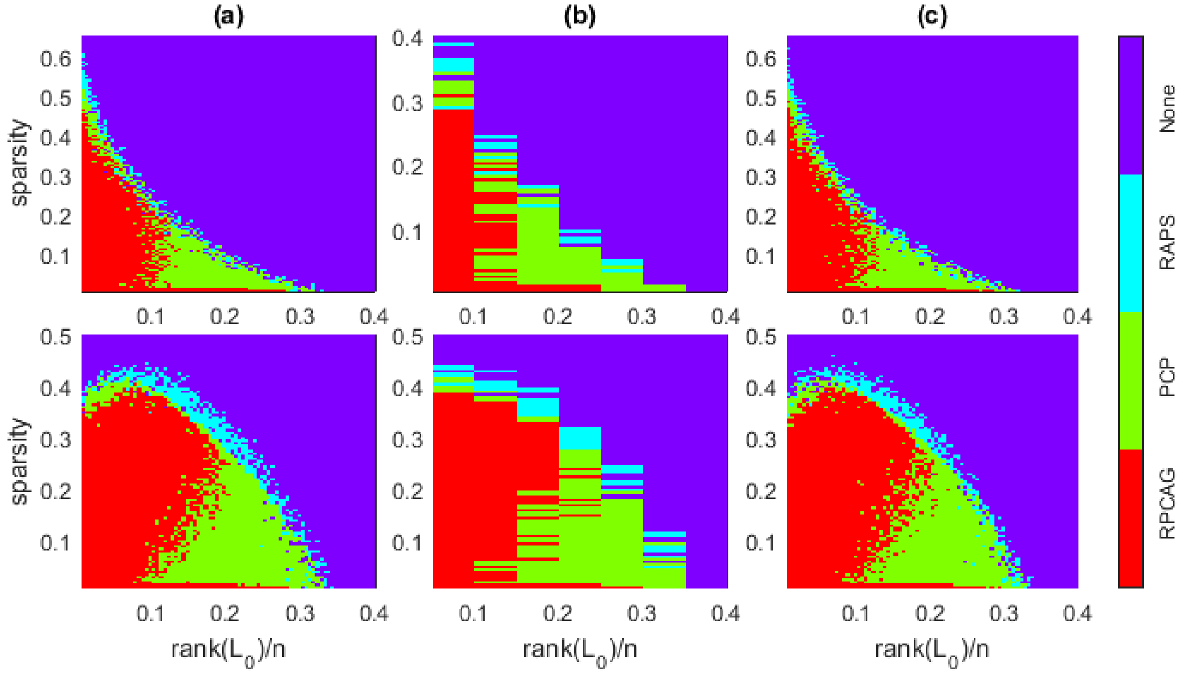


Figure 3.10: Domains of recovery by various algorithms: random signs in row **I** and coherent signs in row **II**. **(a)** for entry-wise corruptions, **(b)** for deficient ranks and **(c)** for distorted singular values.

A direct comparison of RAPS, RPCAG and PCP from simulation studies is presented in Figure 3.10. Simulation results for PSSV are shown in Figure 3.11.

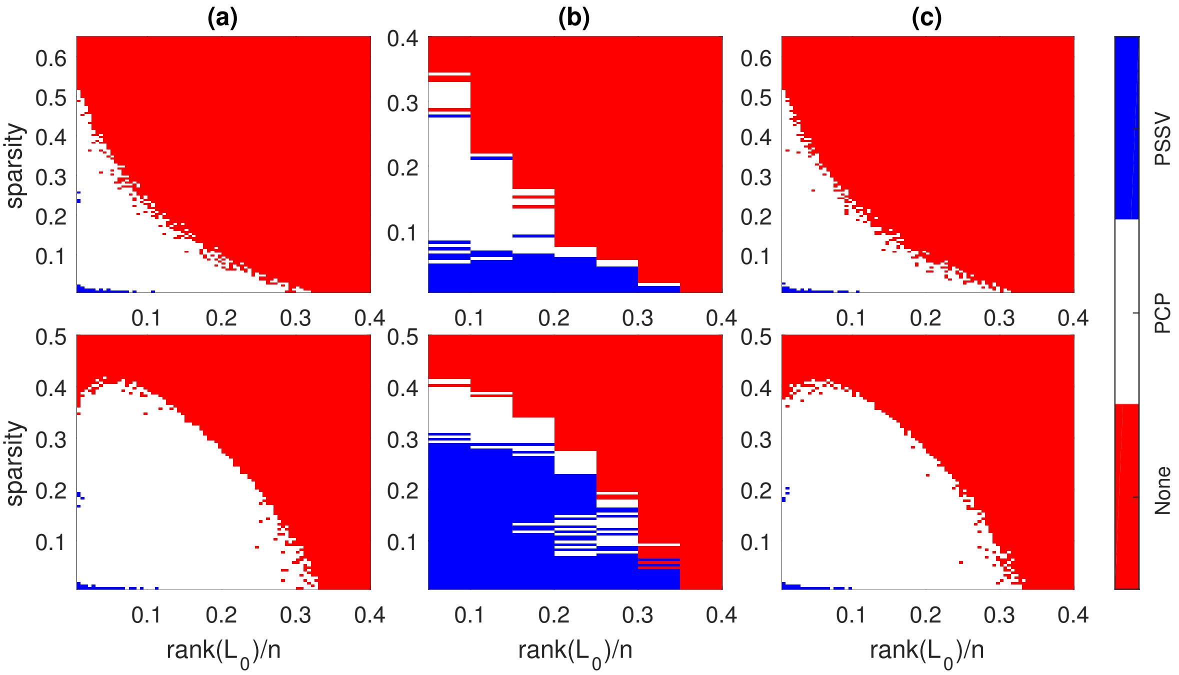


Figure 3.11: Domains of recovery by PSSV: random signs in row **I** and coherent signs in row **II**. **(a)** for entry-wise corruptions, **(b)** for deficient ranks and **(c)** for distorted singular values.

3.4.3 Real-world applications

3.4.4 Data sources

The datasets used herein are listed below:

The Extended Yale Face Database B: http://vision.ucsd.edu/~iskwak/ExtYaleDatabase/ExtYaleB.html.

Performance Evaluation of Tracking and Surveillance Workshop 2006: http://www.cvg.reading.ac.uk/PETS2006/data.html.

I2R Dataset: http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html.

The CMU Multi-PIE Face Database: http://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie/Multi-Pie/Home.html.

The Extended Cohn-Kande Dataset (CK+): http://www.consortium.ri.cmu.edu/ckagree/.

3.4.5 Face denoising

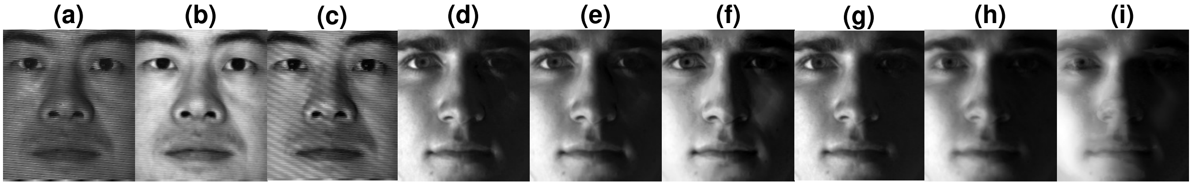


Figure 3.12: Comparison of face denoising ability: **(a,d)** single-person PSSV; **(b,e)** single-person RPCAG; **(c,f)** single-person FRPCAG; **(g)** multi-person PSSV; **(h)** multi-person RPCAG; and **(i)** multi-person FRPCAG;.

Illustration of face denoising ability of PSSV, RPCAG, FRPCAG is presented in Figure 3.12. The average running times of different algorithms for a single subject and multiple subjects are summarised in Table 3.3 [1].

[1] All experiments were performed on a 3.60GHz quad-core computer with 16GB RAM running MATLAB R2016a.

Table 3.3: Running times of various algorithms.

Algorithm	Time	
	Single Subject	Multiple Subjects
K-SVD (X)	9 min	—
K-SVD (Y)	78 min	—
PCP	12s	5 min
PCPS	27s	12 min
PCPF	16s	9 min
PCPSF	19s	8 min
PSSV	13s	5 min
k-NN (X)	7s	4 min
k-NN (Y)	1s	8s
RPCAG	2min	17 min
FRPCAG	8s	1 min

3.4.6 Background Subtraction

Recovered images of the background and the foreground from all methods are listed in Figure 3.13 for Airport and Figure 3.14 for PETS. The running times of different algorithms for Airport and PETS are summarised in Table 3.4.

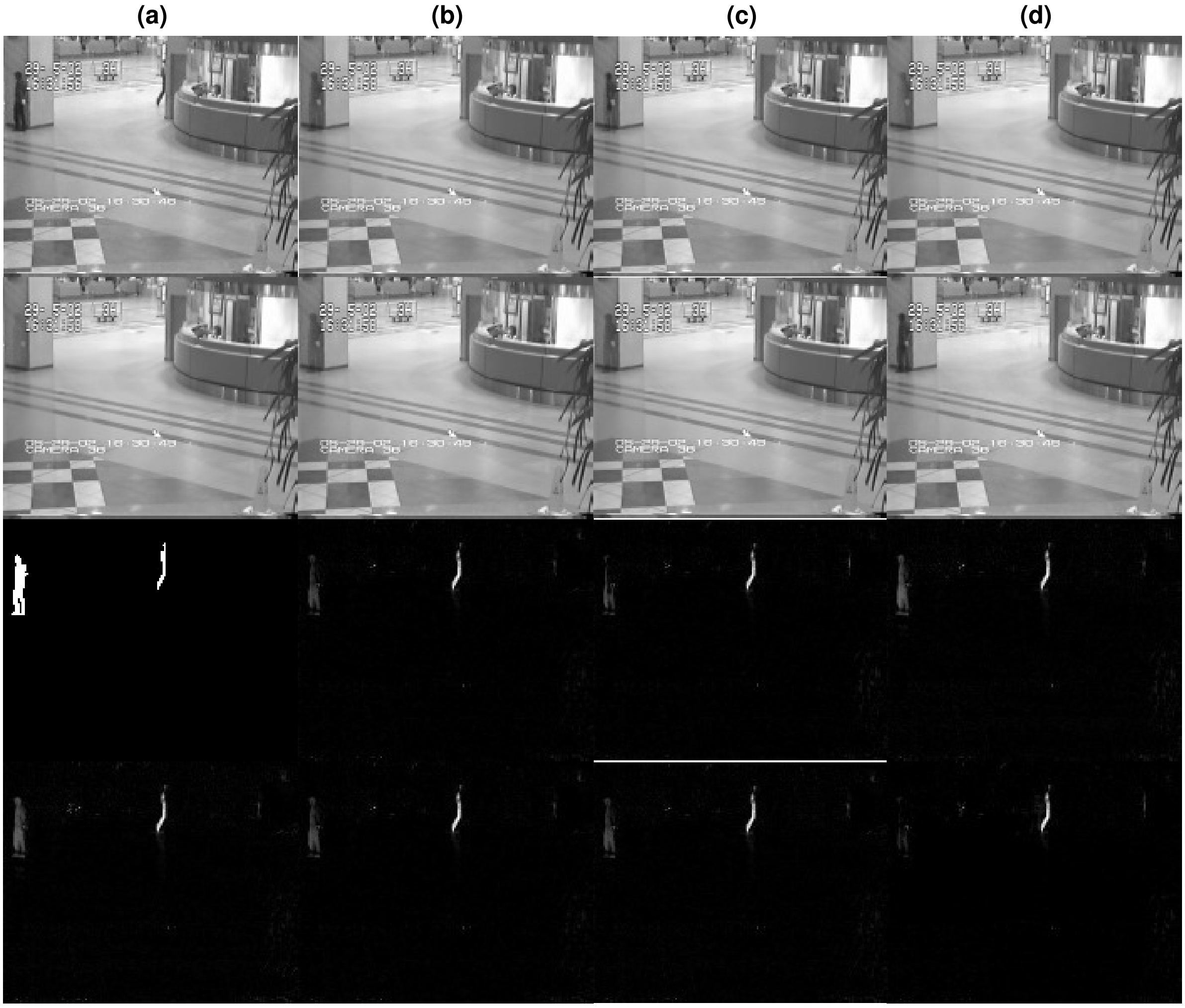


Figure 3.13: Background subtraction results for Airport : row **I** **(a)** original image; row **III** **(a)** ground truth; row **I,III** **(b)** PCP; row **I,III** **(c)** PCP (60 frames); **I,III** **(d)** PCPS (60 frames); row **II,IV** **(a)** PCPS; row **II,IV** **(b)** PSSV; row **II,IV** **(c)** RPCAG; row **II,IV** **(d)** FRPCAG.

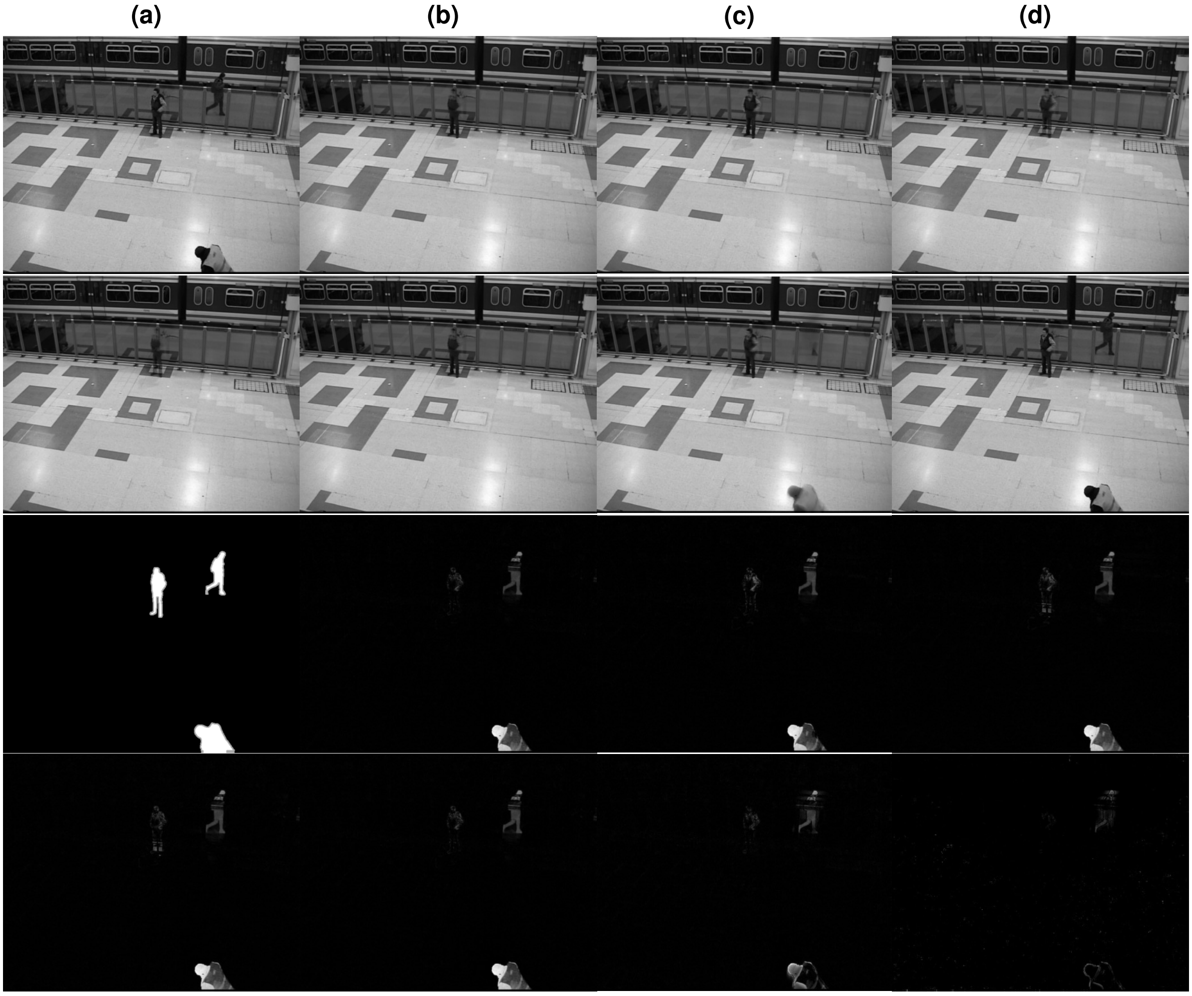


Figure 3.14: Background subtraction results for PETS : row **I (a)** original image; row **III (a)** ground truth; row **I,III (b)** PCP; row **I,III (c)** PCP (60 frames); **I,III (d)** PCPS (60 frames); row **II,IV (a)** PCPS; row **II,IV (b)** PSSV; row **II,IV (c)** RPCAG; row **II,IV (d)** FRPCAG.

Table 3.4: Running times of various algorithms.

Algorithm	Time	
	Airport	PETS
PCP	52s	17 min
PCPS	2 min	36 min
PSSV	51s	17 min
k-NN (X)	52s	2h
k-NN (Y)	1s	24s
RPCAG	7 min	3h
FRPCAG	11s	34s
PCP (60 frames)	52s	3 min
PCPS (60 frames)	20s	7 min

3.4.7 Derivations

Here we give deviations of the various equivalent subproblems for the algorithm quoted in the text:

$$
\begin{aligned}
&\operatorname*{argmin}_{\boldsymbol{H}} \mathcal{L}(\boldsymbol{H},\boldsymbol{E},\boldsymbol{S},\boldsymbol{Z},\boldsymbol{N})\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \kappa||\boldsymbol{E}||_* + \lambda||\boldsymbol{S}||_1 + \langle \boldsymbol{Z}, \boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}\rangle + \frac{\mu}{2}||\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}||_F^2\\
&+\langle \boldsymbol{N}, \boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}\rangle + \frac{\mu}{2}||\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}||_F^2\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \langle \boldsymbol{Z}, \boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}\rangle + \frac{\mu}{2}||\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}||_F^2\\
&+\langle \boldsymbol{N}, \boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}\rangle + \frac{\mu}{2}||\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}||_F^2\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \operatorname{tr}(\boldsymbol{Z}^{\intercal}(\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}))\\
&+\frac{\mu}{2}\operatorname{tr}((\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})^{\intercal}(\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})) + \operatorname{tr}(\boldsymbol{N}^{\intercal}(\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}))\\
&+\frac{\mu}{2}\operatorname{tr}((\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})^{\intercal}(\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y}))\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* - \operatorname{tr}(\boldsymbol{Z}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}) + \operatorname{tr}(\boldsymbol{N}^{\intercal}\boldsymbol{H})\\
&+\frac{\mu}{2}\operatorname{tr}(\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal} - \boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}(\boldsymbol{M}-\boldsymbol{S}) - (\boldsymbol{M}-\boldsymbol{S})^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})\\
&+\frac{\mu}{2}\operatorname{tr}((\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}(\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})\boldsymbol{Y}^{\intercal}\boldsymbol{Y})\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \mu\operatorname{tr}(-\frac{1}{\mu}\boldsymbol{Z}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}) + \mu\operatorname{tr}(\frac{1}{\mu}\boldsymbol{N}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}\boldsymbol{Y})\\
&+\frac{\mu}{2}\operatorname{tr}(\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal} - \boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}(\boldsymbol{M}-\boldsymbol{S}) - (\boldsymbol{M}-\boldsymbol{S})^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})\\
&+\frac{\mu}{2}\operatorname{tr}(\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal} - \boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}(\boldsymbol{E}+\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})\boldsymbol{Y}^{\intercal}\\
&-\boldsymbol{Y}(\boldsymbol{E}+\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \mu\operatorname{tr}(\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal} - \frac{1}{2}\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}(\boldsymbol{M}-\boldsymbol{S}) - \frac{1}{2}(\boldsymbol{M}-\boldsymbol{S})^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}\\
&-\frac{1}{2}\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}(\boldsymbol{E}+\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})\boldsymbol{Y}^{\intercal} - \frac{1}{2}\boldsymbol{Y}(\boldsymbol{E}+\boldsymbol{X}^{\intercal}\boldsymbol{W}\boldsymbol{Y})^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}\\
&-\frac{1}{2\mu}\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{Z} - \frac{1}{2\mu}\boldsymbol{Z}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal} + \frac{1}{2\mu}\boldsymbol{Y}\boldsymbol{H}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{N}\boldsymbol{Y}^{\intercal} + \frac{1}{2\mu}\boldsymbol{Y}\boldsymbol{N}^{\intercal}\boldsymbol{X}^{\intercal}\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \mu\operatorname{tr}((\frac{1}{2}(\boldsymbol{M}-\boldsymbol{S}+\boldsymbol{X}\boldsymbol{E}\boldsymbol{Y}^{\intercal}+\boldsymbol{W}+\frac{1}{\mu}(\boldsymbol{Z}-\boldsymbol{X}\boldsymbol{N}\boldsymbol{Y}^{\intercal})) - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal})^{\intercal}\\
&(\frac{1}{2}(\boldsymbol{M}-\boldsymbol{S}+\boldsymbol{X}\boldsymbol{E}\boldsymbol{Y}^{\intercal}+\boldsymbol{W}+\frac{1}{\mu}(\boldsymbol{Z}-\boldsymbol{X}\boldsymbol{N}\boldsymbol{Y}^{\intercal})) - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}))\\
&=\operatorname*{argmin}_{\boldsymbol{H}}\ ||\boldsymbol{H}||_* + \mu||\frac{1}{2}(\boldsymbol{M}-\boldsymbol{S}+\boldsymbol{W}+\frac{1}{\mu}\boldsymbol{Z}+\boldsymbol{X}(\boldsymbol{E}-\frac{1}{\mu}\boldsymbol{N})\boldsymbol{Y}^{\intercal}) - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^{\intercal}||_F^2
\end{aligned}
\tag{3.12}
$$

$$
\begin{aligned}
&\operatorname*{argmin}_{\boldsymbol{E}} \mathcal{L}(\boldsymbol{H},\boldsymbol{E},\boldsymbol{S},\boldsymbol{Z},\boldsymbol{N})\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ ||\boldsymbol{H}||_* + \kappa||\boldsymbol{E}||_* + \lambda||\boldsymbol{S}||_1 + \langle \boldsymbol{Z}, \boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal\rangle + \frac{\mu}{2}||\boldsymbol{M}-\boldsymbol{S}-\boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal||_F^2\\
&+\langle \boldsymbol{N}, \boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}\rangle + \frac{\mu}{2}||\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}||_F^2\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_* + \langle \boldsymbol{N}, \boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}\rangle + \frac{\mu}{2}||\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}||_F^2\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_* + \operatorname{tr}(\boldsymbol{N}^\intercal(\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}))\\
&+\frac{\mu}{2}\operatorname{tr}((\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y})^\intercal(\boldsymbol{H}-\boldsymbol{E}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}))\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_* + \frac{\mu}{2}\operatorname{tr}(-\frac{2}{\mu}\boldsymbol{N}^\intercal\boldsymbol{E})\\
&+\frac{\mu}{2}\operatorname{tr}(\boldsymbol{E}^\intercal\boldsymbol{E}-\boldsymbol{E}^\intercal(\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y})-(\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y})^\intercal\boldsymbol{E})\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_*\\
&+\frac{\mu}{2}\operatorname{tr}(\boldsymbol{E}^\intercal\boldsymbol{E}-\boldsymbol{E}^\intercal(\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y})-(\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y})^\intercal\boldsymbol{E}-\frac{1}{\mu}\boldsymbol{E}^\intercal\boldsymbol{N}-\frac{1}{\mu}\boldsymbol{N}^\intercal\boldsymbol{E})\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_* + \frac{\mu}{2}\operatorname{tr}((\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}+\frac{1}{\mu}\boldsymbol{N}-\boldsymbol{E})^\intercal(\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}+\frac{1}{\mu}\boldsymbol{N}-\boldsymbol{E}))\\
=&\operatorname*{argmin}_{\boldsymbol{E}}\ \kappa||\boldsymbol{E}||_* + \frac{\mu}{2}||\boldsymbol{H}-\boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}+\frac{1}{\mu}\boldsymbol{N}-\boldsymbol{E}||_F^2
\end{aligned}
\tag{3.13}
$$

$$
\begin{aligned}
&\operatorname*{argmin}_{\boldsymbol{S}} \mathcal{L}(\boldsymbol{H}, \boldsymbol{E}, \boldsymbol{S}, \boldsymbol{Z}, \boldsymbol{N}) \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ ||\boldsymbol{H}||_* + \kappa||\boldsymbol{E}||_* + \lambda||\boldsymbol{S}||_1 + \langle \boldsymbol{Z}, \boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal \rangle + \frac{\mu}{2}||\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal||_F^2 \\
&+ \langle \boldsymbol{N}, \boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal \boldsymbol{W}\boldsymbol{Y} \rangle + \frac{\mu}{2}||\boldsymbol{H} - \boldsymbol{E} - \boldsymbol{X}^\intercal\boldsymbol{W}\boldsymbol{Y}||_F^2 \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 + \langle \boldsymbol{Z}, \boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal \rangle + \frac{\mu}{2}||\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal||_F^2 \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 + \operatorname{tr}(\boldsymbol{Z}^\intercal(\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)) \\
&+ \frac{\mu}{2}\operatorname{tr}((\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)^\intercal(\boldsymbol{M} - \boldsymbol{S} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)) \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 + \frac{\mu}{2}\operatorname{tr}(-\frac{2}{\mu}\boldsymbol{Z}^\intercal\boldsymbol{S}) \\
&+ \frac{\mu}{2}\operatorname{tr}(\boldsymbol{S}^\intercal\boldsymbol{S} - \boldsymbol{S}^\intercal(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal) - (\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)^\intercal\boldsymbol{S}) \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 \\
&+ \frac{\mu}{2}\operatorname{tr}(\boldsymbol{S}^\intercal\boldsymbol{S} - \boldsymbol{S}^\intercal(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal) - (\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal)^\intercal\boldsymbol{S} - \frac{1}{\mu}\boldsymbol{S}^\intercal\boldsymbol{Z} - \frac{1}{\mu}\boldsymbol{Z}^\intercal\boldsymbol{S}) \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 + \frac{\mu}{2}\operatorname{tr}((\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \frac{1}{\mu}\boldsymbol{Z} - \boldsymbol{S})^\intercal(\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \frac{1}{\mu}\boldsymbol{Z} - \boldsymbol{S})) \\
=&\operatorname*{argmin}_{\boldsymbol{S}} \ \lambda||\boldsymbol{S}||_1 + \frac{\mu}{2}||\boldsymbol{M} - \boldsymbol{X}\boldsymbol{H}\boldsymbol{Y}^\intercal + \frac{1}{\mu}\boldsymbol{Z} - \boldsymbol{S}||_F^2
\end{aligned}
\tag{3.14}
$$

3.4.8 Further comments

One might suggest that a potentially better and more direct approach in using the side information is to subtract the side information. That is, do RPCA on $\boldsymbol{M}' = \boldsymbol{M} - \boldsymbol{W}$, where $\boldsymbol{M}$ is the data and $\boldsymbol{W}$ is the noisy side information, to obtain $\boldsymbol{M}' = \boldsymbol{L}' + \boldsymbol{S}$ with $\boldsymbol{L} = \boldsymbol{L}' + \boldsymbol{W}$.

We argue that this is not correct for the following reasons:

- The rank of $\boldsymbol{L}'$ is no smaller than $\boldsymbol{L}$, which does not make the problem any simpler than the original one.
- When $\boldsymbol{W}$ is merged into $\boldsymbol{M}$, the additional information provided by $\boldsymbol{W}$ is lost and the features can on longer be applied.
- When $\boldsymbol{W}$ includes full-rank noise on $\boldsymbol{L}$, $\boldsymbol{L}'$ is not low-rank anymore. This violates the assumption of RPCA.

To verify our claim, we perform the Airport experiment again, but with different side information than that used in the paper. We collect 200 different frames of relatively clean backgrounds and stack them into the side information $\boldsymbol{W}$. Comparison of the suggestion with PCPS and PCP is shown in Figure 3.15, 3.16 and 3.17. It is clearly visible that the low-rank structure cannot be recovered by the suggestion and spurious noises are introduced in the segmentation, whereas PCPS works impeccably segmenting accurately the foreground moving objects leaving a clean background.

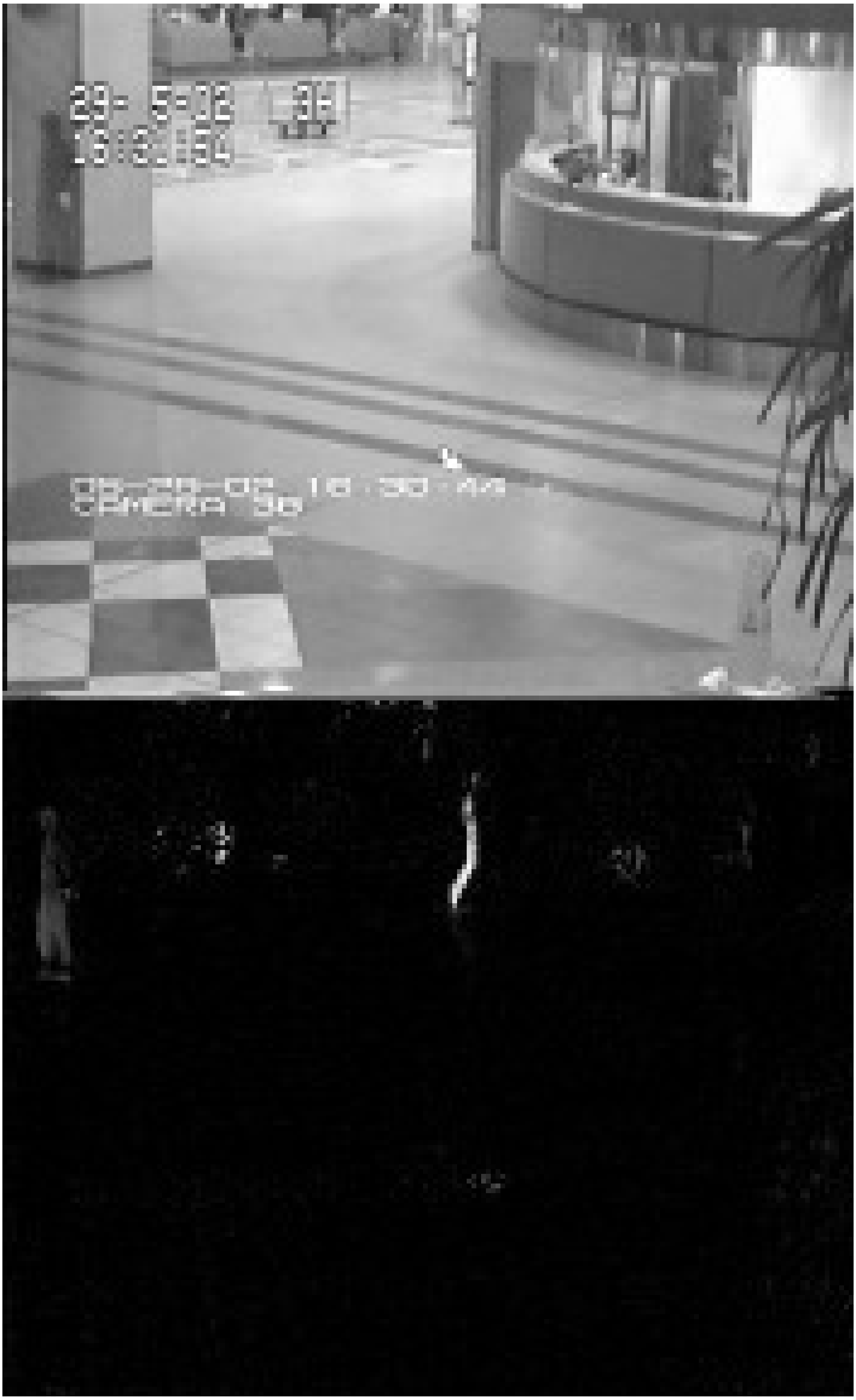


Figure 3.15: Background subtraction by suggestion: background in row $\boldsymbol{I}$ and segmentaion in row $\boldsymbol{II}$.

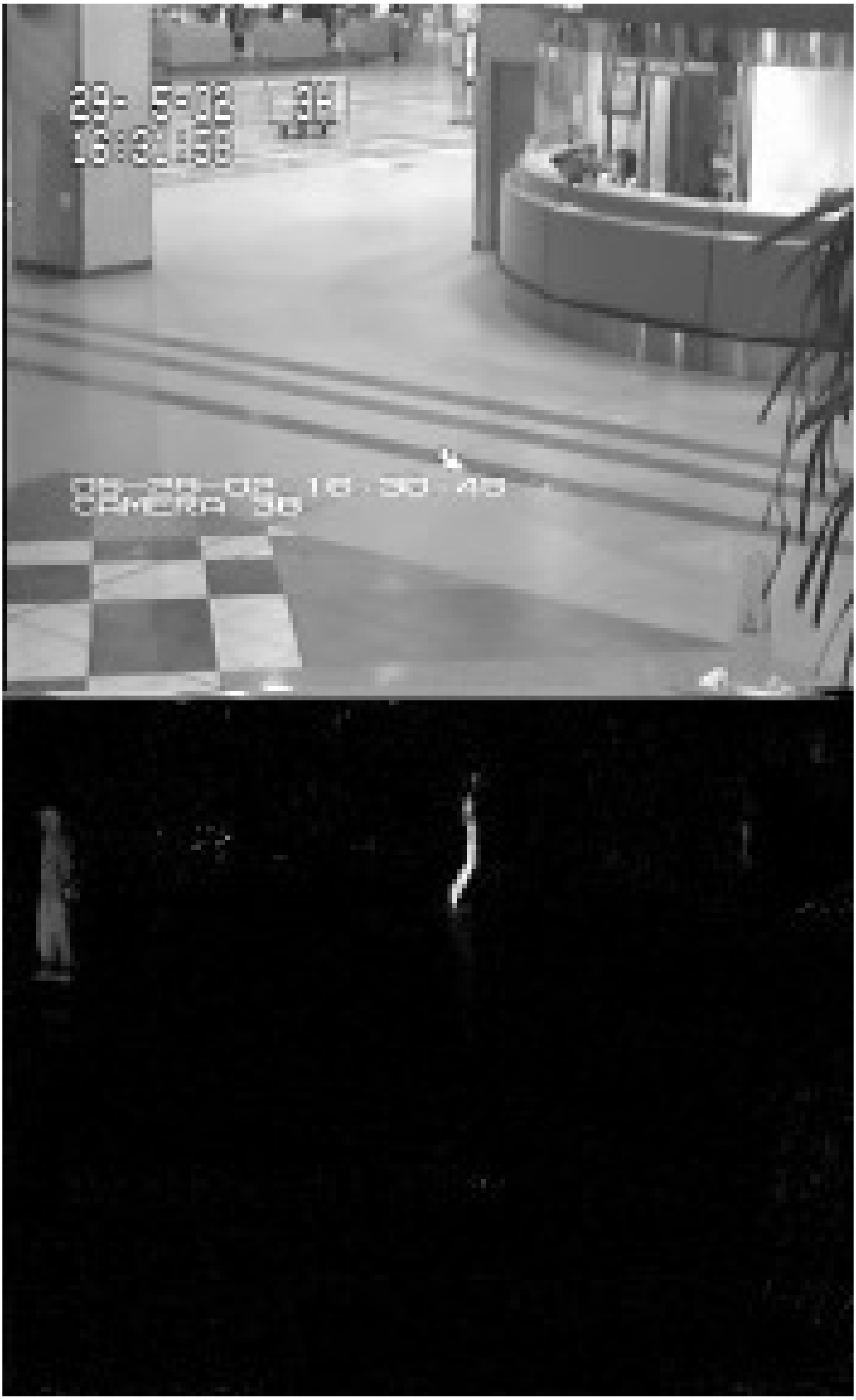


Figure 3.16: Background subtraction by PCPS: background in row $\boldsymbol{I}$ and segmentaion in row $\boldsymbol{II}$.

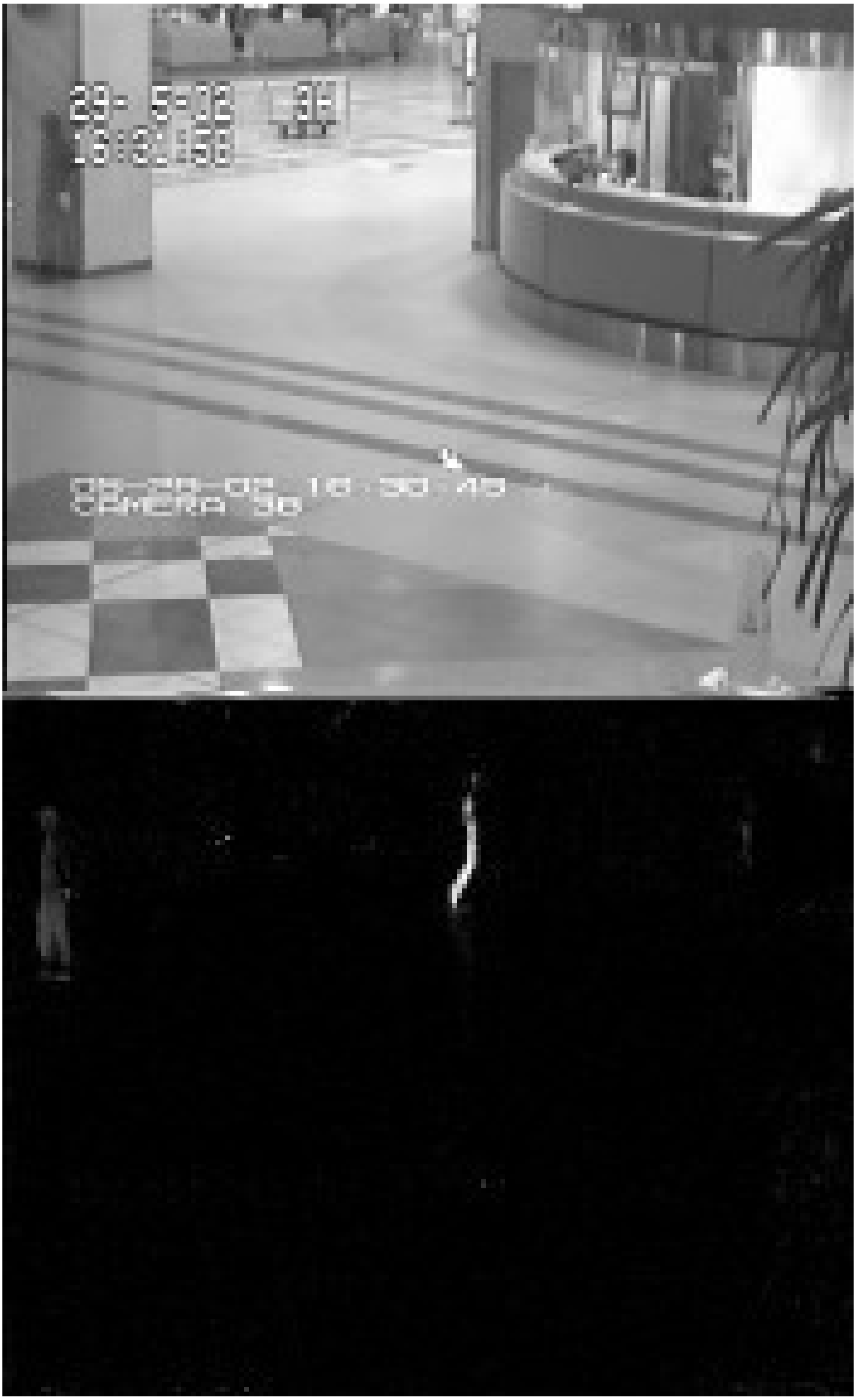


Figure 3.17: Background subtraction by PCP: background in row $\boldsymbol{I}$ and segmentaion in row $\boldsymbol{II}$.

CHAPTER 4

Side Information for Face Completion: a Robust PCA Approach

4.1 Introduction

UV space embeds the manifold of a 3D face into a 2D contiguous atlas. Contiguous UV spaces are natural products of many 3D scanning devices and are often used by 3D Morphable Model (3DMM) construction [9, 13, 87]. Although UV space by nature cannot be constructed from an arbitrary 2D image, a UV map can still be obtained by fitting a 3DMM to the image and sampling the corresponding texture [12]. We illustrate this procedure in Figure 4.1. Unfortunately, due to self-occlusion of the face, those UV maps are often incomplete and lack facial parts that are informative. Once completed, this UV map, combined with the corresponding 3D face, is extremely useful, as it can be used to synthesise 2D faces of arbitrary poses. Afterwards, we can probe image pairs of similar poses to improve recognition performance [22]. Hence, the success of pose-invariant face recognition relies on the quality of UV map completion.

Recovering UV maps from a sequence of related facial frames is a challenging task because self-occlusion at large poses leads to incomplete and missing data. Meanwhile, the imperfection in fitting leads to regional errors. We adapt the approach of robust principal component analysis (RPCA) with missing data [109] to address this difficult problem. In other words, we operate directly on the images themselves rather than on their labels [27]. Principal Component Pursuit (PCP) as proposed in [21, 24] and its variants e.g., [4, 6, 19, 71, 72, 133, 143] are popular algorithms to solve RPCA. PCP employs the nuclear norm and the l_1-norm (convex surrogates of the rank and sparsity constraints, respectively) in order to approximate the original l_0-norm regularised rank minimisation problem. Unavoidably, PCP operates in an isolated manner where domain-dependent prior knowledge [55], i.e.,

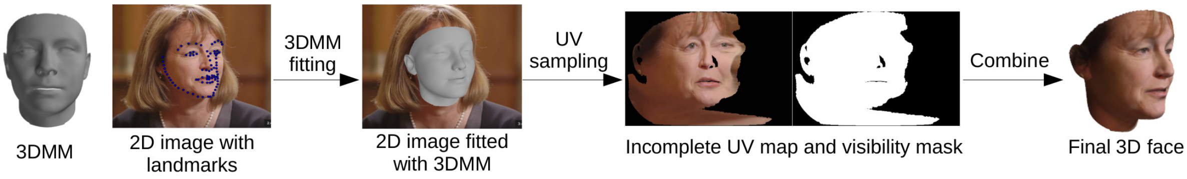


Figure 4.1: The procedure of getting the UV map from an arbitrary 2D image.

side information [129], is always ignored. Moreover, real-world visual data rarely satisfies the stringent assumptions imposed by PCP for exact recovery [20]. These call for a more powerful framework that can assimilate useful priors to alleviate the degenerate or suboptimal solutions of PCP.

It has already been shown that side information is propitious in the context of matrix completion [31,

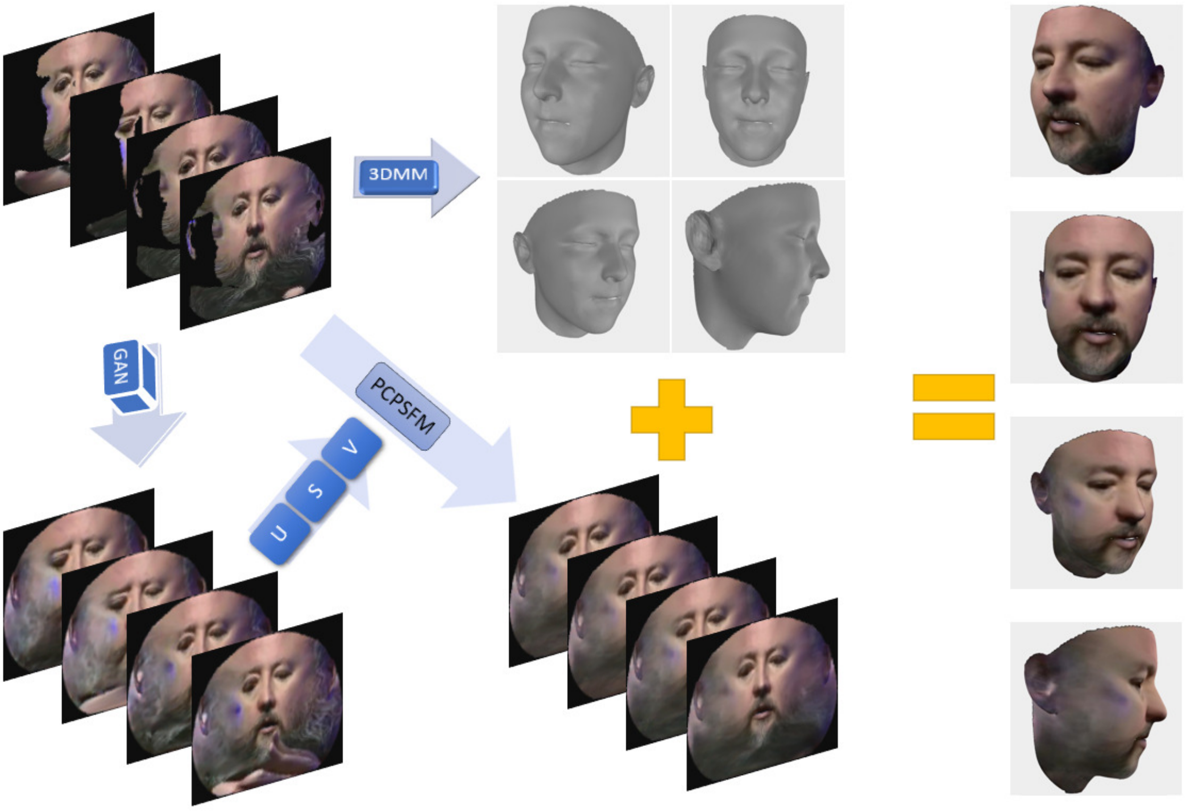


Figure 4.2: Given an input sequence of incomplete UV maps, we extract the shape using 3DMM and perform preliminary completion using GAN. With the left subspace and side information provided by GAN, we then carry out PCPSFM to produce more refined completion results. After that, we attach the completed UV texture to the shape creating images at various poses for face recognition.

134] and compressed sensing [80]. Recently, *noiseless* features have been capitalised on in the PCP framework [33, 68, 69, 101]. In particular, an error-free orthogonal column space was used to drive a person-specific facial deformable model [101]. And such features can also remove dependency on the row-coherence which is beneficial in the case of a union of multiple subspaces [67–69, 73]. More generally, Chiang et al. [33] used both a column and a row space to recover only the weights of their interaction in a simpler problem. The main hindrance to the success of these methods is the need for a set of clean, noise-free data samples in order to determine the column and/or row spaces of the low-rank component. But there are no prescribed way to find them in practice.

On a separate note, rapid advances in neural networks for image inpainting offer an agglomeration of useful priors. Pathak et al. [89] proposed to use context encoders with a reconstruction and an adversarial loss to generate contents for the missing regions that comply with the neighbourhood. Yang et al. [136] further improved inpainting with a multi-scale neural patch synthesis method. This approach is based on a joint optimisation of image content and texture constraints, which not only preserves contextual structures but also produces fine details. Li et al. [64] combined a reconstruction loss, two adversarial losses, and a semantic parsing loss to ensure genuineness and consistency of local-global contents. These methods are by no means definitive for the following reasons: (a) their masks are artificial and do not have semantic correspondence with a 3D face; (b) they do not allow missing regions to be over 50% which is commonplace in our case.

This research is based on our preliminary work [135] but has been extended to 1) the problem of UV completion and 2) to incorporate side information provided by generative adversarial networks. As such, we have extended PCP to take advantage of *noisy* prior information aiming to realise better UV map reconstruction. We then perform pose-invariant face recognition experiments using the completed UV maps. Experimental results indicate the superiority of our framework. The overall workflow is explicated in Figure 4.2. Our contributions are summarised as follows:

- A novel convex program is proposed to use side information, which is a noisy approximation of the low-rank component, within the PCP framework. The proposed method is able to handle missing values while the developed optimisation algorithm has convergence guarantees.
- Furthermore, we extend our proposed PCP model using side information to exploit prior knowledge regarding the column and row spaces of the low-rank component in a more general algorithmic framework.
- In the case of UV completion, we suggest the use of generative adversarial networks to provide subspace features and side information, resulting in a seamless integration of deep learning into

the robust PCA framework.

- We demonstrate the applicability and effectiveness of the proposed approaches on synthetic data as well as on facial image denoising, UV texture completion and pose-invariant face recognition experiments with both quantitative and qualitative evaluation.

The remainder of this chapter is organised as follows. We discuss relevant literature in Section 4.2, while the proposed robust principal component analysis using side information with missing values (PCPSM) along with its extension that incorporates features (PCPSFM) is presented in Section 4.3. In Section 4.4, we first evaluate our proposed algorithms on synthetic and real-world data. Then we introduce GAN as a source of features and side information for the subject of UV completion. Finally, face recognition experiments are presented in the last subsection.

4.2 Related work

Recent advances in convolutional neural networks (CNN) also show great promises in visual feature learning. Context encoders (CE) [89] use an encoder-decoder pipeline where the encoder takes an input image with missing regions producing a latent feature representation and the decoder takes the feature representation generating the missing image content. CE uses a joint loss function:

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}, \tag{4.1}$$

where $\mathcal{L}_{rec}$ is the reconstruction loss and $\mathcal{L}_{adv}$ is the adversarial loss. The reconstruction loss is given by:

$$\mathcal{L}_{rec}(\boldsymbol{x}) = \|\boldsymbol{w} \circ (\boldsymbol{x} - F((\mathbf{1} - \boldsymbol{w}) \circ \boldsymbol{x})\|_2^2, \tag{4.2}$$

where $\boldsymbol{w}$ is a binary mask, $\boldsymbol{x}$ is an example image and CE produces an output $F(\boldsymbol{x})$. The adversarial loss is based on Generative Adversarial Networks (GAN). GAN learns both a generative model G_i from noise distribution $\mathcal{Z}$ to data distribution $\mathcal{X}$ and a discriminative model D_i by the following objective:

$$\mathcal{L}_{a_i} = \min_{G_i}\max_{D_i} \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}[\log(D_i(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{z}\in\mathcal{Z}}[\log(1 - D_i(G_i(\boldsymbol{z})))]. \tag{4.3}$$

For CE, the adversarial loss is modified to

$$\mathcal{L}_{adv} = \max_{D} \mathbb{E}_{\boldsymbol{x}\in\mathcal{X}}[\log(D(\boldsymbol{x})) + \log(1 - D(F((\mathbf{1} - \boldsymbol{w}) \circ \boldsymbol{x})))]. \tag{4.4}$$

Generative face completion [64] uses two discriminators instead with the following objective

$$\mathcal{L} = \mathcal{L}_r + \lambda_1\mathcal{L}_{a_1} + \lambda_2\mathcal{L}_{a_2} + \lambda_3\mathcal{L}_p, \tag{4.5}$$

where $\mathcal{L}_p$ is a parsing loss of pixel-wise softmax and $\mathcal{L}_r$ is the reconstruction loss between the estimated UV texture I_{ij} and the ground truth texture I^*_{ij} of width W and height H

$$\mathcal{L}_p = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} \left| I_{ij} - I^*_{ij} \right| . \tag{4.6}$$

Patch synthesis [136] optimises a loss function of three terms: the holistic content term, the local texture term and the TV-loss term. The content constraint penalises the l_2 difference between the optimisation result and the previous content prediction

$$\mathcal{L}_c = \|\boldsymbol{w} \circ (\boldsymbol{x} - \boldsymbol{x}_i)\|_2^2, \tag{4.7}$$

where $\boldsymbol{x}_i$ is the optimisation result from the last iteration at a coarser scale. The texture constraint penalises the texture appearance across the hole,

$$\mathcal{L}_t = \frac{1}{|\boldsymbol{w}^\phi|} \sum_{i \in \boldsymbol{w}^\phi} \|\boldsymbol{P}_i \circ \boldsymbol{\phi}(\boldsymbol{x}) - \boldsymbol{P}_{nn(i)} \circ \boldsymbol{\phi}(\boldsymbol{x})\|_2^2, \tag{4.8}$$

where $\boldsymbol{w}^\phi$ is the corresponding mask in the VGG-19 feature map $\boldsymbol{\phi}(\boldsymbol{x})$, $|\boldsymbol{w}^\phi|$ denotes the number of patches sampled in $\boldsymbol{w}^\phi$, $\boldsymbol{P}_i$ is the local neural patch at location i, and $nn(i)$ is the nearest neighbor of i. Last, the TV loss encourages smoothness:

$$\mathcal{L}_{TV} = \sum_{i,j \in \boldsymbol{w}^\phi} ((\boldsymbol{x}_{i,j+1} - \boldsymbol{x}_{i,j})^2 + (\boldsymbol{x}_{i+1,j} - \boldsymbol{x}_{i,j})^2). \tag{4.9}$$

4.3 Models

In this section, we propose models of RPCA using side information. In particular, we incorporate side information into PCP by using the trace distance of the difference between the low-rank component and the noisy estimate, which can be seen as a generalisation of compressed sensing with prior information where l_1 norm has been used to minimise the distance between the target signal and side information [80].

4.3.1 The PCPSM and PCPSFM models

Assuming that a noisy estimate of the low-rank component of the data $\boldsymbol{S} \in \mathbb{R}^{n_1 \times n_2}$ is available, we propose the following model of PCP using side information with missing values (PCPSM):

$$\begin{aligned} \min_{\boldsymbol{L},\boldsymbol{E}} \quad & \|\boldsymbol{L}\|_* + \alpha\|\boldsymbol{L} - \boldsymbol{S}\|_* + \lambda\|\boldsymbol{W} \circ \boldsymbol{E}\|_1 \\ \text{s. t.} \quad & \boldsymbol{X} = \boldsymbol{L} + \boldsymbol{E}, \end{aligned} \tag{4.10}$$

where $\alpha > 0, \lambda > 0$ are parameters that weigh the effects of side information and noise sparsity.

The proposed PCPSM can be revamped to generalise the previous attempt of PCPF by the following objective of PCP using side information with features and missing values (PCPSFM):

$$\begin{aligned} \min_{\boldsymbol{H},\boldsymbol{E}} \quad & \|\boldsymbol{H}\|_* + \alpha\|\boldsymbol{H} - \boldsymbol{D}\|_* + \lambda\|\boldsymbol{W} \circ \boldsymbol{E}\|_1 \\ \text{s.t.} \quad & \boldsymbol{X} = \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal + \boldsymbol{E}, \quad \boldsymbol{D} = \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V}, \end{aligned} \tag{4.11}$$

where $\boldsymbol{H} \in \mathbb{R}^{d_1 \times d_2}, \boldsymbol{D} \in \mathbb{R}^{d_1 \times d_2}$ are bilinear mappings for the recovered low-rank matrix $\boldsymbol{L}$ and side information $\boldsymbol{S}$ respectively. Note that the low-rank matrix $\boldsymbol{L}$ is recovered from the optimal solution $(\boldsymbol{H}^*, \boldsymbol{E}^*)$ to objective (4.11) via $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{H}^*\boldsymbol{V}^\intercal$. If side information $\boldsymbol{S}$ is not available, PCPSFM reduces to PCPF with missing values by setting α to zero. If the features $\boldsymbol{U}, \boldsymbol{V}$ are not present either, PCP with missing values can be restored by fixing both of them at identity. However, when only the side information $\boldsymbol{S}$ is accessible, objective (4.11) is transformed back into PCPSM.

4.3.2 The algorithm

If we substitute $\boldsymbol{B}$ for $\boldsymbol{H} - \boldsymbol{D}$ and orthogonalise $\boldsymbol{U}$ and $\boldsymbol{V}$, the optimisation problem (4.11) is identical to the following convex but non-smooth problem:

$$\begin{aligned} \min_{\boldsymbol{H},\boldsymbol{E}} \quad & \|\boldsymbol{H}\|_* + \alpha\|\boldsymbol{B}\|_* + \lambda\|\boldsymbol{W} \circ \boldsymbol{E}\|_1 \\ \text{s.t.} \quad & \boldsymbol{X} = \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal + \boldsymbol{E}, \quad \boldsymbol{B} = \boldsymbol{H} - \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V}, \end{aligned} \tag{4.12}$$

which is amenable to the multi-block alternating direction method of multipliers (ADMM).

The corresponding augmented Lagrangian of (4.12) is:

$$\begin{aligned} l(\boldsymbol{H}, \boldsymbol{B}, \boldsymbol{E}, \boldsymbol{Z}, \boldsymbol{N}) &= \|\boldsymbol{H}\|_* + \alpha\|\boldsymbol{B}\|_* + \lambda\|\boldsymbol{W} \circ \boldsymbol{E}\|_1 \\ &+ \langle \boldsymbol{Z}, \boldsymbol{X} - \boldsymbol{E} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal \rangle + \frac{\mu}{2}\|\boldsymbol{X} - \boldsymbol{E} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal\|_F^2 \\ &+ \langle \boldsymbol{N}, \boldsymbol{H} - \boldsymbol{B} - \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V} \rangle + \frac{\mu}{2}\|\boldsymbol{H} - \boldsymbol{B} - \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V}\|_F^2, \end{aligned} \tag{4.13}$$

where $\boldsymbol{Z} \in \mathbb{R}^{n_1 \times n_2}$ and $\boldsymbol{N} \in \mathbb{R}^{d_1 \times d_2}$ are Lagrange multipliers and μ is the learning rate.

The ADMM operates by carrying out repeated cycles of updates till convergence. During each cycle, $\boldsymbol{H}, \boldsymbol{B}, \boldsymbol{E}$ are updated serially by minimising (4.13) with other variables fixed. Afterwards, Lagrange multipliers $\boldsymbol{Z}, \boldsymbol{N}$ are updated at the end of each iteration. Direct solutions to the single variable minimisation subproblems rely on the shrinkage and the singular value thresholding operators [21]. Let $\mathcal{S}_\tau(a) \equiv \operatorname{sgn}(a)\max(|a| - \tau, 0)$ serve as the shrinkage operator, which naturally extends to matrices, $\mathcal{S}_\tau(\boldsymbol{A})$, by applying it to matrix $\boldsymbol{A}$ element-wise. Similarly, let $\mathcal{D}_\tau(\boldsymbol{A}) \equiv \boldsymbol{M}\mathcal{S}_\tau(\boldsymbol{\Sigma})\boldsymbol{Y}^\intercal$ be

Algorithm 6 ADMM solver for PCPSFM

```
Input: Observation $\boldsymbol{X}$, mask $\boldsymbol{W}$, side information $\boldsymbol{S}$, features $\boldsymbol{U}, \boldsymbol{V}$, parameters $\alpha, \lambda > 0$, scaling ratio $\beta > 1$.
1: Initialize: $\boldsymbol{Z} = \boldsymbol{0}$, $\boldsymbol{N} = \boldsymbol{B} = \boldsymbol{H} = \boldsymbol{0}$, $\beta = \frac{1}{\|\boldsymbol{X}\|_2}$.
2: while not converged do
3:     $\boldsymbol{E} = \mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{X} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal + \frac{1}{\mu}\boldsymbol{Z}) \circ \boldsymbol{W} + (\boldsymbol{X} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal + \frac{1}{\mu}\boldsymbol{Z}) \circ (\boldsymbol{1} - \boldsymbol{W})$
4:     $\boldsymbol{H} = \boldsymbol{U}^\intercal \mathcal{D}_{\frac{1}{2\mu}}(\frac{1}{2}(\boldsymbol{X} - \boldsymbol{E} + \frac{1}{\mu}\boldsymbol{Z} + \boldsymbol{U}(\boldsymbol{B} + \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V} - \frac{1}{\mu}\boldsymbol{N})\boldsymbol{V}^\intercal))\boldsymbol{V}$
5:     $\boldsymbol{B} = \mathcal{D}_{\alpha\mu^{-1}}(\boldsymbol{H} - \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V} + \frac{1}{\mu}\boldsymbol{N})$
6:     $\boldsymbol{Z} = \boldsymbol{Z} + \mu(\boldsymbol{X} - \boldsymbol{E} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal)$
7:     $\boldsymbol{N} = \boldsymbol{N} + \mu(\boldsymbol{H} - \boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V})$
8:     $\mu = \mu \times \beta$
9: end while
Return: $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal$, $\boldsymbol{E}$
```

the singular value thresholding operator on real matrix $\boldsymbol{A}$, with $\boldsymbol{A} = \boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{Y}^\intercal$ being the singular value decomposition (SVD) of $\boldsymbol{A}$.

Minimising (4.13) w.r.t. $\boldsymbol{H}$ at fixed $\boldsymbol{B}, \boldsymbol{E}, \boldsymbol{Z}, \boldsymbol{N}$ is equivalent to the following:

$$\arg\min_{\boldsymbol{H}} \ \|\boldsymbol{H}\|_* + \mu\|\boldsymbol{P} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal\|_F^2, \tag{4.14}$$

where $\boldsymbol{P} = \frac{1}{2}(\boldsymbol{X} - \boldsymbol{E} + \frac{1}{\mu}\boldsymbol{Z} + \boldsymbol{U}(\boldsymbol{B} + \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V} - \frac{1}{\mu}\boldsymbol{N})\boldsymbol{V}^\intercal)$. Its solution is shown to be $\boldsymbol{U}^\intercal \mathcal{D}_{\frac{1}{2\mu}}(\boldsymbol{P})\boldsymbol{V}$. Furthermore, for $\boldsymbol{B}$,

$$\arg\min_{\boldsymbol{B}} \ l = \arg\min_{\boldsymbol{B}} \ \alpha\|\boldsymbol{B}\|_* + \frac{\mu}{2}\|\boldsymbol{Q} - \boldsymbol{B}\|_F^2, \tag{4.15}$$

where $\boldsymbol{Q} = \boldsymbol{H} - \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V} + \frac{1}{\mu}\boldsymbol{N}$, whose update rule is $\mathcal{D}_{\frac{\alpha}{\mu}}(\boldsymbol{Q})$, and for $\boldsymbol{E}$,

$$\arg\min_{\boldsymbol{E}} \ l = \arg\min_{\boldsymbol{E}} \ \lambda\|\boldsymbol{W} \circ \boldsymbol{E}\|_1 + \frac{\mu}{2}\|\boldsymbol{R} - \boldsymbol{E}\|_F^2, \tag{4.16}$$

where $\boldsymbol{R} = \boldsymbol{X} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal + \frac{1}{\mu}\boldsymbol{Z}$ with a closed-form solution $\mathcal{S}_{\lambda\mu^{-1}}(\boldsymbol{R}) \circ \boldsymbol{W} + \boldsymbol{R} \circ (\boldsymbol{1} - \boldsymbol{W})$. Finally, Lagrange multipliers are updated as usual:

$$\boldsymbol{Z} = \boldsymbol{Z} + \mu(\boldsymbol{X} - \boldsymbol{E} - \boldsymbol{U}\boldsymbol{H}\boldsymbol{V}^\intercal), \tag{4.17}$$

$$\boldsymbol{N} = \boldsymbol{N} + \mu(\boldsymbol{H} - \boldsymbol{B} - \boldsymbol{U}^\intercal \boldsymbol{S}\boldsymbol{V}). \tag{4.18}$$

The overall algorithm is summarised in Algorithm 6.

4.3.3 Complexity and convergence

Orthogonalisation of the features $\boldsymbol{U}, \boldsymbol{V}$ via the Gram-Schmidt process has an operation count of $O(n_1 d_1^2)$ and $O(n_2 d_2^2)$ respectively. The $\boldsymbol{H}$ update in Step 4 is the most costly step of each iteration in

Algorithm 6. Specifically, the SVD required in the singular value thresholding action dominates with $O(\min(n_1 n_2^2, n_1^2 n_2))$ complexity. Note that this complexity is shared by both of our proposed PCPSM and PCPSFM algorithms, as well as exsiting PCP and LRR algorithms.

A direct extension of the ADMM has been applied to our 3-block separable convex objective. Its global convergence is proved in **Theorem 4**. We have also used the fast continuation technique already applied to the matrix completion problem [119] to increase μ incrementally for accelerated superlinear performance [97]. The cold start initialisation strategies for variables $\boldsymbol{H}, \boldsymbol{B}$ and Lagrange multipliers $\boldsymbol{Z}, \boldsymbol{N}$ are described in [17]. Besides, we have scheduled $\boldsymbol{E}$ to be updated first and taken the initial learning rate μ as suggested in [66]. As for stopping criteria, we have employed the Karush-Kuhn-Tucker (KKT) feasibility conditions. Namely, within a maximum number of 1000 iterations, when the maximum of $\|\boldsymbol{X} - \boldsymbol{E}_k - \boldsymbol{U}\boldsymbol{H}_k\boldsymbol{V}^\intercal\|_F/\|\boldsymbol{X}\|_F$ and $\|\boldsymbol{H}_k - \boldsymbol{B}_k - \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V}\|_F/\|\boldsymbol{X}\|_F$ dwindles from a pre-defined threshold ϵ, the algorithm is terminated, where k signifies values at the k^{th} iteration.

Theorem 4. *Let the iterative squence $\{(\boldsymbol{E}^k, \boldsymbol{H}^k, \boldsymbol{B}^k, \boldsymbol{Z}^k, \boldsymbol{N}^k)\}$ be generated by the direct extension of ADMM, Algorithm 6, then the sequence $\{(\boldsymbol{E}^k, \boldsymbol{H}^k, \boldsymbol{B}^k, \boldsymbol{Z}^k, \boldsymbol{N}^k)$ converges to a Karush-Kuhn-Tucher (KKT) point in the fully observed case.*

Proof. We first show that function $\theta_3(\boldsymbol{x}_3) = \|\boldsymbol{E}\|_1$ is sub-strong monotonic. From [21], we know that $(\boldsymbol{x}_1^*, \boldsymbol{x}_2^*, \boldsymbol{x}_3^*, \boldsymbol{\lambda}^*) = (\boldsymbol{H}_0, \boldsymbol{E}_0, \boldsymbol{B}_0, \boldsymbol{Z}_0)$ is a KKT point, where $\boldsymbol{H}_0 = \boldsymbol{U}^\intercal\boldsymbol{L}_0\boldsymbol{V}$, $\boldsymbol{B}_0 = \boldsymbol{H}_0 - \boldsymbol{U}^\intercal\boldsymbol{S}\boldsymbol{V}$, $(\boldsymbol{Z}_0)_{ij} = \lambda[\text{sgn}(\boldsymbol{E}_0)]_{ij}$, if $(i,j) \in \Omega$ and $|(\boldsymbol{Z}_0)_{ij}| < \lambda$, otherwise. Since $\theta_3(\boldsymbol{x}_3)$ is convex, by definition, we have

$$\theta_3(\boldsymbol{x}_3^*) \geq \theta_3(\boldsymbol{x}_3) + \langle \boldsymbol{y}_3, \boldsymbol{x}_3^* - \boldsymbol{x}_3 \rangle, \quad \forall \boldsymbol{x}_3 \text{ and } \forall \boldsymbol{y}_3 \in \partial\theta_3(\boldsymbol{x}_3). \tag{4.19}$$

Since $\boldsymbol{A}_3$ is identity in (4.12), we have

$$\begin{aligned}
&\theta_3(\boldsymbol{x}_3) - \theta_3(\boldsymbol{x}_3^*) + \langle \boldsymbol{A}_3^\intercal\lambda^*, \boldsymbol{x}_3^* - \boldsymbol{x}_3 \rangle \\
=&\lambda\|\boldsymbol{E}\|_1 - \lambda\|\boldsymbol{E}_0\|_1 + \langle \boldsymbol{Z}_0, \boldsymbol{E}_0 \rangle - \langle \boldsymbol{Z}_0, \boldsymbol{E} \rangle, \\
=&\lambda\|\boldsymbol{E}\|_1 - \langle \boldsymbol{Z}_0, \boldsymbol{E} \rangle \\
\geq&0,
\end{aligned} \tag{4.20}$$

where the third line follows from $(\boldsymbol{Z}_0)_{ij} = \lambda[\text{sgn}(\boldsymbol{E}_0)]_{ij}$ when $(i,j) \in \Omega$ and $(\boldsymbol{E}_0)_{ij} = 0$ when $(i,j) \notin \Omega$, and the fourth line follows from $|(\boldsymbol{Z}_0)_{ij}| \leq \lambda$, $|(\boldsymbol{Z}_0)_{ij}\boldsymbol{E}_{ij}| \leq |(\boldsymbol{Z}_0)_{ij}||E_{ij}|$ and $\|\boldsymbol{E}\|_1 = \sum_{i,j} |\boldsymbol{E}_{ij}|$. As $\boldsymbol{E}$ is bounded, there always exists $\mu > 0$ such that

$$\lambda\|\boldsymbol{E}\|_1 - \langle \boldsymbol{Z}_0, \boldsymbol{E} \rangle \geq \mu\|\boldsymbol{E} - \boldsymbol{E}_0\|_F^2. \tag{4.21}$$

Thus, overall we have

$$\theta_3(\boldsymbol{x}_3) \geq \theta_3(\boldsymbol{x}_3^*) + \langle \boldsymbol{A}_3^\intercal\boldsymbol{\lambda}^*, \boldsymbol{x}_3 - \boldsymbol{x}_3^* \rangle + \mu\|\boldsymbol{E} - \boldsymbol{E}_0\|_F^2. \tag{4.22}$$

Combining with (4.19), we arrive at

$$\langle \boldsymbol{y}_3 - \boldsymbol{A}_3^\intercal \boldsymbol{\lambda}^*, \boldsymbol{x}_3 - \boldsymbol{x}_3^* \rangle \geq \mu |\boldsymbol{x}_3 - \boldsymbol{x}_3^*|^2, \ \forall \boldsymbol{x}_3 \text{ and } \forall \boldsymbol{y}_3 \in \partial\theta_3(\boldsymbol{x}_3), \tag{4.23}$$

which shows that $\|\boldsymbol{E}\|_1$ satisfies the sub-strong monotonicity assumption.

Additionally, $\|\boldsymbol{H}\|_*, \|\boldsymbol{B}\|_*$ are close and proper convex and $\boldsymbol{A}$'s have full column rank. We thus deduce that the direct extension of ADMM, Algorithm 6, applied to objective (4.12) is convergent according to [115]. □

4.4 Experiments

4.4.1 Parameter calibration

In this section, we illustrate the enhancement made by side information through both numerical simulations and real-world applications. First, we explain how parameters used in our implementation are tuned. Second, we compare the recoverability of our proposed algorithms with state-of-the-art methods for incorporating features or dictionary, viz. PCPF [17] and LRR [11] on synthetic data as well as the baseline PCP [9] when there are no features available. Last, we show how powerful side information

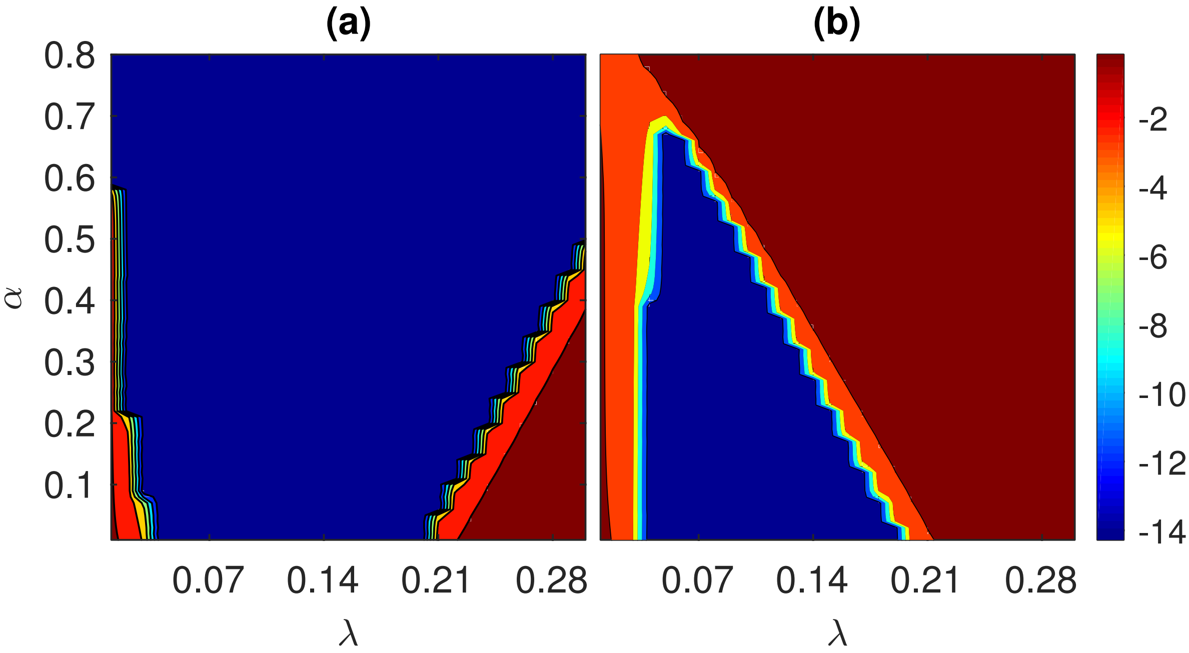


Figure 4.3: Log-scale relative error ($\log \frac{\|\boldsymbol{L}-\boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F}$) of PCPSM **(a)** when side information is perfect ($\boldsymbol{S} = \boldsymbol{L}_0$) and **(b)** when side information is the observation ($\boldsymbol{S} = \boldsymbol{M}$).

can be for the task of UV completion in post-invariant face recognition, where both features and side information are derived from generative adversarial networks.

For LRR, a clean subspace $\boldsymbol{U}$ is used as in [101] instead of the observation $\boldsymbol{X}$ itself as the dictionary. PCP is solved via the inexact ALM [66] and the heuristics for predicting the dimension of principal singular space is not adopted here due to its lack of validity on uncharted real data [51]. We also include Partial Sum of Singular Values (PSSV) [84] in our comparison for its stated advantage in view of the limited number of images available. The stopping criteria for PCPF, LRR, PCP and PSSV are all set to the same KKT optimality conditions for reasons of consistency.

In order to tune the algorithmic parameters, we first conduct a benchmark experiment as follows: a low-rank matrix $\boldsymbol{L}_0$ is generated from $\boldsymbol{L}_0 = \boldsymbol{J}\boldsymbol{K}^\intercal$, where $\boldsymbol{J}, \boldsymbol{K} \in \mathbb{R}^{200\times 10}$ have entries from a $\mathcal{N}(0, 0.005)$ distribution; a 200×200 sparse matrix $\boldsymbol{E}_0$ is generated by randomly setting 38000 entries to zero with others taking values of ± 1 with equal probability; side information $\boldsymbol{S}$ is assumed perfect, that is, $\boldsymbol{S} = \boldsymbol{L}_0$; $\boldsymbol{U}$ is set as the left-singular vectors of $\boldsymbol{L}_0$; and $\boldsymbol{V}$ is set as the right-singular vectors of $\boldsymbol{L}_0$; all entries are observed. It has been found that a scaling ratio $\beta = 1.1$, a tolerance threshold $\epsilon = 10^{-7}$ and a maximum step size $\mu = 10^7$ to avoid ill-conditioning can bring all models except PSSV to convergence with a recovered $\boldsymbol{L}$ of rank 10, a recovered $\boldsymbol{E}$ of sparsity 5% and an accuracy $\|\boldsymbol{L} - \boldsymbol{L}_0\|_F / \|\boldsymbol{L}_0\|_F$ on the order of 10^{-6}. Still, these apply to PSSV as is done similarly in [84].

Although theoretical determination of α and λ is beyond the scope of this work, we nevertheless provide empirical guidance based on extensive experiments. A parameter weep in the $\alpha - \lambda$ space for perfect side information is shown in Figure 4.3(a) and for observation as side information in Figure 4.3(b) to impart a lower bound and a upper bound respectively. It can be easily seen that $\lambda = 1/\sqrt{200}$ (or $\lambda = 1/\sqrt{\max(n_1, n_2)}$ for a general matrix of dimension $n_1 \times n_2$) from Robust PCA works well in both cases. Conversely, α depends on the quality of the side information. When the side information is accurate, a large α should be selected to capitalise upon the side information as much as possible, whereas when the side information is improper, a small α should be picked to sidestep the dissonance caused by the side information. Here, we have discovered that a κ value of 0.2 works best with synthetic data and a value of 0.5 is suited for public video sequences, both of which will be used in all experiments in subsequent sections together with other aforementioned parameter settings. It is worth emphasising again that prior knowledge of the structural information about the data yields more appropriate values for α and λ.

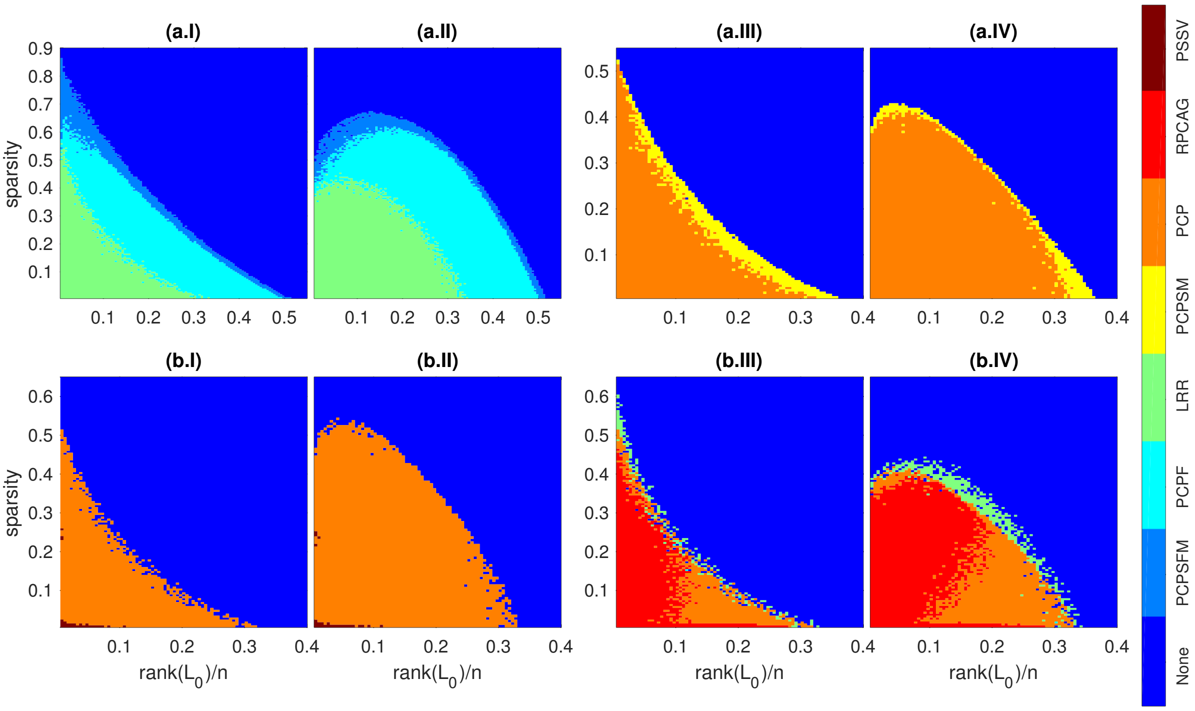


Figure 4.4: Domains of recovery by various algorithms in the fully observed case: **(I,III)** for random signs and **(II,IV)** for coherent signs.

4.4.2 Phase transition on synthetic datasets

We now focus on the recoverability problem, i.e. recovering matrices of varying ranks from errors of varying sparsity. True low-rank matrices are created via $\boldsymbol{L}_0 = \boldsymbol{J}\boldsymbol{K}^\intercal$, where $200 \times r$ matrices $\boldsymbol{J}, \boldsymbol{K}$ have independent elements drawn randomly from a Gaussian distribution of mean 0 and variance $5 \cdot 10^{-3}$, thus r is the rank of $\boldsymbol{L}_0$. Next, we generate 200×200 error matrices $\boldsymbol{E}_0$, which possess $\rho_s \cdot 200^2$ non-zero elements located randomly within the matrix. We consider two types of entries for $\boldsymbol{E}_0$: Bernoulli ± 1 and $\mathcal{P}_\Omega(\text{sgn}(\boldsymbol{L}_0))$, where $\mathcal{P}$ is the projection operator. $\boldsymbol{X} = \boldsymbol{L}_0 + \boldsymbol{E}_0$ thus becomes the simulated observation. For each (r, ρ_s) pair, three observations are constructed. The recovery is successful if for all these three problems, the following criteria regarding the recovered $\boldsymbol{L}$ is met:

$$\frac{\|\boldsymbol{L} - \boldsymbol{L}_0\|_F}{\|\boldsymbol{L}_0\|_F} < 10^{-3}. \tag{4.24}$$

In addition, let $\boldsymbol{L}_0 = \boldsymbol{M}\boldsymbol{\Sigma}\boldsymbol{Y}^\intercal$ be the SVD of $\boldsymbol{L}_0$. Feature $\boldsymbol{U}$ is formed by randomly interweaving column vectors of $\boldsymbol{M}$ with d arbitrary orthonormal bases for the null space of $\boldsymbol{M}^\intercal$, while permuting the expanded columns of $\boldsymbol{Y}$ with d random orthonormal bases for the kernel of $\boldsymbol{Y}^\intercal$ forms feature

$\boldsymbol{V}$. Hence, the feasibility conditions are fulfilled: $\mathbb{C}(\boldsymbol{U}) \supseteq \mathbb{C}(\boldsymbol{L}_0)$, $\mathbb{C}(\boldsymbol{V}) \supseteq \mathbb{C}(\boldsymbol{L}_0^\intercal)$, where $\mathbb{C}$ is the column space operator.

For each trial, we construct the side information by directly adding small Gaussian noise to each element of $\boldsymbol{L}_0$: $L_{ij} \to L_{ij} + \mathcal{N}(0, 2.5r \cdot 10^{-9})$, $i, j = 1, 2, \cdots, 200$. As a result, the standard deviation of the error in each element is 1% of that among the elements themselves. On average, the Frobenius percent error, $\|\boldsymbol{S} - \boldsymbol{L}_0\|_F / \|\boldsymbol{L}_0\|_F$, is 1%. Such side information is genuine in regard to the fact that classical PCA with accurate rank is not able to eliminate the noise [105]. We set d to 10 throughout.

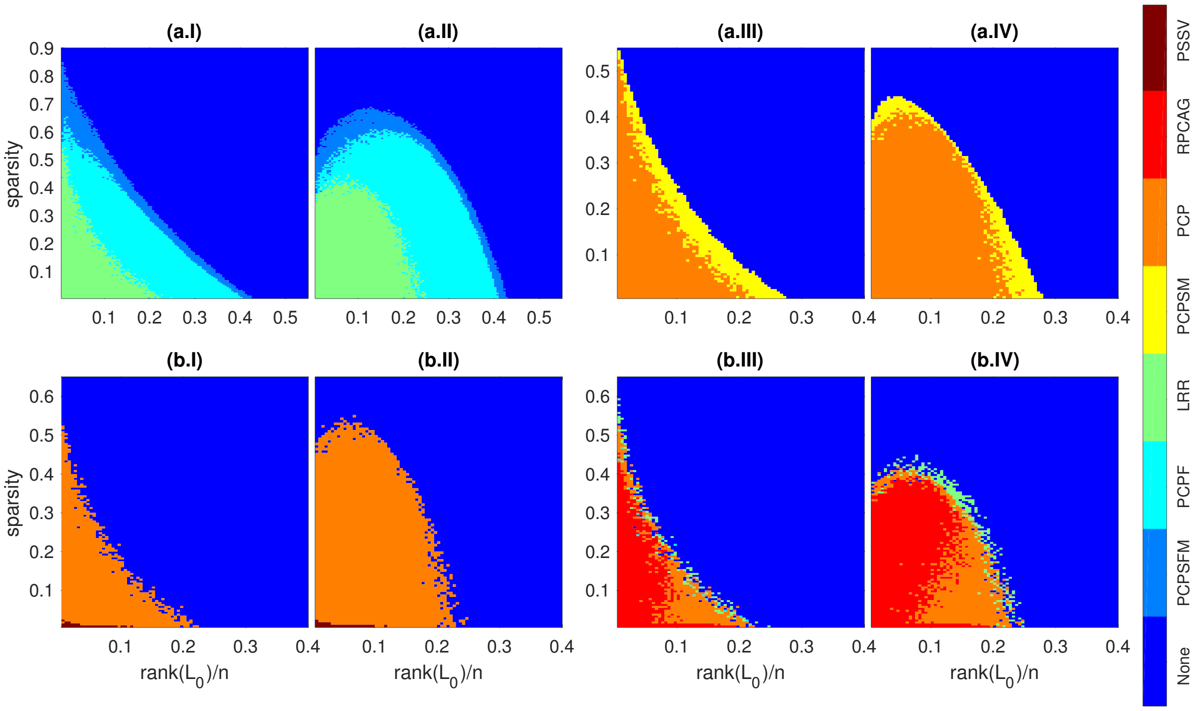


Figure 4.5: Domains of recovery by various algorithms in the partially observed case: **(I,III)** for random signs and **(II,IV)** for coherent signs.

Full observation Figures 4.4 (a.I) and (a.II) plot results from PCPF, LRR and PCPSFM. On the other hand, the situation with no available features is investigated in Figures 4.4 (a.III) and 4.4 (a.IV) for PCP and PCPSM. The frontier of PCPF has been advanced by PCPSFM everywhere for both sign types. Especially at low ranks, errors with much higher density can be removed. Without features, PCPSM surpasses PCP by and large, with significantly more recovery at small sparsity levels for both sign cases. Results from RPCAG and PSSV are worse than PCP with LRR marginally improving (see Figures 4.4(b.I), (b.II), (b.III) and b(IV)).

Partial observation Figures 4.5 (a.I) and (a.II) map out the results for PCPF, LRR and PCPSFM when 10% of the elements are occluded and Figures 4.5 (a.III) and (a.IV) for featureless PCP and PCPSM. In all cases, areas of recovery are reduced. However, there are now larger gaps between PCPF and PCPSFM, so as for PCP and PCPSM. This marks the usefulness of side information particularly in the event of missing observations. We realise that in unrecoverable areas, PCPSM and PCPSFM still obtain much smaller values of $\|\boldsymbol{L} - \boldsymbol{L}_0\|_F$. FRPCAG fails to recover anything at all.

4.4.3 Face denoising

If a surface is convex Lambertian and the lighting is isotropic and distant, then the rendered model spans a 9-D linear subspace [7]. Nonetheless, facial images are only approximately so because facial harmonic planes have negative pixels and real lighting conditions entail unavoidable occlusion and albedo variations. It is thus more reasonable to decompose facial image formation as a low-rank component for face description and a sparse component for defects. In pursuit of this low-rank portrayal, we suggest that there can be further boost to the performance of facial characterisation by leveraging an image which faithfully represents the subject.

We consider images of a fixed pose under different illuminations from the extended Yale B database for testing. All 64 images were studied for each person. 32556×64 observation matrices were formed by vectorising each 168×192 image and the side information was chosen to be the average of all images, tiled to the same size as the observation matrix for each subject. In addition, 5% of the

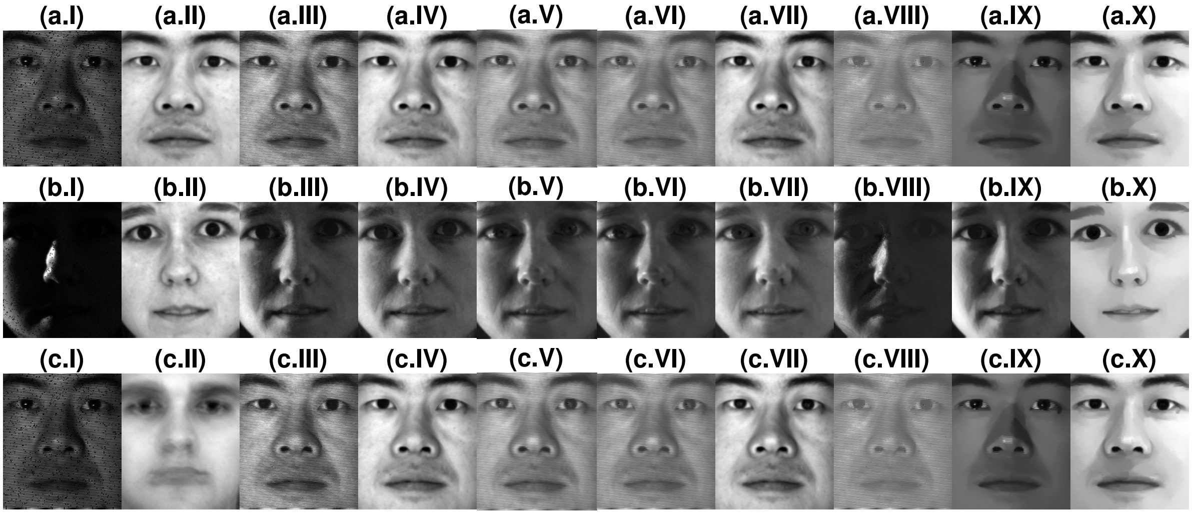


Figure 4.6: Comparison of face denoising ability: **(I)** Observation; **(II)** side information; **(III)** PCP; **(IV)** PCPSM; **(V)** LRR; **(VI)** PCPF; **(VII)** PCPFSM; **(VIII)** PSSV; **(IX)** RPCAG; and **(X)** FRPCAG.

randomly selected pixels within each image were set as missing entries.

For LLR, PCPF and PCPSFM to run, we learn the feature dictionary following an approach by Vishal et al. [88], which is a popular method for extracting high-level attributes [142]. In a nutshell, the feature learning process can be treated as a sparse encoding problem. More specifically, we simultaneously seek a dictionary $\boldsymbol{D} \in \mathbb{R}^{n_1 \times c}$ and a sparse representation $\boldsymbol{B} \in \mathbb{R}^{c \times n_2}$ such that:

$$\min_{\boldsymbol{D},\boldsymbol{B}} \|\boldsymbol{M} - \boldsymbol{D}\boldsymbol{B}\|_F^2 \quad \mathrm{s.\,t.} \quad \gamma_i \leq t \text{ for } i = 1 \ldots n_2, \tag{4.25}$$

where c is the number of atoms, γ_i counts the number of non-zero elements in each sparsity code and t is the sparsity constraint factor. This can be solved by the K-SVD algorithm [2]. Here, feature $\boldsymbol{U}$ is the dictionary $\boldsymbol{D}$ and feature $\boldsymbol{V}$ corresponds to a similar solution using the transpose of the observation matrix as input. For implementation details, we set c to 40, t to 40 and used 10 iterations for each subject.

As a visual illustration, two challenging cases are exhibited in Figure 4.6. For subject #2, it is clearly evident that PCPSM and PCPSFM outperform the best existing methods through the complete elimination of acquisition faults. More surprisingly, PCPSFM even manages to restore the flash in the pupils that is barely present in the side information. For subject #34, PCPSM indubitably reconstructs a more vivid right eye than that from PCP which is only discernible. With that being said, PCPSFM still prevails by uncovering more shadows, especially around the medial canthus of the right eye, and revealing a more distinct crease in the upper eyelid as well a more translucent iris. We further unmask the strength of PCPSM and PCPSFM by considering the stringent side information made of the average of 10 other subjects. Surprisingly, PCPSM and PCPSFM still manage to remove the noise and recover an authentic image (Figure 4.6 (c.IV) and 4.6 (c.VII)). We also notice that PSSV, RPCAG, FRPCAG do not improve upon PCP as in simluation experiments. Thence, we will focus on comparisons with PCP, LRR, PCPF only.

4.4.4 UV map completion

We concern ourselves with the problem of completing the UV texture for each of a sequence of video frames. That is, we apply PCPSM and PCPSFM to a collection of incomplete textures lifted from a video. This parameter-free approach is advantageous to a statistical texture model such as the 3D Morphable Model (3DMM) [10,16] by virtue of its difficulty in reconstructing unseen images captured ’in-the-wild’ (using any commercial cameras in arbitrary conditions).

Texture extraction

Given a 2D image, we extract its UV texture by fitting the 3DMM. More specifically, following [14], three parametric models are employed. These are a 3D shape model (4.26), a texture model (4.27) and a camera model (4.28):

$$\boldsymbol{\mathcal{S}}(\boldsymbol{p}) = \overline{\boldsymbol{s}} + \boldsymbol{U}_s\boldsymbol{p}, \tag{4.26}$$

$$\boldsymbol{\mathcal{T}}(\boldsymbol{\lambda}) = \overline{\boldsymbol{t}} + \boldsymbol{U}_t\boldsymbol{\lambda}, \tag{4.27}$$

$$\boldsymbol{\mathcal{W}}(\boldsymbol{p}, \boldsymbol{c}) = \boldsymbol{\mathcal{P}}(\boldsymbol{\mathcal{S}}(\boldsymbol{p}), \boldsymbol{c}), \tag{4.28}$$

where $\boldsymbol{p} \in \mathbb{R}^{n_s}, \boldsymbol{\lambda} \in \mathbb{R}^{n_t}$ and $\boldsymbol{c} \in \mathbb{R}^{n_c}$ are shape, texture and camera parameters to optimise; $\boldsymbol{U}_s \in \mathbb{R}^{3N\times n_s}$ and $\boldsymbol{U}_t \in \mathbb{R}^{3N\times n_t}$ are the shape and texture eigenbases respectively, with N being the number of vertices in the shape model; $\overline{\boldsymbol{s}} \in \mathbb{R}^{3N}$ and $\overline{\boldsymbol{t}} \in \mathbb{R}^{3N}$ are the corresponding means of shape and texture models, which are learnt from facial scans of 10000 individuals [16]; $\boldsymbol{\mathcal{P}}(\boldsymbol{s}, \boldsymbol{c}) : \mathbb{R}^{3N} \to \mathbb{R}^{2N}$ is a perspective camera transformation function. The complete cost function for 3DMM fitting is:

$$\begin{aligned} \min_{\boldsymbol{p},\boldsymbol{\lambda},\boldsymbol{c}} &\|\boldsymbol{F}(\boldsymbol{\mathcal{W}}(\boldsymbol{p}, \boldsymbol{c})) - \boldsymbol{\mathcal{T}}(\boldsymbol{\lambda})\|^2 + \beta_l\|\boldsymbol{\mathcal{W}}(\boldsymbol{p}, \boldsymbol{c})) - \boldsymbol{s}_l\|^2 \\ &+ \beta_s\|\boldsymbol{p}\|^2_{\boldsymbol{\Sigma}_s^{-1}} + \beta_t\|\boldsymbol{\lambda}\|^2_{\boldsymbol{\Sigma}_t^{-1}}, \end{aligned} \tag{4.29}$$

where $\boldsymbol{F}(\boldsymbol{\mathcal{W}}(\boldsymbol{p}, \boldsymbol{c}))$ denotes the operation of sampling the feature image onto the projected 2D locations. The second term is a landmark term with weighting β_l in order to accelerate in-the-wild 3DMM fitting, where the 2D shape, $\boldsymbol{s}_l$, is provided by [18]. The final two terms are regularisation terms to counter over-fitting, where $\boldsymbol{\Sigma}_s$ and $\boldsymbol{\Sigma}_t$ are diagonal matrices with the main diagonal being eigenvalues of the shape and texture models respectively. Eq. 4.29 is solved by the Gauss-Newton optimisation framework (see [14] for details). We empirically set $\beta_l = 10^5$, $\beta_s = 3 \times 10^6$ and $\beta_t = 1$ following [15, 36]. Note that any landmark localisation techniques [62] can be applied within our framework and the visible mask of facial region is a natural product of the 3DMM fitting process.

Quantitative evaluation

We quantitatively evaluate the completed UV maps by our proposed methods on the 4DFAB dataset [30]. 4DFAB is the first 3D dynamic facial expression dataset designed for biometric applications, where 180 participants are invited to attend four sessions at different times. Hence, to complete UV maps for one session, we can leverage images from another session as side information. For each of 5 randomly selected subjects, one dynamic sequence of 155 frames is randomly cut from the second session. After vectorisation, a 32556×155 observation matrix is formed. To produce UV masks of

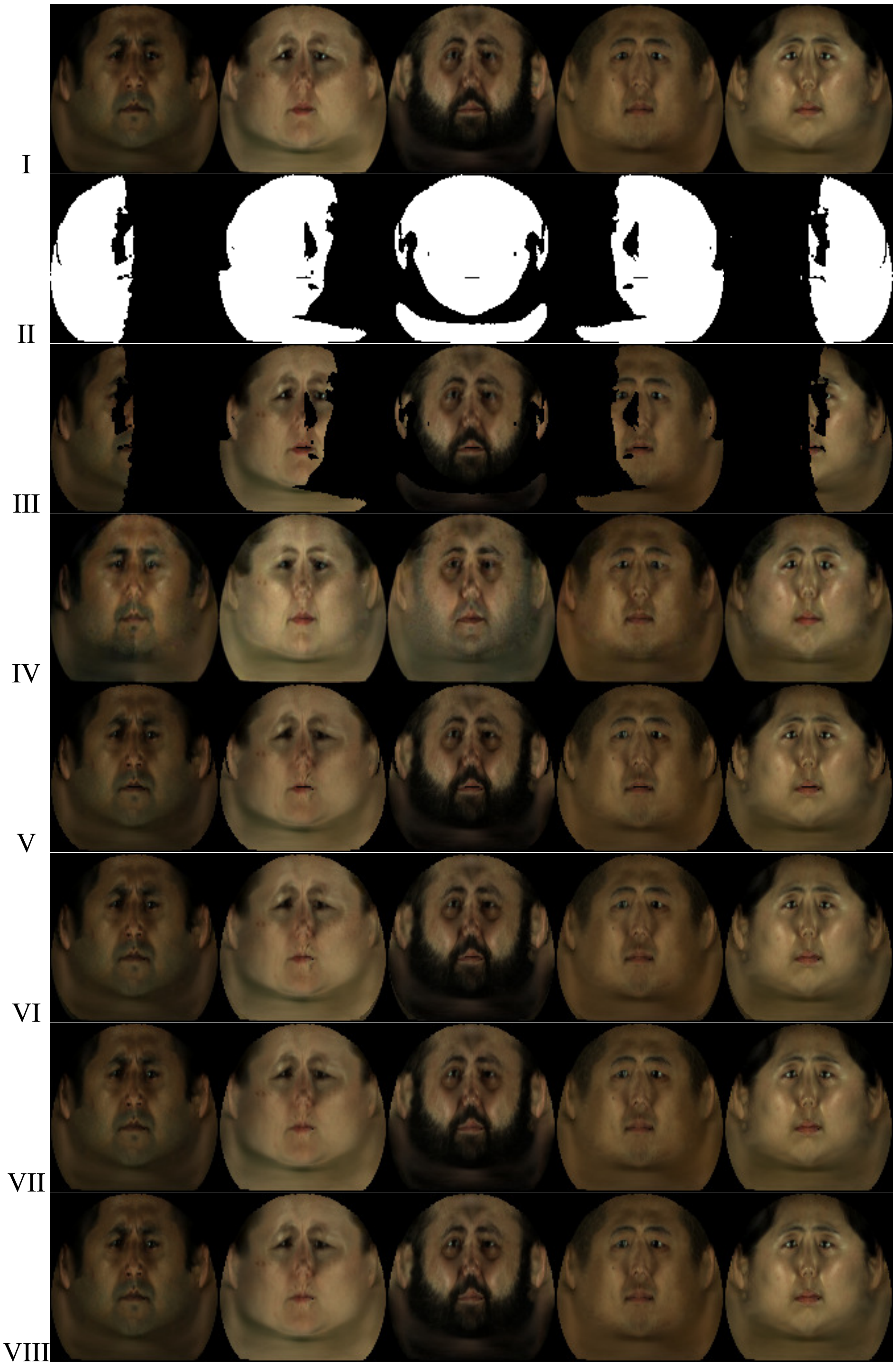


Figure 4.7: (row I) original sequences; (row II) random masks; (row III) sample inputs; (row IV) side information; (row V) PCP; (row VI) PCPSM; (row VII) LRR; (row VIII) PCPSFM.

Table 4.1: Quantitative measures of UV completion by various algorithms on the 4DFAB dataset.

Subject		#1	#2	#3	#4	#5
PSNR (dB)	PCP	35.99 ±0.79	26.75 ±0.88	32.65 ±0.88	31.33 ±0.99	29.10 ±1.68
	PCPSM	**39.56** ±1.30	**30.63** ±1.47	**34.66** ±1.29	**35.86** ±1.85	**32.80** ±2.93
	LRR	40.94 ±2.13	30.69 ±1.71	36.38 ±2.10	35.94 ±2.53	33.97 ±3.93
	PCPSFM	**41.48** ±2.06	**31.46** ±1.69	**37.29** ±2.37	**36.60** ±2.36	**34.80** ±4.14
SSIM	PCP	0.973±0.004	0.922±0.012	0.962±0.010	0.956±0.007	0.949±0.013
	PCPSM	**0.987**±0.004	**0.952**±0.013	**0.969**±0.010	**0.981**±0.006	**0.973**±0.013
	LRR	0.990±0.005	0.952±0.013	0.975±0.010	0.982±0.007	0.978±0.014
	PCPSFM	**0.991**±0.004	**0.958**±0.013	**0.979**±0.010	**0.984**±0.007	**0.981**±0.013

different poses, we rotate each face with different yaw and pitch angles. The yaw angle ranges from $-90°$ to $90°$ in steps of $6°$, whereas the pitch angle is selected from $\{-10°, -5°, 0°, 5°, 10°\}$. Therefore, for each subject, a set of 155 unique masks are generated. We also tiled one image of the same subject from the first session into a 32556×155 matrix as side information. $\boldsymbol{U}$ is provided by the left singular vector of the original sequence while $\boldsymbol{V}$ is set to the identity.

From Figure 4.7, we observe that (I) RPCA approaches can deal with cases where more than 50% of the pixels are missing; (II) imperfect side information (shaved beard, removed earrings and different lightings) still help with the recovery process. We record peak signal-to-noise ratios (PSNR) and structural similarity indices (SSIM) between the completed UV maps and the original maps in Table 4.1. It is evident that with the assistance of side information, much higher fidelity can be achieved. The use of imperfect side information nearly comes on a par with perfect features.

Generative adversarial networks

More often than not, ground-truth $\boldsymbol{U}$, $\boldsymbol{V}$ are not accessible to us for in-the-wild videos. Learning methods such as (4.25) must be leveraged to acquire $\boldsymbol{U}$ or $\boldsymbol{V}$. However, (4.25) is not ideal: (I) it is not robust to errors of arbitrary magnitude; (II) it cannot handle missing values; (III) it requires exhaustive search of optimal parameters which vary from video to video; (IV) it only admits greedy solutions[1]. As a matter of fact, we can use GAN to produce authentic pseudo ground-truth. Then we apply truncated singular value decomposition to the vectorised frames and use the obtained left and right singular vectors as $\boldsymbol{U}$ and $\boldsymbol{V}$ subspace features respectively. Moreover, such completed sequence provides us with good side information. For each color channel, we average the video frames before tiling it back to the original length. This resulting matrix is taken as side information. For GAN, we employ the image-to-image conditional adversarial network [54] (appropriately customised) to conduct

[1] There is a variant of KSVD [76] that can fill holes which are smaller than the size of the atoms. We evaluate it against our GAN-based approach in Figure 3 and Table 1 of the appendix.

UV completion. Details regarding the architecture and training of GAN can be found in the appendix.

Qualitative demonstration

To examine the ability of our proposed methods on in-the-wild images. We perform experiments on the 300VW dataset [112]. This dataset contains 114 in-the-wild videos that exhibit large variations in pose, expression, illumination, background, occlusion, and image quality. Each video shows exactly one person, and each frame is annotated with 68 facial landmarks. We perform 3DMM fitting on these videos and lift one corresponding UV map for each frame, where the visibility mask is produced by z-buffering based on the fitted mesh. Side information is generated by taking the average of the completed UVs from GAN. $\boldsymbol{U}$ and $\boldsymbol{V}$ are assigned to the singular vectors of the completed texture sequence from GAN.

We display results for one sample frame from each of 9 arbitrary videos in Figure 4.8. As evident from the images, GAN alone has unavoidable drawbacks: (I) when 3DMM fitting is not accurate, GAN is unable to correct such defects; (II) when the image itself contains errors, GAN is unable to remove them. On the other hand, PCP often fails to produce a complete UV. PCPSM always produces a completed UV texture, which is an improvement over PCP, but it generates undesirable boundaries. Visually, LRR and PCPSFM have the best performance, being able to produce good completed UVs for a large variety of poses, identities, lighting conditions and facial characteristics. This justifies the quality of subspaces and side information from GAN for use in the robust PCA framework. We also synthesise 2D faces of three different poses using the the completed UV maps in Figure 4.9.

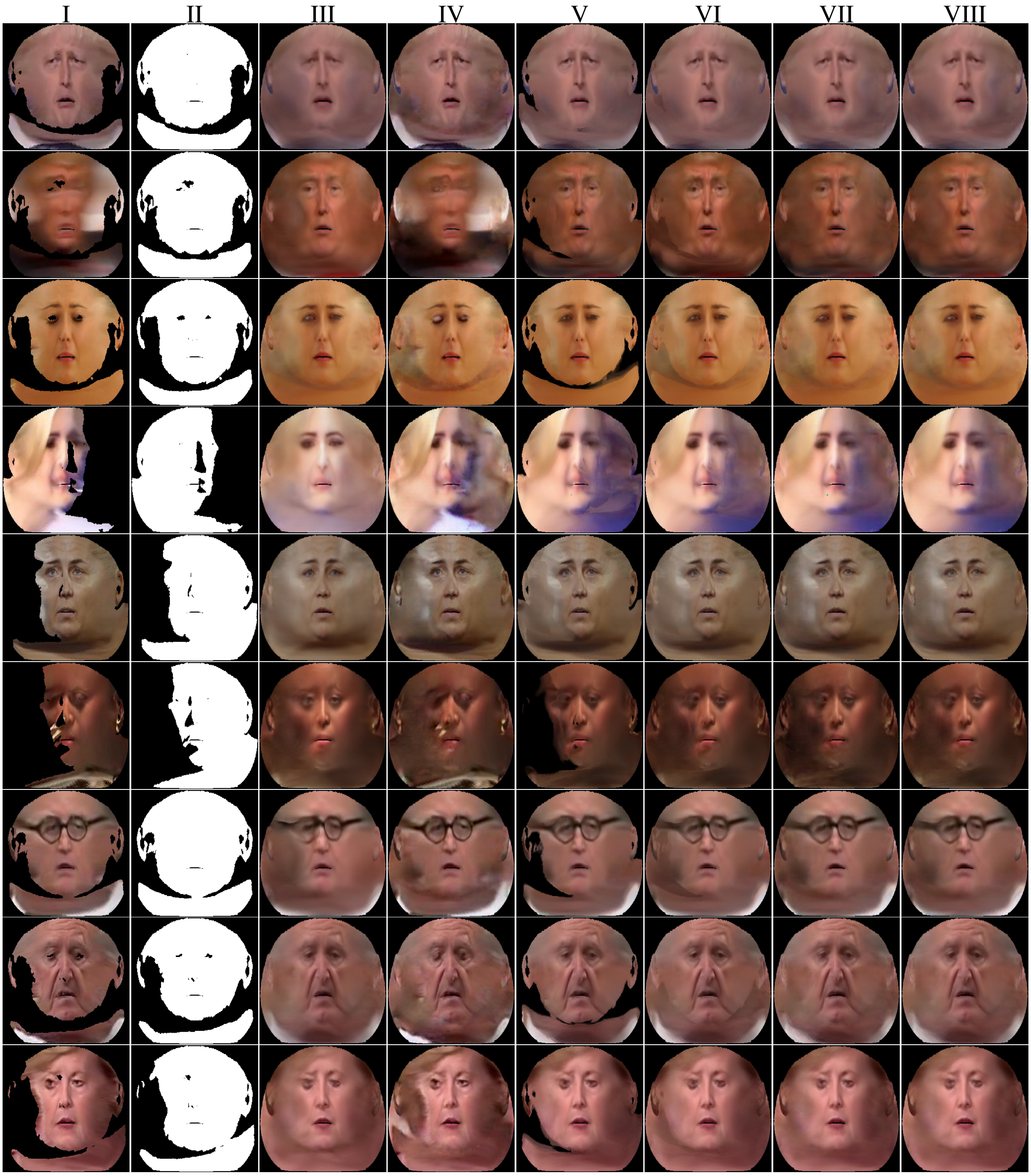


Figure 4.8: 300VW: (column I) sample frame; (column II) mask; (column III) side information; (column IV) GAN; (column V) PCP; (column VI) PCPSM; (column VII) LRR; (column VIII) PCPSFM.

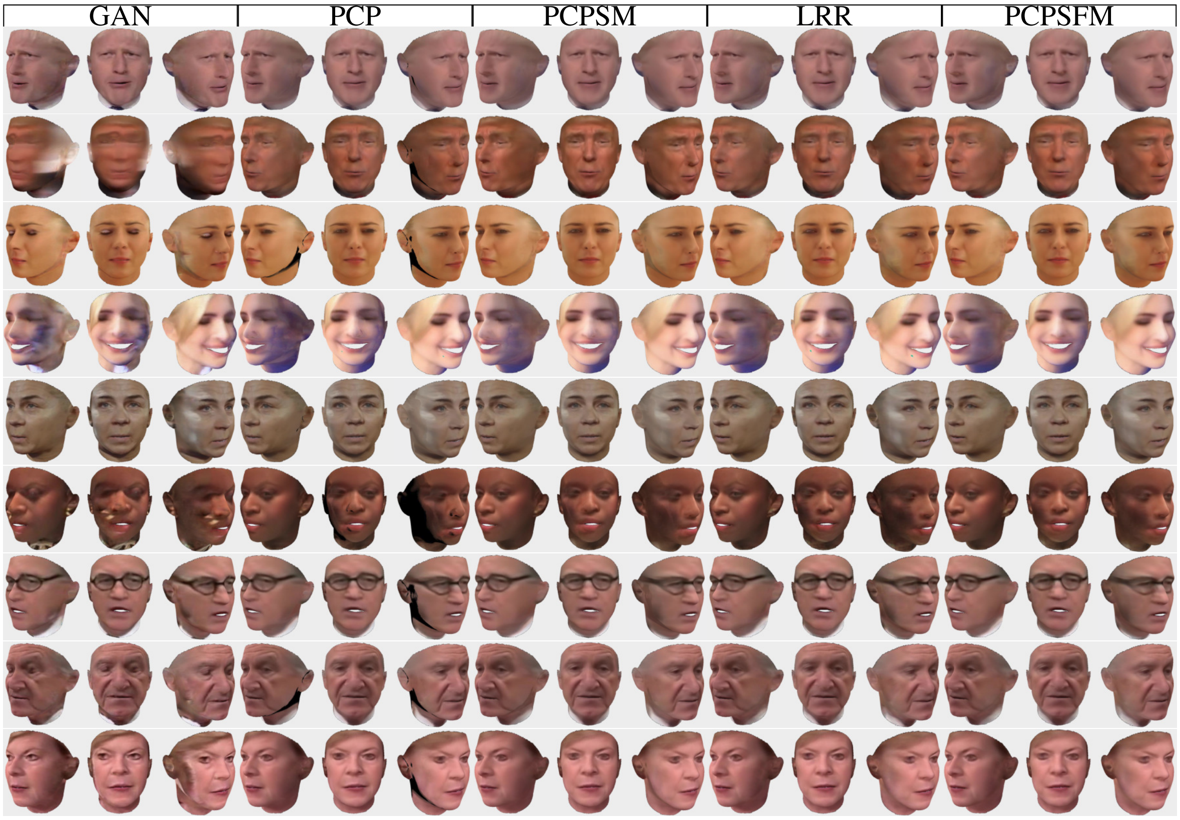


Figure 4.9: 2D face synthesis of three views ($-45°$, $0°$, $45°$) from the completed UV maps by various methods.

4.4.5 Face recognition

Face recognition is a crucial element of biometrics [28, 86, 93, 102, 111, 116, 118]. In this work, we focus on the set-based face verification, i.e. to decide whether two sets of facial images are of the same person or not. One face set could consist of one or multiple samples of the same person (e.g. still images, or frames from a video of the person, or a mixture of both). Therefore, traditional face verification is a special case of the set-based face verification.

The simplest approach to the set-based face verification problem is to generate a feature vector per image, aggregate them into one vector to represent the set (e.g. calculate the feature centre by average), and then compute the cosine similarity between sets. However, the combination rule of averaging is oversimplified since not all face images in one set are of equal importance. The features derived from a profile face is probably of less importance than the features coming from a frontal face as there is signal loss due to self-occlusion under pose variations.

More specifically, we focus on pose-invariant face recognition. Modern approaches to pose-invariant face recognition include pose-robust feature extraction [41], multi-view subspace learning [42], face frontalisation by synthesis [36, 40], etc. Nonetheless, these methods often fall short of expectations either due to fundamental limitations or inability to fuse with other useful methods. For example, Generalised Multi-view Analysis [110] cannot take account of pose normalisation [39] or deep neural network-based pose-robust feature extraction [56], and vice versa. Hence, it is fruitful to provide a framework where information from different perspectives can be fused together to deliver better prediction.

We quantitatively evaluate our proposed fusion methods by carrying out set-based face verification experiments. The experiments are performed on four standard databases, namely CFP [104], IJB [58, 78, 123], YTF [124] and PaSC [8]. Evaluation results on these benchmarks will be given in the next few sections. Overall, the proposed method outperforms current state-of-the-art approaches [23, 86, 102, 130, 131] by a large margin.

Face Feature Embedding

We employ ArcFace [37] with ResNet50 [50] as the backbone. The additive angular margin loss ($m = 0.35$) is used to train a 512-D facial feature embedding model on the VGG2 training set [23], which contains 3,141,890 images from 8,631 identities. Following [37], we use five facial landmarks (eye centres, nose tip and mouth corners) [139] to normalise the face images by similarity transformation. The faces are cropped and resized to 112×112. Figure 4.10 illustrates the set-based face feature embedding used for face verification. For one facial image set, we first extract 3D face shapes and incomplete UV maps via 3DMM fitting [14]. Then, we utilise the proposed UV completion methods (GAN [36], PCP, PCPSM, LRR and PCPSFM) to derive completed UV maps. Frontal faces are synthesised from the full UV maps and the 3D shapes, which are then fed into the feature embedding network. A set of 512-D features from the last fully connected layer of network, is used to compute the feature centre and eventually taken as the feature descriptor.

Evaluation Metrics

In this work, we employ the standard 1:1 verification protocol. The performance is reported by the true accept (positive) rates (TAR) vs. false accept (positive) rates (FAR) (from the receiver operating characteristics (ROC) curve). Following [130], we are interested in the TAR values where FAR=1e-4 and FAR=1e-5, which is also the security level for financial applications. Apart from the ROC curve, we also calculate the best threshold value from the positive and negative pairs, and report the corresponding classification accuracy for each method on the YTF dataset.

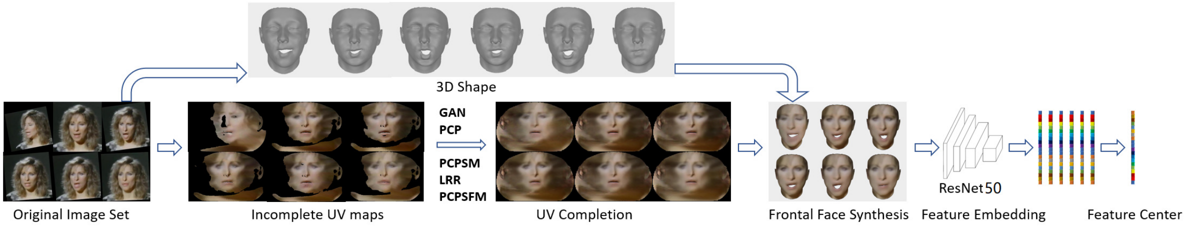


Figure 4.10: The proposed pipeline for video-based face recognition. The 3DMM [14] is fitted on the frames of the video and the incompleted UV maps are estimated. The trained GAN [36] is then used to provide an initial estimate of the side information and the proposed methodology is applied to generate the completed UV maps. The 3D model is reused to render the images in the frontal view. Deep neural network is used to extract features from all frames and the average of the features is used to represent the video.

Ablation Experiments on CFP

The CFP dataset [104] consists of 500 subjects, each of which has 10 frontal and 4 profile images. For each subject, we construct four sets (with 3, 3, 4 and 4 faces respectively) where each set includes at least one profile face. For set-based face verification on CFP, we extensively compare all possible $3,000$ positive pairs and $1,996$K negative pairs.

As shown in Table 4.2 and Figure 4.11, we compare the proposed methods with several baseline methods. It can be clearly observed that by leveraging subspace features or side information from GAN (LRR/PCPSM), we ameliorate the recognition results in terms of TAR over the vanilla PCP, while a further boost in performance can be achieved when both of them are considered together (PCPSFM). Compared to the result of ArcFace, the proposed PCPSFM achieves a TAR improvement of 1.7% at FAR=1e-5.

Table 4.2: Verification TAR on the CFP dataset, the higher TAR the better.

Method	FAR=1e-6	FAR=1e-5	FAR=1e-4
ArcFace	0.901	0.950	0.989
GAN+ArcFace	0.905	0.957	0.991
PCP+ArcFace	0.902	0.953	0.990
LRR+ArcFace	0.911	0.963	0.993
PCPSM+ArcFace	0.907	0.961	0.991
PCPSFM+ArcFace	0.916	0.967	0.993

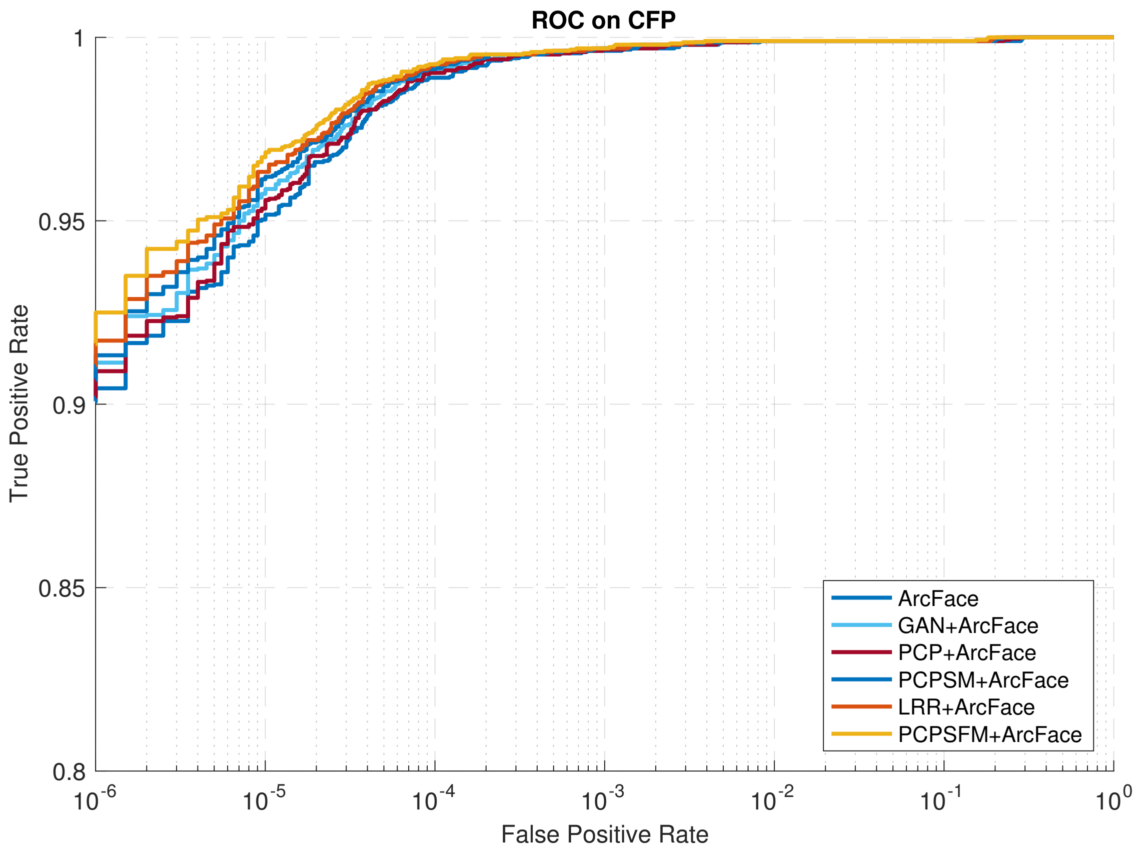


Figure 4.11: ROC curves on the CFP dataset.

Experiments on IJB

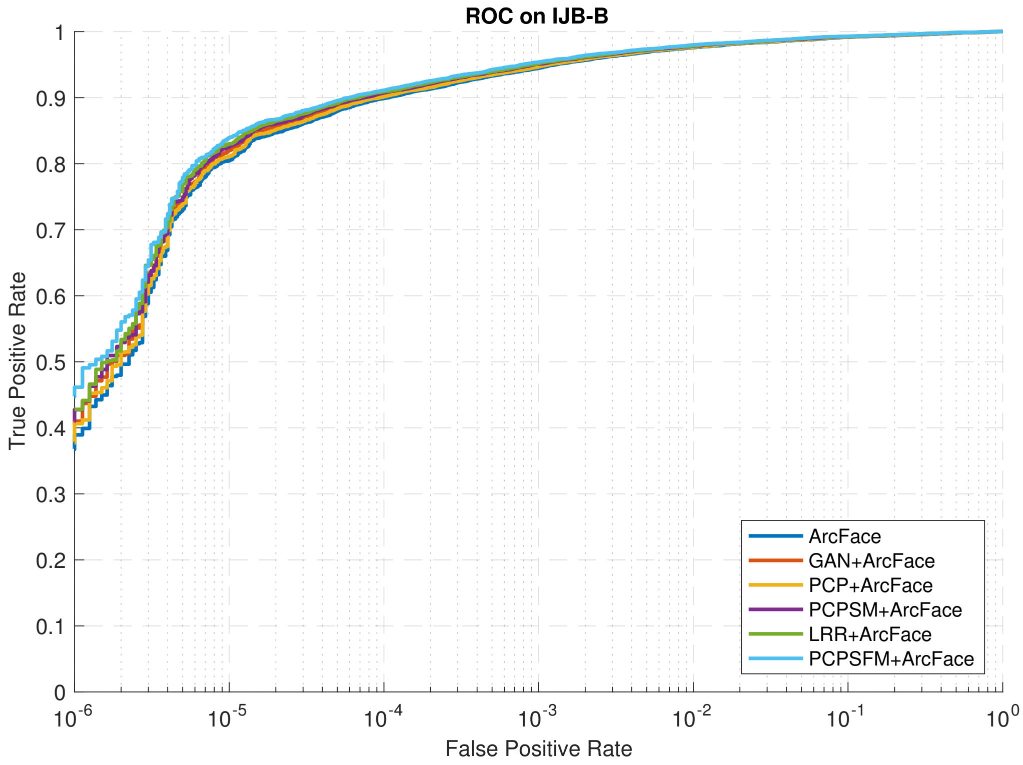
The IARPA Janus Benchmarks have been gradually enlarged from IJB-A [58] to IJB-B [123] and IJB-C [78]. The IJB-A dataset contains $5,712$ images and $2,085$ videos from 500 subjects, with an average of 11.4 images and 4.2 videos per subject. The IJB-B dataset is an extension of IJB-A, which contains $1,845$ subjects with 21.8K still images and 55K frames from $7,011$ videos. In total, there are $12,115$ templates with $10,270$ genuine matches and 8M impostor matches. The IJB-C dataset is a further extension of IJB-B, having $3,531$ subjects with 31.3K still images and 117.5K frames from $11,779$ videos. In total, there are $23,124$ templates with $19,557$ genuine matches and $15,639$K impostor matches. All images and videos from the IARPA Janus Benchmarks are captured under unconstrained environment and show large variations in expression and image qualities. Since IJB-A has been superseded by IJB-B with its images being a subset of IJB-B, we only report the results on IJB-B and IJB-C.

In Figure 4.12, we illustrate the ROC curves of the proposed method against the baselines. We see that ArcFace [37] achieves strong performance. However, PCPSFM further increases the performance through incorporating feature subspace and side information even when there are some low-resolution face images within the template. This is because the proposed method can integrate information from different face images within the template and therefore make the final template feature representation robust. To conduct fair comparison with other methods [23, 130, 131], no flip test and face detection scores are used during evaluation even though both tricks are known to improve the performance.

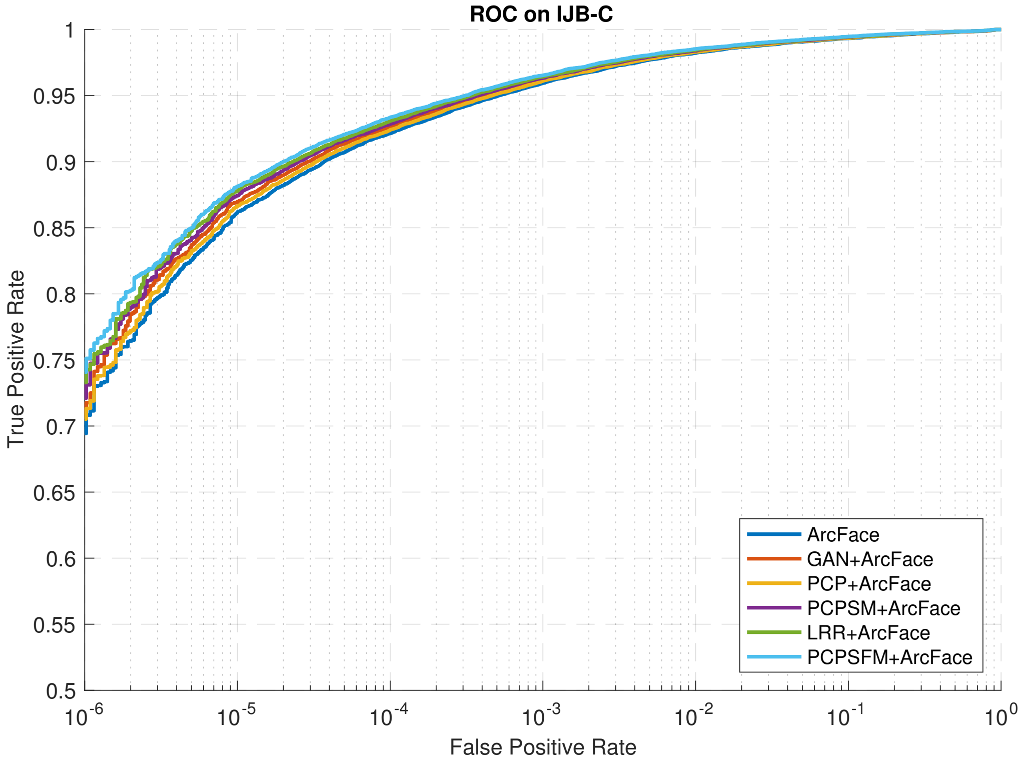
In Table 4.3 and 4.4, comparisons between the proposed PCPSFM and the most recent methods [23, 26, 92, 130, 131] are made. We can see from the results that the baseline method, ArcFace [37], already achieves similar or even better performance compared to the methods proposed in [26, 92]. With the assistance of the proposed PCPSFM, our method achieves the best result on both IJB-B and IJB-C datasets outperforming counterparts [26, 92] even with less identities in the training data and a smaller CNN embedding network.

Table 4.3: 1:1 verification TAR on the IJB-B dataset (Higher is better).

Method	FAR=1e-4	FAR=1e-3
GOTS [123]	0.160	0.330
VGGFaces [86, 123]	0.550	0.720
FPN [26]	0.832	0.916
Light CNN [128]	0.877	0.920
Centre Loss [122]	0.807	0.900
Crystal Loss [92]	0.898	0.944
Whitelam et al. [123]	0.540	0.700
Navaneeth et al. [11]	0.685	0.830
ResNet50 [23]	0.784	0.878
SENet50 [23]	0.800	0.888
ResNet50+SENet50 [23]	0.800	0.887
MN-v [131]	0.818	0.902
MN-vc [131]	0.831	0.909
ResNet50+DCN(Kpts) [130]	0.850	0.927
ResNet50+DCN(Divs) [130]	0.841	0.930
SENet50+DCN(Kpts) [130]	0.846	0.935
SENet50+DCN(Divs) [130]	0.849	0.937
ArcFace [37]	0.899	0.945
GAN+ArcFace	0.904	0.949
PCP+ArcFace	0.901	0.947
PCPSM+ArcFace	0.907	0.951
LRR+ArcFace	0.909	0.952
PCPSFM+ArcFace	**0.911**	**0.954**



(a) ROC for IJB-B



(b) ROC for IJB-C

Figure 4.12: ROC curves of 1:1 verification protocol on the IJB-B and IJB-C dataset.

Table 4.4: 1:1 verification TAR on the IJB-C dataset (Higher is better).

Method	FAR=1e-4	FAR=1e-3
Centre Loss [122]	0.853	0.912
Crystal Loss [92]	0.919	0.957
GOTS [123]	0.160	0.320
FaceNet [102]	0.490	0.660
VGG [86]	0.600	0.750
ResNet50 [23]	0.825	0.900
SENet50 [23]	0.840	0.910
ResNet50+SENet50 [23]	0.841	0.909
MN-v [131]	0.852	0.920
MN-vc [131]	0.862	0.927
ResNet50+DCN(Kpts) [130]	0.867	0.940
ResNet50+DCN(Divs) [130]	0.880	0.944
SENet50+DCN(Kpts) [130]	0.874	0.944
SENet50+DCN(Divs) [130]	0.885	0.947
ArcFace [37]	0.921	0.959
GAN+ArcFace	0.926	0.962
PCP+ArcFace	0.924	0.961
PCPSM+ArcFace	0.928	0.963
LRR+ArcFace	0.931	0.964
PCPSFM+ArcFace	**0.934**	**0.965**

Experiments on YTF

The YouTube Face (YTF) dataset [124] consists of $3,425$ videos from $1,595$ different people. The clip duration varies from 48 frames to $6,070$ frames. The average length is 181.3 frames. We follow the *unrestricted with labelled outside data* protocol and report the results on $5,000$ video pairs ($2,500$ positive pairs and $2,500$ negative pairs).

This dataset is very challenging not only due to the rich pose variations but also the serious compression artifacts. We compare the performance of the proposed method with current state-of-the-art approaches on the YTF dataset. In Table 4.5, we list the verification accuracy for the best-performing deep learning methods. We see that our GAN model alone is among the best reported architectures and it outperforms the classical PCP. Nonetheless, their fusion (PCPSM, LRR and PCPSFM) is superior to either of them. More specifically, PCPSM improves PCP and GAN by 0.12% and 0.06% respectively. Regarding LRR, the improvements over PCP and GAN are 0.16% and 0.10% respectively. Overall, PCPSFM achieves the best result, i.e., 0.12% over PCPSM and 0.08% over LRR. We also plot the ROC curves for these methods in Figure 4.13. In Table 4.6, we list the TAR values under different FAR values. The proposed PCPSFM achieves highest TAR (83.0%) at FAR=1e-3. Arguably, the proposed PCPSFM does improve the accuracy of video-based face verification.

Table 4.5: Verification accuracy (%) of different methods on the YTF dataset.

Methods	Images	Acc (%)
DeepID [116]	0.2M	93.20
VGG Face [86]	2.6M	**97.30**
Deep Face [118]	4M	91.40
FaceNet [102]	200M	95.10
Center Loss [122]	0.7M	94.9
Range Loss [141]	1.5M	93.70
Sphere Loss [75]	0.5M	95.0
Marginal Loss [38]	4M	95.98
ArcFace	3.1M	97.52
GAN+ArcFace	3.1M	97.66
PCP+ArcFace	3.1M	97.60
PCPSM+ArcFace	3.1M	97.72
LRR+ArcFace	3.1M	97.76
PCPSFM+ArcFace	3.1M	**97.84**

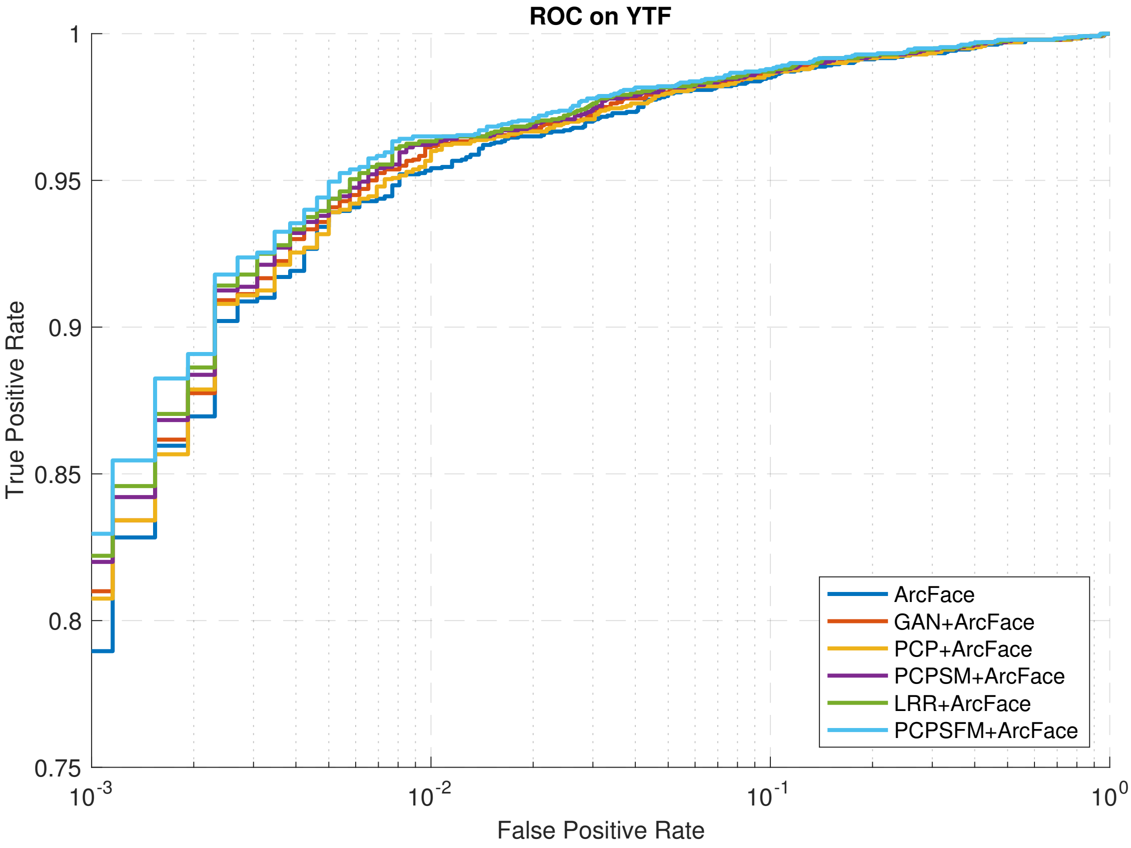


Figure 4.13: ROC curves of the proposed methods on the YouTube Faces database under the "restricted" protocol.

Table 4.6: Verification TAR on the YTF dataset (Higher is better).

Method	FAR=1e-3	FAR=1e-2	FAR=1e-1
ArcFace	0.790	0.953	0.985
GAN+ArcFace	0.810	0.961	0.987
PCP+ArcFace	0.807	0.957	0.986
PCPSM+ArcFace	0.820	0.962	0.987
LRR+ArcFace	0.822	0.963	0.987
PCPSFM+ArcFace	**0.830**	**0.965**	**0.988**

Experiments on PaSC

The PaSC dataset [8] includes $9,376$ still images and $2,802$ videos from 293 people. The images are evenly split with respect to the distance to the camera, alternative sensors, frontal versus not-frontal views and different environments. There are three protocols for face verification: comparing still images to still images, videos to videos, and still images to videos. Since we have conducted image-to-image and video-to-video experiments in previous sections, we only report image-to-video results on PaSC with the public evaluation toolkit.

As the PaSC dataset [8] includes static images and videos of the same people, it is very interesting to explore face verification performance between modalities: static image to dynamic video. Simply put, given only a few images of a person, can we verify this person in the subsequent video that he/she is seen or claimed to be seen? To set up this experiment, we prepare a query set of $1,401$ handheld (or alternatively controlled) videos and a target set comprising of $9,376$ still images from 293 identities. Figure 4.14 presents the ROC curve of each method. In Table 4.7, we report the TAR at different FARs. The proposed PCPSFM significantly improves TAR from 82.4% to 85.7% at FAR=1e-5. In [8], the baseline method only obtains TAR of 42% at FAR=1e-2, whereas our method PCPSFM achieves TAR of 99.0% at FAR=1e-2.

Table 4.7: Verification TAR on the PaSC dataset (Higher is better).

Method	FAR=1e-5	FAR=1e-4	FAR=1e-3
ArcFace	0.824	0.948	0.976
GAN+ArcFace	0.833	0.953	0.979
PCP+ArcFace	0.824	0.950	0.978
PCPSM+ArcFace	0.839	0.953	0.979
LRR+ArcFace	0.849	0.954	0.980
PCPSFM+ArcFace	**0.857**	**0.956**	**0.981**

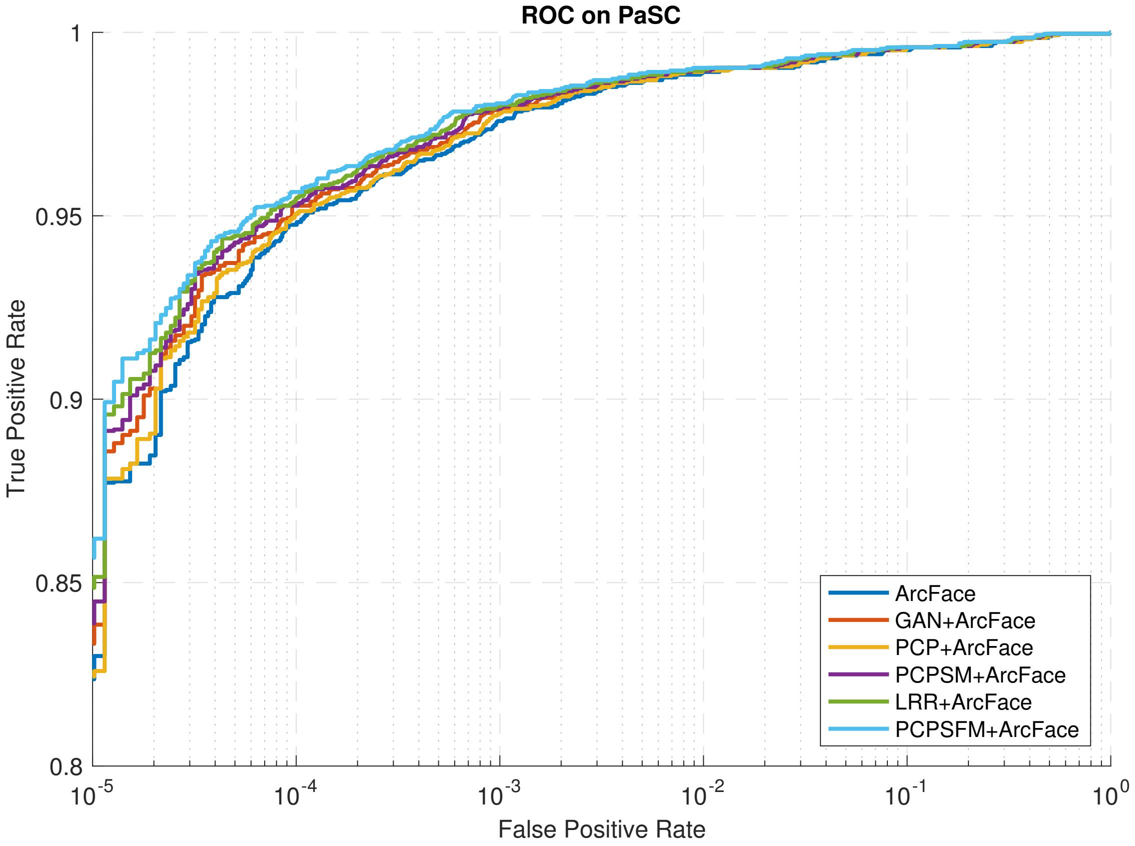


Figure 4.14: ROC curves of the proposed methods on the PaSC dataset.

4.5 Appendix

4.5.1 Generative adversarial networks

For GAN, we employ the image-to-image conditional adversarial network [54] to conduct UV completion. As is shown in Figure 4.15, there are two main components in the image-to-image conditional GAN: a generator module and a discriminator module.

Generator Module Given incomplete UV texture input, the generator G works as an auto-encoder to construct completed instances. We adopt the pixel-wise l_1 norm as the reconstruction loss:

$$L_{gen} = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} \left| I_{ij} - I_{ij}^{*} \right|, \tag{4.30}$$

where I_{ij} is the estimated UV texture and I_{ij}^{*} is the ground truth texture of width W and height H.

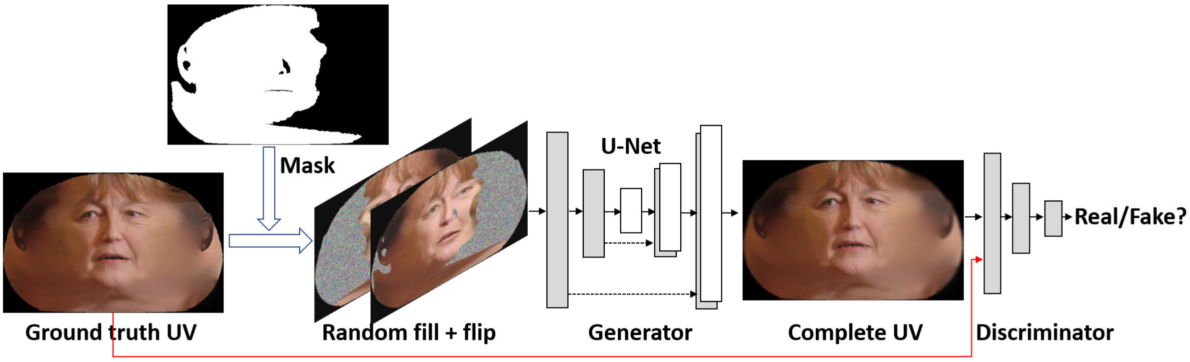


Figure 4.15: Our GAN framework consists of one generator and one discriminator. The generator takes the incomplete UV map as input and outputs the full UV map. The discriminator is learnt to validate the genuineness of the synthesised UV texture. Note that only the generator is used at the testing stage.

To preserve the image information in the original resolution, we follow the encoder-decoder design in [54], where skip connections between mirrored layers in the encoder and decoder stacks are made. We first fill the incomplete UV texture with random noise and then concatenate it with its mirror image as the generator input. Since the face is not exactly symmetric, we have avoided using symmetry loss as in [53]. Also, unlike the original GAN model [48] which is initialised from a noise vector, the hidden representations obtained from our encoder capture more variations as well as relationships between invisible and visible regions, and thus help the decoder fill up the missing regions.

Discriminator Module Although the previous generator module can fill missing pixels with small reconstruction errors, it does not guarantee the output textures to be visually realistic and informative. With only the pixel-wise l_1 reconstruction loss, the UV completion results would be quite blurry and missing important details. To improve the quality of synthetic images and encourage more photo-realistic results, we adopt a discriminator module D to distinguish real and fake UVs. The adversarial loss, which is a reflection of how the generator could maximally fool the discriminator and how well the discriminator could distinguish between real and fake UVs, is defined as

$$\begin{aligned} L_{adv} = \min_{G} \max_{D} \mathbb{E}_{\boldsymbol{x} \sim p_d(\boldsymbol{x}), \boldsymbol{y} \sim p_d(\boldsymbol{y})} \left[\log D(\boldsymbol{x}, \boldsymbol{y}) \right] + \\ \mathbb{E}_{\boldsymbol{x} \sim p_d(\boldsymbol{x}), \boldsymbol{z} \sim p_z(\boldsymbol{z})} \left[\log (1 - D(\boldsymbol{x}, G(\boldsymbol{x}, \boldsymbol{z}))) \right], \end{aligned} \tag{4.31}$$

where $p_z(\boldsymbol{z})$, $p_d(\boldsymbol{x})$ and $p_d(\boldsymbol{y})$ represent the distributions (Gaussian) of the noise variable $\boldsymbol{z}$, the partial UV texture $\boldsymbol{x}$ and the full UV texture $\boldsymbol{y}$ respectively.

Objective Function The final loss function for the proposed UV-GAN is a weighted sum of generator loss and discriminator loss:

$$L = L_{gen} + \lambda L_{adv\_g}. \tag{4.32}$$

where λ is the weight to balance generator loss and discriminator loss. In all our experiments, we empirically set $\lambda = 10^{-2}$ following [36, 64] as best UV completion results can be obtained under this setting.

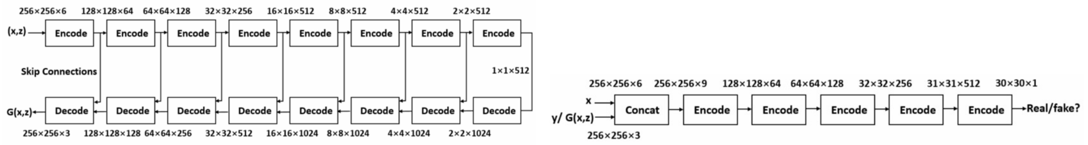
Architecture The network architecture of [54] is adopted here[2]. The encoder unit consists of convolution, batch normalisation and ReLU, while the decoder unit consists of deconvolution, batch normalisation and ReLU. The convolution involves 4×4 spatial filters applied with stride 2. Convolution in the encoder and the discriminator is also downsampled by a factor of 2, while in the decoder it is upsampled by a factor of 2.

As shown in Figure 4.16(a), the generator utilises the U-Net [98] architecture which has skip connections between i^{th} layer in the encoder and the $(n-i)^{th}$ layer in the decoder, where n is the total number of layers. These skip connections concatenate activations from the i^{th} layer to the $(n-i)^{th}$ layer. Note that batch normalisation is not applied to the first Conv64 layer in the encoder. All ReLUs in the encoder are leaky, with slope 0.2, whereas ReLUs in the decoder are not leaky.

For the discriminator, we use the 70×70 PatchGAN as in [54]. In Figure 4.16(b), we depict the architecture of the discriminator. Again, batch normalisation is not applied to the first Conv64 layer. However, all ReLUs are now leaky, with slope 0.2. We have also set the stride of the last two encoder modules to 1.

Training We train our networks from scratch by initialising the weights from a Gaussian distribution with zero mean and 0.02 standard deviation. In order to train our UV completion model by pairwise image data, we make use of both under-controlled and in-the-wild UV datasets. For the under-controlled UV data, we randomly select 180 subjects from the 4DFAB dataset [30]. For the in-the-wild UV data, we employ the pseudo-complete UVs from the UMD video dataset [5] via Poisson blending [91]. We have meticulously chosen videos with large pose variations such that coverage of different poses is adequate. In the end, we have a combined UV dataset of 1,892 identities with 5,638 unique UV maps.

[2] https://github.com/phillipi/pix2pix



(a) Generator architecture

(b) Discriminator architecture

Figure 4.16: The encoder unit consists of convolution, batch normalisation and ReLU, and the decoder unit consists of de-convolution, batch normalisation and ReLU. The input to the generator is the occluded UV map x that is filled with random noise z and concatenated with its flipped image. The input to the discriminator is the original input x and either the ground-truth UV map y or the generated UV map $G(x, z)$.

4.5.2 Deep face feature embedding networks

We use Arcface[3] [37] for 512-D facial feature embedding with resnet-50 and additive angular margin loss. The size of all the convolutional filters is 3×3 with stride 1. And the kernel size of max-pooling is set to 2×2 with stride 2. The network is initialised from the Gaussian distribution and trained on the VGG training set (c3.1 million images) under the supervisory signals of additive angular margin loss. After an initial learning rate of 0.1, we successively contract it by a factor of 10 at the 6^{th}, 14^{th}, 22^{th} and 30^{th} epoch. We train the network in parallel on four GPUs so the overall batch size is 128×4. The input face size of the network is 112×112 pixels.

4.5.3 Dictionary Learning

We compare the performance of different methods for learning subspace features by doing recognition experiments on the recovered images of the CFP dataset. For KSVD, we use the formulation of eq. 30 of the main text, where we set c and t to 13 and used 10 iterations. For Non-homogeneous KSVD, we modify eq. 30 according to [76]. Given a weight matrix $\beta \in \mathbb{R}^{n_1 \times n_2}$, we simultaneously seek a dictionary $\boldsymbol{D} \in \mathbb{R}^{n_1 \times c}$ and a sparse representation $\boldsymbol{B} \in \mathbb{R}^{c \times n_2}$ such that:

$$\min_{\boldsymbol{D},\boldsymbol{B}} \|\beta \otimes (\boldsymbol{M} - \boldsymbol{D}\boldsymbol{B})\|_F^2 \quad \text{s. t.} \quad \gamma_i \leq t \text{ for } i = 1 \ldots n_2, \tag{4.33}$$

where $\otimes$ is element-wise multiplication, c is the number of atoms, γ_i's count the number of non-zero elements in each sparsity code and t is the sparsity constraint factor. Again, we set c and t to 13 and used 10 iterations with a boolean mask for β that denotes missing regions of the image.

The set-based face verification performance for LRR using subspace features generated by KSVD (KSVD+ArcFace), non-homogeneous KSVD (WKSVD+ArcFace) and GAN (LRR+ArcFace) is shown

[3] https://github.com/deepinsight/insightface

in Figure 4.11 and Table 4.2. Although there can be a small gain by introducing non-homogeneity into KSVD, GAN is obviously preferable to both KSVD algorithms in extracting subspace features. Consequently, we adopt the approach of using GAN for the source of subspace features as well as side information, S.

Table 4.8: Verification TAR on the CFP dataset (Higher is better).

Method	FAR=1e-6	**FAR=1e-5**	**FAR=1e-4**	FAR=1e-3	FAR=1e-2	FAR=1e-1
ArcFace	0.901	0.950	0.989	0.996	0.999	0.999
KSVD+ArcFace	0.892	0.950	0.989	0.996	0.999	0.999
WKSVD+ArcFace	0.895	0.954	0.989	0.996	0.999	0.999
LRR+ArcFace	0.911	0.963	0.993	0.997	0.999	0.999

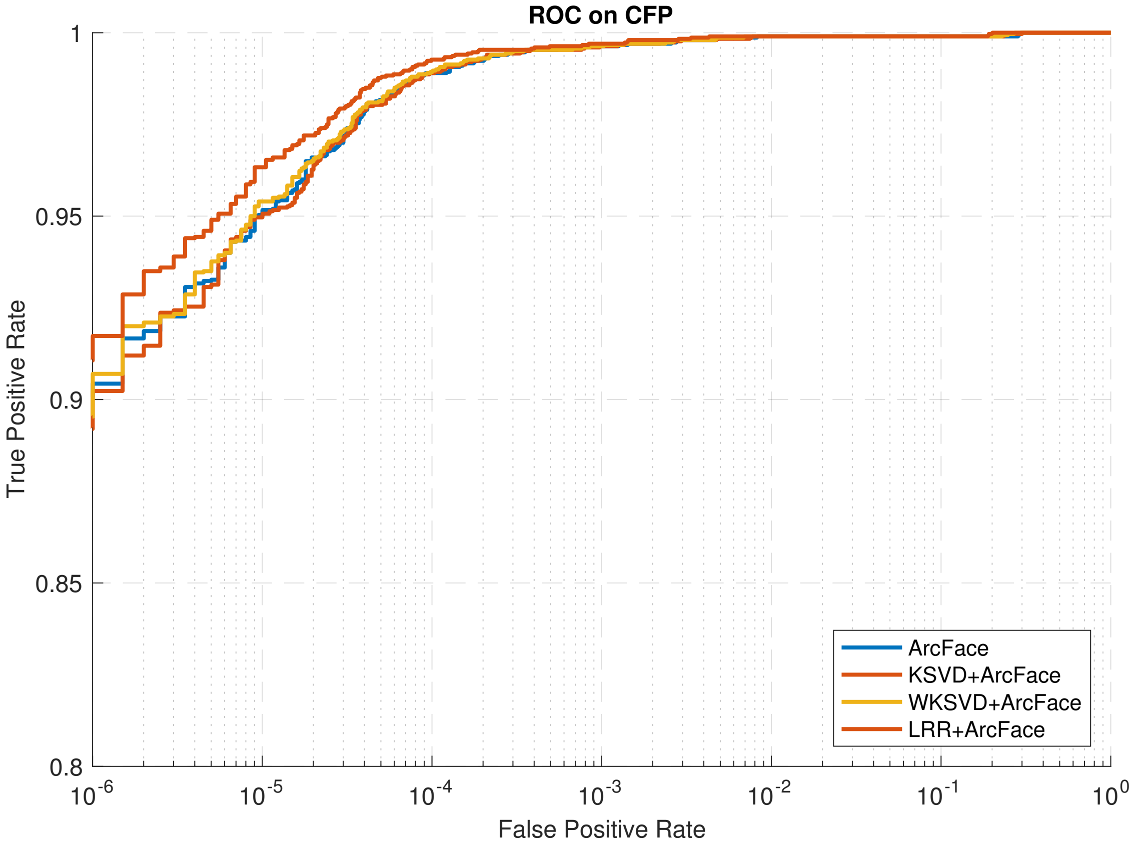


Figure 4.17: ROC curves on the CFP dataset.

4.5.4 Quantitative measures of UV completion

For each video sequence, we plot the probability distributions of PSNR and SSIM values of all the images recovered by algorithms PCP, LRR, PCPSM, PCPSFM in Figure 4.18- 4.22. To test for difference between their means, we assume that scores from any algorithm are normally distributed (this is a good approximation by central limit theorem as the data size, 155, is quite large). We use unbiased estimators for the population variances. The p-values under the null hypothesis, there is no difference between the mean scores between any two algorithms, are summarised in Table 4.9- 4.13.

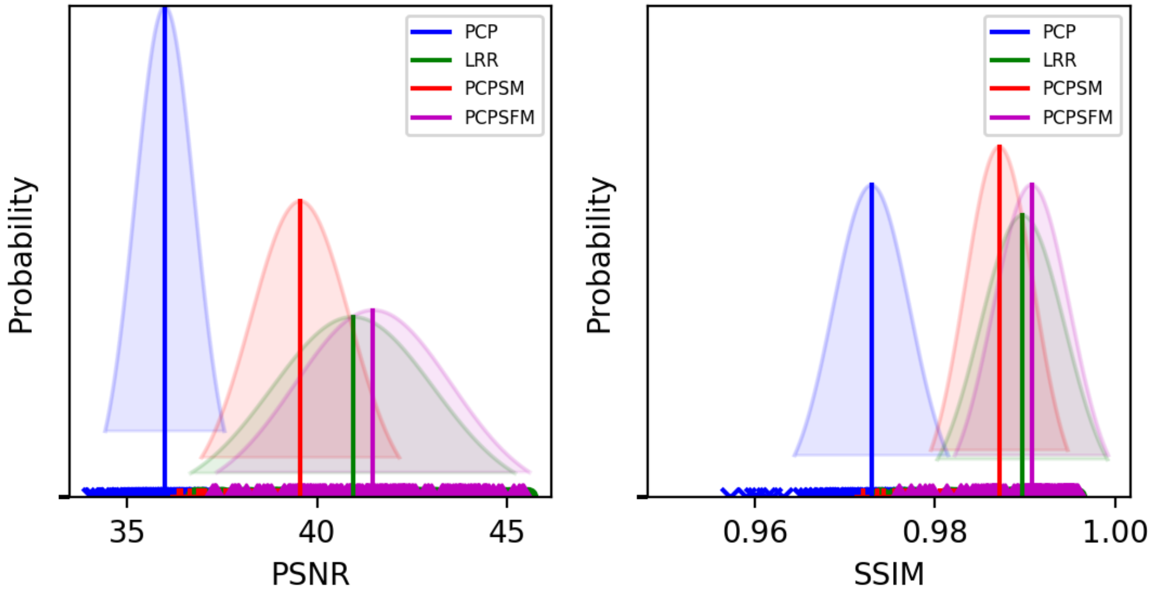


Figure 4.18: Distributions of PSNR and SSIM for all algorithms on Sequence 1 of the 4DFAB dataset.

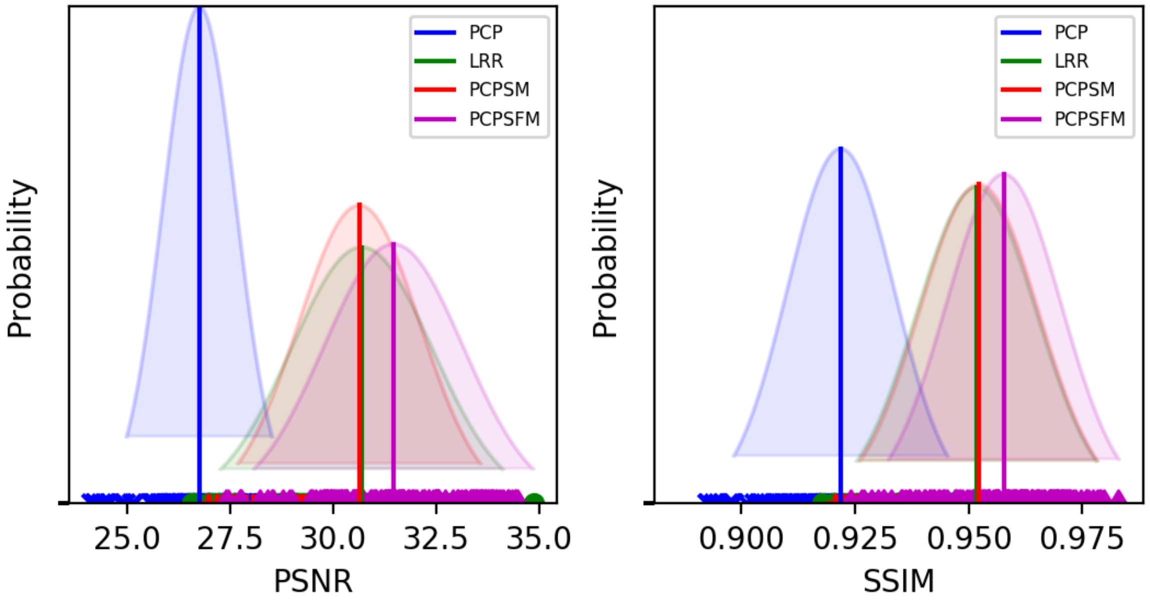


Figure 4.19: Distributions of PSNR and SSIM for all algorithms on Sequence 2 of the 4DFAB dataset.

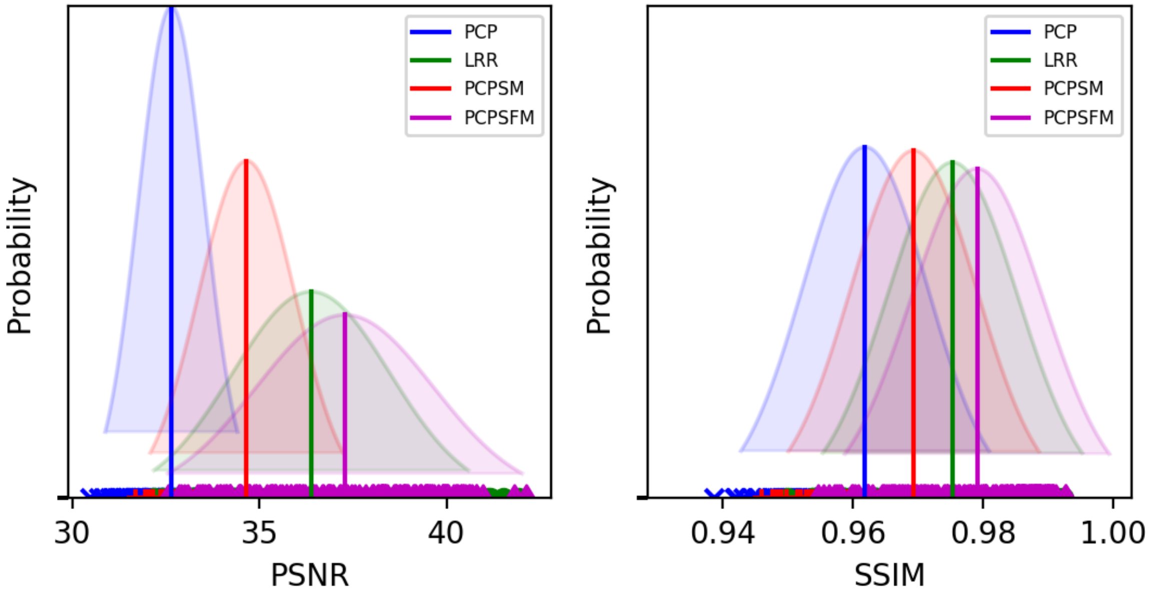


Figure 4.20: Distributions of PSNR and SSIM for all algorithms on Sequence 3 of the 4DFAB dataset.

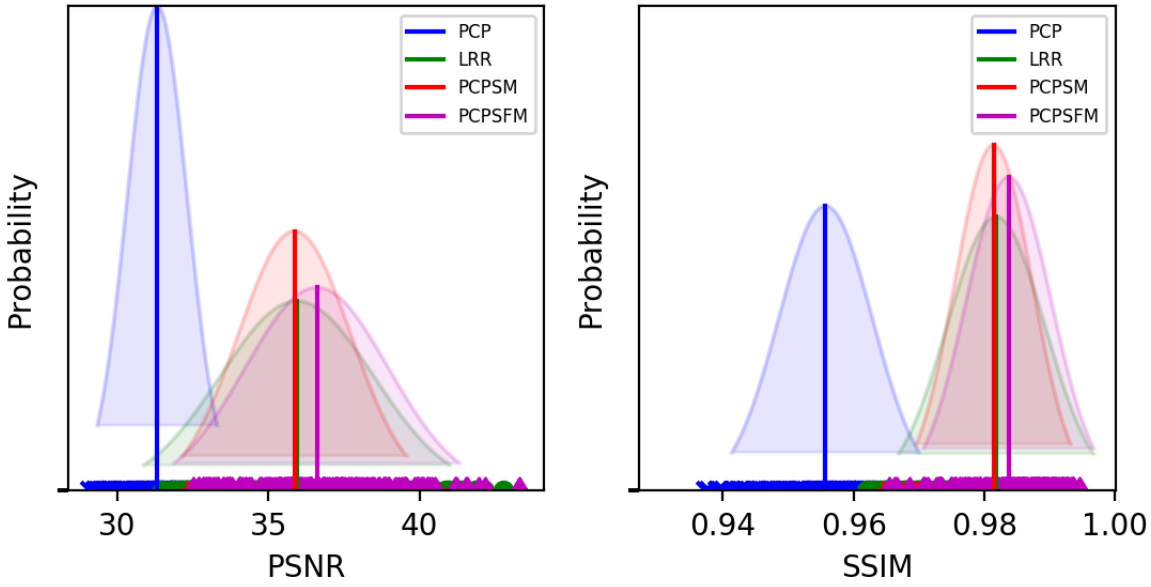


Figure 4.21: Distributions of PSNR and SSIM for all algorithms on Sequence 4 of the 4DFAB dataset.

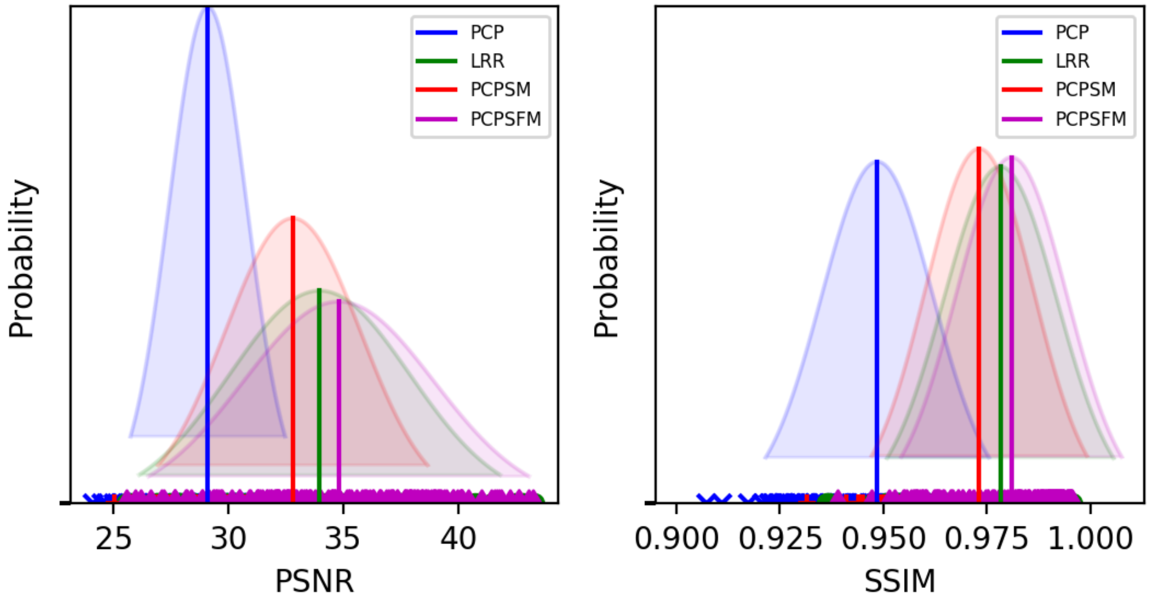


Figure 4.22: Distributions of PSNR and SSIM for all algorithms on Sequence 5 of the 4DFAB dataset.

Table 4.9: P-values of PSNR and SSIM for any pair of algorithms on Sequence 1 of the 4DFAB dataset.

PSNR				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	0.01
PCPSFM	<0.01	<0.01	0.01	-

SSIM				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	0.02
PCPSFM	<0.01	<0.01	0.02	-

Table 4.10: P-values of PSNR and SSIM for any pair of algorithms on Sequence 2 of the 4DFAB dataset.

PSNR				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	0.37	<0.01
LRR	<0.01	0.37	-	<0.01
PCPSFM	<0.01	<0.01	<0.01	-

SSIM				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	0.37	<0.01
LRR	<0.01	0.37	-	<0.01
PCPSFM	<0.01	<0.01	<0.01	-

Table 4.11: P-values of PSNR and SSIM for any pair of algorithms on Sequence 3 of the 4DFAB dataset.

PSNR				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	<0.01
PCPSFM	<0.01	<0.01	<0.01	-

SSIM				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	<0.01
PCPSFM	<0.01	<0.01	<0.01	-

Table 4.12: P-values of PSNR and SSIM for any pair of algorithms on Sequence 4 of the 4DFAB dataset.

PSNR				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	0.37	<0.01
LRR	<0.01	0.37	-	0.01
PCPSFM	<0.01	<0.01	0.01	-

SSIM				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	0.31	<0.01
LRR	<0.01	0.31	-	<0.01
PCPSFM	<0.01	<0.01	<0.01	-

Table 4.13: P-values of PSNR and SSIM for any pair of algorithms on Sequence 5 of the 4DFAB dataset.

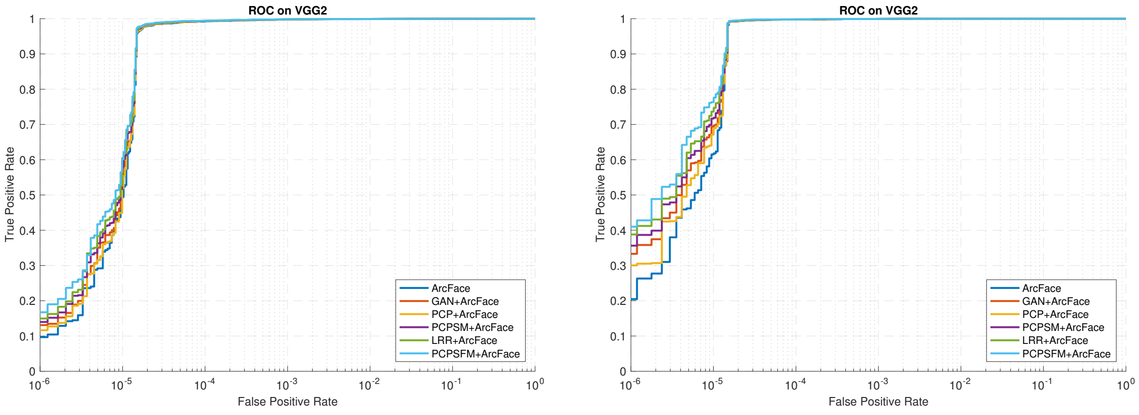
PSNR				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	0.04
PCPSFM	<0.01	<0.01	0.04	-

SSIM				
P-values	PCP	PCPSM	LRR	PCPSFM
PCP	-	<0.01	<0.01	<0.01
PCPSM	<0.01	-	<0.01	<0.01
LRR	<0.01	<0.01	-	0.04
PCPSFM	<0.01	<0.01	0.04	-

4.5.5 Verification experiments on VGG

The VGG2 dataset [23] contains a test set of 500 identities (169,396 images). VGG2 has large variations in pose, age, illumination, ethnicity and profession. To facilitate the evaluation of face matching across different poses, VGG2 provides a face template list for 368 subjects, which contains 2 front templates, 2 three-quarter templates and 2 profile templates. Each template includes 5 images. For set-based face verification on VGG2, we first utilise the released template settings where each subject contains 6 view-specific sets. Thus, there are $5,520$ positive pairs and $2,431$K negative pairs. Since the proposed method targets on information fusion from different views, we also create 5 view-mixed sets for every subject, each of which includes 6 face images from three different views. Thus, there are $3,680$ positive pairs and $1,688.2$K negative pairs.

In Figure 4.23 and Table 4.14 and 4.15, we compare the proposed methods with baseline methods. We observe significant improvement through incorporating feature subspace and side information in the proposed PCPSFM. For the single view set setting, PCPSFM improves TAR by 10.3% at FAR=1e-5 compared to that of ArcFace. For the mixed view set setting, PCPSFM improves TAR by 14.8% at FAR=1e-5 compared to that of ArcFace. The improvement on the mixed view set setting is much higher than on the single view set setting, which indicates that the proposed PCPSFM can effectively integrate information from different views so as to improve pose-invariant face recognition.



(a) ROC for Single View Sets

(b) ROC for Mixed View Sets

Figure 4.23: ROC curves on the VGG2 test set.

4.5.6 Running times

We benchmark running times of various algorithms on the PaSC dataset. We selected 3 videos of different lengths and repeated each experiment 5 times on a machine with Intel Xeon E5-1650 6-core

Table 4.14: Verification TAR on the VGG2 single-view test set (Higher is better).

Method	FAR=1e-6	**FAR=1e-5**	**FAR=1e-4**	FAR=1e-3	FAR=1e-2	FAR=1e-1
ArcFace	0.079	0.462	0.992	0.998	0.999	1.000
GAN+ArcFace	0.122	0.481	0.993	0.998	0.999	1.000
PCP+ArcFace	0.102	0.449	0.993	0.998	0.999	1.000
PCPSM+ArcFace	0.140	0.516	0.993	0.998	0.999	1.000
LRR+ArcFace	0.145	0.526	0.994	0.998	0.999	1.000
PCPSFM+ArcFace	**0.166**	**0.565**	**0.994**	**0.998**	**0.999**	**1.000**

Table 4.15: Verification TAR on the VGG2 mixed-view test set (Higher is better).

Method	FAR=1e-6	**FAR=1e-5**	**FAR=1e-4**	FAR=1e-3	FAR=1e-2	FAR=1e-1
ArcFace	0.205	0.615	0.998	0.999	1.000	1.000
GAN+ArcFace	0.333	0.694	0.998	0.999	1.000	1.000
PCP+ArcFace	0.300	0.671	0.998	0.999	1.000	1.000
PCPSM+ArcFace	0.356	0.717	0.998	0.999	1.000	1.000
LRR+ArcFace	0.388	0.736	0.998	0.999	1.000	1.000
PCPSFM+ArcFace	**0.410**	**0.763**	**0.998**	**0.999**	**1.000**	**1.000**

3.50GHz CPU and 16GB RAM. Table 4.17 summarises the running time results. Table 4.16 also gives the running time (per image) of each processing step in our pipeline.

Table 4.16: Average running times of each processing step in our pipeline.

Processing step	Per image running time
Face detection	15.7ms
Face alignment	12.2ms
3DMM fitting	20s
GAN	25.6ms
PCPSFM	<1.5s
Face rendering [34, 44]	0.6ms
Feature embedding	8.9ms

Table 4.17: Average running times of various algorithms. WKSVD stands for non-homogenous KSVD [76].

	Average running time		
Method	video length = 10	video length = 40	video length = 200
KSVD	1.4s	23.2s	12.1min
WKSVD	54.6s	4.2min	1.7hr
GAN	256.2ms	1.0s	5.1s
PCP	2.4s	18.6s	2.5min
LRR	2.5s	23.5s	3.2min
PCPSM	5.5s	36.5s	5.5min
PCPSFM	3.4s	31.0s	3.9min

CHAPTER 5

Robust Low-rank Tensor Modelling Using Tucker and CP Decomposition

5.1 Introduction

Here we present two novel robust tensor methods based on Tucker and CP decomposition that recover the latent low-rank component from noisy observations by relaxing (1.14), which is NP-hard. In section 5.2, we review relevant literature on matrix and tensor algorithms. In section 5.3, we explain our proposed tensor methods in detail. In section 5.4, we demonstrate the advantages of our models on both synthetic data and a real-world dataset.

5.2 Related work

Much recent research on subspace analysis for the matrix case has direct applicability to tensor data. The costly singular value decomposition step in classical PCP prohibits large-scale analysis. A general approach to mitigate this issue is to look for a factorisation of the low rank component $\boldsymbol{A}$. ORPCA [70] uses a linear combination of the active subspace, $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}$, $\boldsymbol{U}^\intercal\boldsymbol{U} = \boldsymbol{I}$, where bilinear factors $\boldsymbol{U} \in \mathbb{R}^{m\times k}$ and $\boldsymbol{V} \in \mathbb{R}^{k\times n}$ are the principal components and the combination coefficients respectively and k is an upper bound of rank($\boldsymbol{A}$).

5.3 Models

5.3.1 Soft and hard thresholding operators

For fixed $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, the optimal analytical solution for $\min_{\boldsymbol{\mathcal{Y}}} \ \kappa \|\boldsymbol{\mathcal{Y}}\|_1 + \frac{1}{2}\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{Y}}\|_F^2$ is given by the soft thresholding operation $\boldsymbol{\Theta}_\kappa(\boldsymbol{\mathcal{X}})$, where

$$\Theta_\kappa(\boldsymbol{\mathcal{X}})_{\iota_1 \cdots \iota_N} = (\mathcal{X}_{\iota_1 \cdots \iota_N} - \kappa)_+ - (-\mathcal{X}_{\iota_1 \cdots \iota_N} - \kappa)_+. \tag{5.1}$$

And for fixed $\boldsymbol{X} \in \mathbb{R}^{m \times n}$, the optimal analytical solution for $\min_{\boldsymbol{Y}} \ \kappa \|\boldsymbol{X}\|_* + \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2$ is given by the hard thresholding operation $\boldsymbol{\Phi}_\kappa(\boldsymbol{X})$, where

$$\boldsymbol{\Phi}_\kappa(\boldsymbol{X}) = \boldsymbol{U}\boldsymbol{\Theta}_\kappa(\boldsymbol{S})\boldsymbol{V}^\intercal, \tag{5.2}$$

for singular value decomposition $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\intercal$.

5.3.2 Tensor Orthonormal Robust PCA

Generalisation of ORPCA to tensors corresponds to the following factorisation of the low-rank component $\boldsymbol{\mathcal{L}}$:

$$\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{V}} \times_1 \boldsymbol{U}_1 \times \cdots \times_N \boldsymbol{U}_N \equiv \boldsymbol{\mathcal{V}} \times_{i=1}^N \boldsymbol{U}_i, \ \boldsymbol{U}_i^\intercal \boldsymbol{U}_i = \boldsymbol{I}, \tag{5.3}$$

which is exactly the HOSVD [63] of $\boldsymbol{\mathcal{L}}$ and the following relationship holds

$$\|\boldsymbol{L}_{(i)}\|_* = \|\boldsymbol{V}_{(i)}\|_*. \tag{5.4}$$

Based on the above, (1.15) can be re-written as

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{V}},\boldsymbol{\mathcal{S}}} \sum_{i=1}^N \alpha_i \|\boldsymbol{V}_{(i)}\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1, \\ &\text{s.t. } \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{V}} \times_{i=1}^N \boldsymbol{U}_i + \boldsymbol{\mathcal{S}}, \quad \boldsymbol{U}_i^\intercal \boldsymbol{U}_i = \boldsymbol{I}, \quad 1 \le i \le N. \end{aligned} \tag{5.5}$$

To separate variables, we make the substitution $\boldsymbol{V}_{(i)} = \boldsymbol{J}_i$, to arrive at an equivalent problem:

$$\begin{aligned} &\min_{\boldsymbol{J}_i,\boldsymbol{\mathcal{S}}} \sum_{i=1}^N \alpha_i \|\boldsymbol{J}_i\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1, \\ &\text{s.t. } \boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{V}} \times_{i=1}^N \boldsymbol{U}_i + \boldsymbol{\mathcal{S}}, \quad \boldsymbol{U}_i^\intercal \boldsymbol{U}_i = \boldsymbol{I}, \\ &\quad \boldsymbol{V}_{(i)} = \boldsymbol{J}_i, \quad 1 \le i \le N. \end{aligned} \tag{5.6}$$

To apply ADMM, the augmented Lagrangian of (5.6) is constructed first:

$$\begin{aligned}\mathcal{L}(\boldsymbol{J}_i, \boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{S}}, \boldsymbol{U}_i, \boldsymbol{\mathcal{Y}}, \boldsymbol{Z}_i) = &\sum_{i=1}^{N} \alpha_i \|\boldsymbol{J}_i\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \\ &+ \langle \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i - \boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{Y}} \rangle + \sum_{i=1}^{N} \langle \boldsymbol{V}_{(i)} - \boldsymbol{J}_i, \boldsymbol{Z}_i \rangle \\ &+ \frac{\mu}{2} \|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i - \boldsymbol{\mathcal{S}}\|_F^2 + \sum_{i=1}^{N} \frac{\mu}{2} \|\boldsymbol{V}_{(i)} - \boldsymbol{J}_i\|_F^2,\end{aligned} \tag{5.7}$$

where $\boldsymbol{U}_i^\mathsf{T} \boldsymbol{U}_i = \boldsymbol{I}$ has not been incorporated.

$\boldsymbol{J}_i$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{J}_i)$:

$$\begin{aligned}\boldsymbol{J}_i &= \arg\min_{\boldsymbol{J}_i} \alpha_i \mu^{-1} \|\boldsymbol{J}_i\|_* + \frac{1}{2} \|\boldsymbol{J}_i - (\boldsymbol{V}_{(i)} + \frac{1}{\mu} \boldsymbol{Z}_i)\|_F^2 \\ &= \boldsymbol{\Phi}_{\alpha_i \mu^{-1}}(\boldsymbol{V}_{(i)} + \frac{1}{\mu} \boldsymbol{Z}_i)\end{aligned} \tag{5.8}$$

$\boldsymbol{\mathcal{V}}$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{\mathcal{V}})$:

$$\begin{aligned}\boldsymbol{\mathcal{V}} = &\arg\min_{\boldsymbol{\mathcal{V}}} \langle \boldsymbol{\mathcal{V}}, -(\mu(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{S}}) + \boldsymbol{\mathcal{Y}}) \times_{i=1}^{N} \boldsymbol{U}_i^\mathsf{T} \rangle \\ &+ \sum_{i=1}^{N} \langle \boldsymbol{\mathcal{V}} - \boldsymbol{\mathcal{J}}_i, \boldsymbol{\mathcal{Z}}_i \rangle + \frac{\mu}{2} \|\boldsymbol{\mathcal{V}}\|_F^2 + \sum_{i=1}^{N} \frac{\mu}{2} \|\boldsymbol{\mathcal{V}} - \boldsymbol{\mathcal{J}}_i\|_F^2,\end{aligned} \tag{5.9}$$

where we have used the fact that $\boldsymbol{U}_i^\mathsf{T} \boldsymbol{U}_i = \boldsymbol{I}$, the Frobenius norm is invariant under rotations and $\boldsymbol{\mathcal{J}}_i, \boldsymbol{\mathcal{Z}}_i$ are the inverse of mode-i matricisations, $\boldsymbol{J}_i, \boldsymbol{Z}_i$ respectively. To obtain $\boldsymbol{\mathcal{V}}$, setting the gradient of (5.9) to zero gives:

$$\boldsymbol{\mathcal{V}} = \frac{1}{N+1}((\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{S}} + \frac{1}{\mu} \boldsymbol{\mathcal{Y}}) \times_{i=1}^{N} \boldsymbol{U}_i^\mathsf{T} + \sum_{i=1}^{N} (\boldsymbol{\mathcal{J}}_i - \frac{1}{\mu} \boldsymbol{\mathcal{Z}}_i)). \tag{5.10}$$

$\boldsymbol{\mathcal{S}}$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{\mathcal{S}})$:

$$\begin{aligned}\boldsymbol{\mathcal{S}} =& \arg\min_{\boldsymbol{\mathcal{S}}} \lambda \mu^{-1} \|\boldsymbol{\mathcal{S}}\|_1 \\ &+ \frac{1}{2} \|\boldsymbol{\mathcal{S}} - (\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i + \frac{1}{\mu} \boldsymbol{\mathcal{Y}})\|_F^2 \\ =& \boldsymbol{\Theta}_{\lambda \mu^{-1}}(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i + \frac{1}{\mu} \boldsymbol{\mathcal{Y}}).\end{aligned} \tag{5.11}$$

$\boldsymbol{U}_i$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{U}_i)$ subject to $\boldsymbol{U}_i^\mathsf{T} \boldsymbol{U}_i = \boldsymbol{I}$:

$$\begin{aligned}\boldsymbol{U}_i =& \arg\min_{\boldsymbol{U}_i} \frac{1}{2} \|\boldsymbol{X}_{(i)} - \boldsymbol{S}_{(i)} + \frac{1}{\mu} \boldsymbol{Y}_{(i)} - \boldsymbol{U}_i \boldsymbol{B}_i\|_F^2, \\ &\text{where } \boldsymbol{B}_i = (\boldsymbol{\mathcal{V}} \times_{j=1}^{i-1} \boldsymbol{U}_j \times_{j=i+1}^{N} \boldsymbol{U}_j)_{(i)}.\end{aligned} \tag{5.12}$$

Algorithm 7 ADMM solver for TORPCA

```
Input: Observation $\boldsymbol{\mathcal{X}}$, parameter $\lambda > 0$, scaling $\kappa > 1$, weights $\alpha_i$, ranks $k_i$
Initialise: $\boldsymbol{J}_i = \boldsymbol{Z}_i = \mathbf{0}$, $\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{Y}} = \mathbf{0}$, $\boldsymbol{\mathcal{V}} = \mathbf{0}$, $\boldsymbol{U}_i =$ first $k_i$ left singular vectors of $\boldsymbol{X}_{(i)}$, $\mu > 0$
while not converged do
    for $i \in \{1, 2, \cdots, N\}$ do
        $\boldsymbol{J}_i = \boldsymbol{\Phi}_{\alpha_i \mu^{-1}}(\boldsymbol{V}_{(i)} + \frac{1}{\mu}\boldsymbol{Z}_i)$
    end for
    $\boldsymbol{\mathcal{V}} = \frac{1}{N+1}((\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{S}} + \frac{1}{\mu}\boldsymbol{\mathcal{Y}}) \times_{i=1}^{N} \boldsymbol{U}_i^\mathsf{T} + \sum_{i=1}^{N}(\boldsymbol{\mathcal{J}}_i - \frac{1}{\mu}\boldsymbol{\mathcal{Z}}_i))$
    $\boldsymbol{\mathcal{S}} = \boldsymbol{\Theta}_{\lambda\mu^{-1}}(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i + \frac{1}{\mu}\boldsymbol{\mathcal{Y}})$
    for $i \in \{1, 2, \cdots, N\}$ do
        $\boldsymbol{B}_i = (\boldsymbol{\mathcal{V}} \times_{j=1}^{i-1} \boldsymbol{U}_j \times_{j=i+1}^{N} \boldsymbol{U}_j)_{(i)}$
        $\boldsymbol{C}_i \boldsymbol{D}_i \boldsymbol{V}_i^\mathsf{T} = (\boldsymbol{X}_{(i)} - \boldsymbol{S}_{(i)} + \frac{1}{\mu}\boldsymbol{Y}_{(i)})\boldsymbol{B}_i^\mathsf{T}$
        $\boldsymbol{U}_i = \boldsymbol{C}_i \boldsymbol{V}_i^\mathsf{T}$
    end for
    $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{Y}} + \mu(\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{V}} \times_{i=1}^{N} \boldsymbol{U}_i - \boldsymbol{\mathcal{S}})$
    for $i \in \{1, 2, \cdots, N\}$ do
        $\boldsymbol{Z}_i = \boldsymbol{Z}_i + \mu(\boldsymbol{V}_{(i)} - \boldsymbol{J}_i)$
    end for
    $\mu = \mu \times \kappa$
end while
Return: $\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{S}}, \boldsymbol{U}_i$
```

If we have the following SVD

$$(\boldsymbol{X}_{(i)} - \boldsymbol{S}_{(i)} + \frac{1}{\mu}\boldsymbol{Y}_{(i)})\boldsymbol{B}_i^\mathsf{T} = \boldsymbol{C}_i \boldsymbol{D}_i \boldsymbol{V}_i^\mathsf{T}, \tag{5.13}$$

then according to the *Reduced Rank Procrustes Theorem* [144], the solution is given by

$$\boldsymbol{U}_i = \boldsymbol{C}_i \boldsymbol{V}_i^\mathsf{T}. \tag{5.14}$$

The complete algorithm is presented in Algorithm 0.

5.3.3 Tensor robust CP decomposition

Let $\boldsymbol{U}^{(i)} = [\boldsymbol{a}_1^{(i)}, \boldsymbol{a}_2^{(i)}, \cdots, \boldsymbol{a}_r^{(i)}]$, then we can express $\boldsymbol{\mathcal{L}}$ compactly as $\boldsymbol{\mathcal{L}} = \boldsymbol{U}^{(1)} \circ \boldsymbol{U}^{(2)} \circ \cdots \circ \boldsymbol{U}^{(N)}$. In particular, it can be shown that $\text{rank}_i(\boldsymbol{\mathcal{L}}) \leq \text{rank}(\boldsymbol{U}_i)$, for $1 \leq i \leq N$. So, it is beneficial to solve the following objective

$$\min_{\boldsymbol{U}_i, \boldsymbol{\mathcal{S}}} \sum_{i=1}^{N} \alpha_i \|\boldsymbol{U}_i\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1, \quad \boldsymbol{\mathcal{X}} = U_1 \circ \boldsymbol{U}_2 \circ \cdots \circ \boldsymbol{U}_N + \boldsymbol{\mathcal{S}}. \tag{5.15}$$

Again, we make the substitution $\boldsymbol{U}_i = \boldsymbol{J}_i$ before performing ADMM, which leads to the following problem

$$\begin{aligned} &\min_{\boldsymbol{J}_i,\boldsymbol{\mathcal{S}}} \sum_{i=1}^{N} \alpha_i \|\boldsymbol{J}_i\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \\ &\text{s.t. } \boldsymbol{\mathcal{X}} = \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N + \boldsymbol{\mathcal{S}}, \ \ \boldsymbol{U}_i = \boldsymbol{J}_i, \ \ 1 \le i \le N. \end{aligned} \tag{5.16}$$

The corresponding augmented Lagrangian is

$$\begin{aligned} \mathcal{L}(\boldsymbol{J}_i, \boldsymbol{U}_i, \boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{Y}}, \boldsymbol{Z}_i) &= \sum_{i=1}^{N} \alpha_i \|\boldsymbol{J}_i\|_* + \lambda \|\boldsymbol{\mathcal{S}}\|_1 \\ &+ \langle \boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N - \boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{Y}} \rangle + \sum_{i=1}^{N} \langle \boldsymbol{U}_i - \boldsymbol{J}_i, \boldsymbol{Z}_i \rangle \\ &+ \frac{\mu}{2} \|\boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N - \boldsymbol{\mathcal{S}}\|_F^2 + \sum_{i=1}^{N} \frac{\mu}{2} \|\boldsymbol{U}_i - \boldsymbol{J}_i\|_F^2. \end{aligned} \tag{5.17}$$

$\boldsymbol{J}_i$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{J}_i)$:

$$\begin{aligned} \boldsymbol{J}_i &= \arg\min_{\boldsymbol{J}_i} \alpha_i \mu^{-1} \|\boldsymbol{J}_i\|_* + \frac{1}{2} \|\boldsymbol{J}_i - (\boldsymbol{U}_i + \frac{1}{\mu} \boldsymbol{Z}_i)\|_F^2 \\ &= \boldsymbol{\Phi}_{\alpha_i \mu^{-1}} (\boldsymbol{U}_i + \frac{1}{\mu} \boldsymbol{Z}_i). \end{aligned} \tag{5.18}$$

$\boldsymbol{U}_i$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{U}_i)$:

$$\begin{aligned} \boldsymbol{U}_i = \arg\min_{\boldsymbol{U}_i} &\langle \boldsymbol{X}_{(i)} - \boldsymbol{U}_i \tilde{\boldsymbol{U}}_i - \boldsymbol{S}_{(i)}, \boldsymbol{Y}_{(i)} \rangle + \langle \boldsymbol{U}_i - \boldsymbol{J}_i, \boldsymbol{Z}_i \rangle \\ &+ \frac{\mu}{2} \|\boldsymbol{X}_{(i)} - \boldsymbol{U}_i \tilde{\boldsymbol{U}}_i - \boldsymbol{S}_{(i)}\|_F^2 + \frac{\mu}{2} \|\boldsymbol{U}_i - \boldsymbol{J}_i\|_F^2, \\ &\text{where } \tilde{\boldsymbol{U}}_i = (\boldsymbol{U}_N \odot \cdots \odot \boldsymbol{U}_{i+1} \odot \boldsymbol{U}_{i-1} \odot \cdots \odot \boldsymbol{U}_1)^\intercal \end{aligned} \tag{5.19}$$

Setting the derivative of (5.19) to zero gives:

$$\begin{aligned} \boldsymbol{U}_i = &((\boldsymbol{X}_{(i)} - \boldsymbol{S}_{(i)} + \frac{1}{\mu} \boldsymbol{Y}_{(i)}) \tilde{\boldsymbol{U}}_i^\intercal \\ &+ \boldsymbol{J}_i - \frac{1}{\mu} \boldsymbol{Z}_i)(\tilde{\boldsymbol{U}}_i \tilde{\boldsymbol{U}}_i^\intercal + \boldsymbol{I})^{-1}. \end{aligned} \tag{5.20}$$

$\boldsymbol{\mathcal{S}}$ is updated by the minimiser of $\mathcal{L}(\boldsymbol{\mathcal{S}})$:

$$\begin{aligned} \boldsymbol{\mathcal{S}} &= \arg\min_{\boldsymbol{\mathcal{S}}} \lambda \mu^{-1} \|\boldsymbol{\mathcal{S}}\|_1 \\ &+ \frac{1}{2} \|\boldsymbol{\mathcal{S}} - (\boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N) + \frac{1}{\mu} \boldsymbol{\mathcal{Y}}\|_F^2 \\ &= \boldsymbol{\Theta}_{\lambda \mu^{-1}} (\boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N + \mu^{-1} \boldsymbol{\mathcal{Y}}). \end{aligned} \tag{5.21}$$

The complete algorithm is described in Algorithm 0.

Algorithm 8 ADMM solver for TRCPD

```
Input: Observation $\boldsymbol{\mathcal{X}}$, parameter $\lambda > 0$, scaling $\kappa > 1$, weights $\alpha_i$, rank $k$
Initialise: $\boldsymbol{J}_i = \boldsymbol{U}_i =$ rand, $\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{Y}} = \mathbf{0}$, $\boldsymbol{Z}_i = \mathbf{0}$, $\mu > 0$
while not converged do
    for $i \in \{1, 2, \cdots, N\}$ do
        $\boldsymbol{J}_i = \boldsymbol{\Phi}_{\alpha_i \mu^{-1}}(\boldsymbol{U}_i + \frac{1}{\mu}\boldsymbol{Z}_i)$
        $\tilde{\boldsymbol{U}}_i = (\boldsymbol{U}_N \odot \cdots \odot \boldsymbol{U}_{i+1} \odot \boldsymbol{U}_{i-1} \odot \cdots \odot \boldsymbol{U}_1)^\intercal$
        $\boldsymbol{U}_i = ((\boldsymbol{X}_{(i)} - \boldsymbol{S}_{(i)} + \frac{1}{\mu}\mathbf{Y}_{(i)})\tilde{\boldsymbol{U}}_i^\intercal + \boldsymbol{J}_i - \frac{1}{\mu}\boldsymbol{Z}_i)(\tilde{\boldsymbol{U}}_i\tilde{\boldsymbol{U}}_i^\intercal + \boldsymbol{I})^{-1}$
    end for
    $\boldsymbol{\mathcal{S}} = \boldsymbol{\Theta}_{\lambda\mu^{-1}}(\boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N + \frac{1}{\mu}\boldsymbol{\mathcal{Y}})$
    $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{Y}} + \mu(\boldsymbol{\mathcal{X}} - \boldsymbol{U}_1 \circ \cdots \circ \boldsymbol{U}_N - \boldsymbol{\mathcal{S}})$
    for $i \in \{1, 2, \cdots, N\}$ do
        $\boldsymbol{Z}_i = \boldsymbol{Z}_i + \mu(\boldsymbol{U}_i - \boldsymbol{J}_i)$
    end for
    $\mu = \mu \times \kappa$
end while
Return: $\boldsymbol{U}_i, \boldsymbol{\mathcal{S}}$
```

5.3.4 Complexity and convergence

For ease of exposition, we assume that $d_1, \cdots, d_N = \zeta$. For TORPCA, the most expansive calculation in each iteration is the i-mode product which has a time complexity of $O(Nr\zeta^N)$. For TRCPD, the dominant term is the chain of matrix outer products which costs $O(Nr\zeta^N)$. Note that both methods have lower complexity than RSTD whose complexity is $O(N\zeta^{N+1})$ due to SVD if $r < \zeta$.

Although both of our proposed tensor methods are non-convex, we have empirically found that the warm initialisation of using the first k_i left singular vectors of $\boldsymbol{X}_{(i)}$ for $\boldsymbol{U}_i$ works well for TORPCA and uniform initialisation of $\boldsymbol{U}_i, 1 \leq i \leq k$ from $[0, 1]$ suffices for TRCPD (see Section 5.4).

5.4 Experiments

5.4.1 Implementation details

For stopping criteria, we use one of the KKT opimality conditions, $\frac{\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{L}} - \boldsymbol{\mathcal{S}}\|_F}{\|\boldsymbol{\mathcal{X}}\|_F} < \delta$ and we have set $\delta = 10^{-7}$. The initial value of μ is set to 10^{-3}, which is geometrically increased by a factor of $\kappa = 1.2$ up to 10^9. The weights α_i are assumed equal.

5.4.2 Simulation

We first evaluate the performance of all algorithms on synthetic data. A low-rank tensor $\boldsymbol{\mathcal{L}} \in \mathbb{R}^{100\times100\times100}$ is generated via $\boldsymbol{\mathcal{L}} = \boldsymbol{U}_1 \circ \boldsymbol{U}_2 \circ \boldsymbol{U}_3$, where elements of $\boldsymbol{U}_1, \boldsymbol{U}_2, \boldsymbol{U}_3 \in \mathbb{R}^{100\times8}$ are independently sampled from the standard Gaussian distribution. The variance of $\boldsymbol{\mathcal{L}}$ is normalised to 1 afterwards. A sparse tensor $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{100\times100\times100}$ is constructed by uniform sampling from $[-10, 10]$. Then only 20% of the elements are kept, with others set to zero.

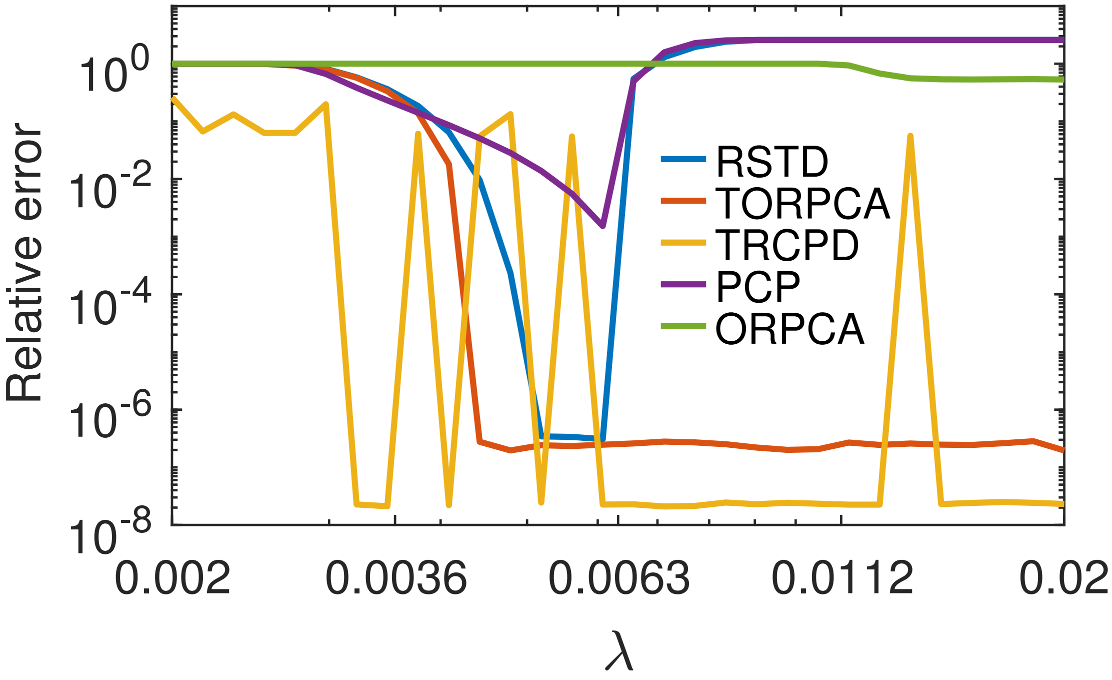


Figure 5.1: Relative error from all algorithms for a range of λ.

Each tensor algorithm takes $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{S}}$ as input, whereas matrix algorithms take mode-1 matricisation of $\boldsymbol{\mathcal{X}}$ as input. Since $\text{rank}(\boldsymbol{\mathcal{L}}) \leq 8$, the rank k in TRCPD is set to 8 and the ranks k_i in TORPCA are all set to 8 because $\text{rank}_i(\boldsymbol{\mathcal{L}}) \leq 8$. The relative error $\frac{\|\boldsymbol{\mathcal{L}}-\tilde{\boldsymbol{\mathcal{L}}}\|_F}{\|\boldsymbol{L}\|_F}$ averaged over 5 trials against λ is plotted for the optimal $\tilde{\boldsymbol{\mathcal{L}}}$ in each algorithm in Fig 5.1. The total execution time for each algorithm versus λ is shown in Fig 5.2.

It is clear that tensor methods are superior to matrix-based methods. Particularly, TRCPD performs the best and TORPCA is also better than RSTD. Both TRCPD and TORPCA are stable in terms of λ whereas RSTD depends on tuning heavily. The execution time confirms our complexity analysis. Both of TORCPA and TRCPD are significantly faster than RSTD.

5.4.3 Facial image denoising

It is well understood that a convex Lambertian surface, viz. faces, under distant and isotropic lighting has a low-rank underlying model. In light of this, we consider images of a fixed pose under different

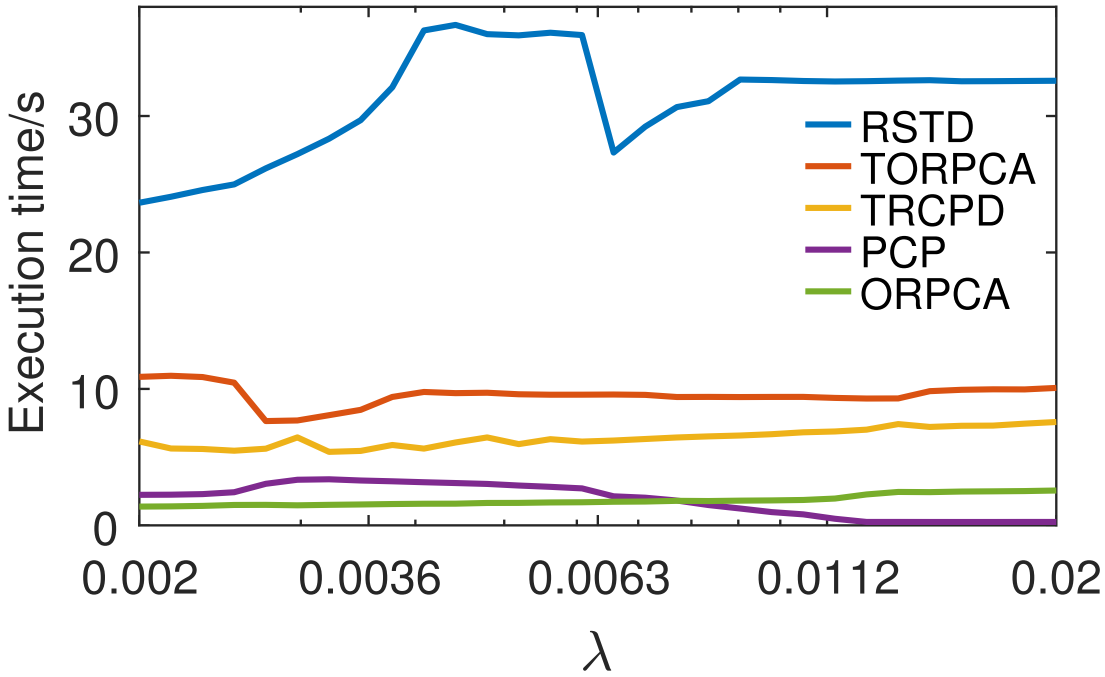


Figure 5.2: Running time of all algorithms as λ varies.

illuminations from the extended Yale B database for benchmarking. All 64 images for one person were studied. For matrix-based methods, 32556×64 observation matrices were formed by vectorising each 168×192 image. All images are also re-scaled such that every pixel lies in $[0, 1]$.

• **Salt & Pepper Noise** Salt & pepper noise is observed in real images, commonly caused by data transmission errors. To apply salt & pepper noise, we randomly set pixels to black (0) or white (1) with equal probability. This is close to the Laplacian noise hypothesis, where noise is heavy, non-Gaussian and potentially wide-ranging. We test an extreme case, where 60% of all the pixels are affected.

• **Partial Occlusion** Partial occlusion is ubiquitous in visual information, which can usually be completed during human visual perception [103]. For the partial occlusion noise, we generate randomly sized patches at random locaions. The maximum dimension is 160 pixels and the occlusion is full of Salt & Pepper noise.

The successful application of various algorithms requires careful tuning of the algorithmic parameters. These include the penalty parameter λ, an estimate of $k = \text{rank}(\boldsymbol{\mathcal{L}})$ and $k_i = \text{rank}_i(\boldsymbol{\mathcal{L}}) = \lceil d_i \times \alpha \rceil$. The ranges of interest and the optimal choices are summarised in Table 5.1.

Reconstruction from salt & pepper noise is illustrated in the first row of Fig 5.3, where the first

Table 5.1: Optimal parameter choices for all algorithms used in different experiments.

Senario	Algorithm	$\lambda \in [10^{-4}, 10^{-1}]$	$k \in \{10, 20, \cdots, 200\}$	$\alpha \in \{0.1, 0.2, \cdots, 0.9\}$
Salt & Pepper	RSTD	0.0092	—	—
	TORPCA	0.2000	—	0.2
	TRCPD	0.0134	160	—
	ORPCA	0.0621	20	—
occlusion	RSTD	0.0076	—	—
	TORPCA	0.0190	—	0.2
	TRCPD	0.0017	50	—
	ORPCA	0.0300	40	—

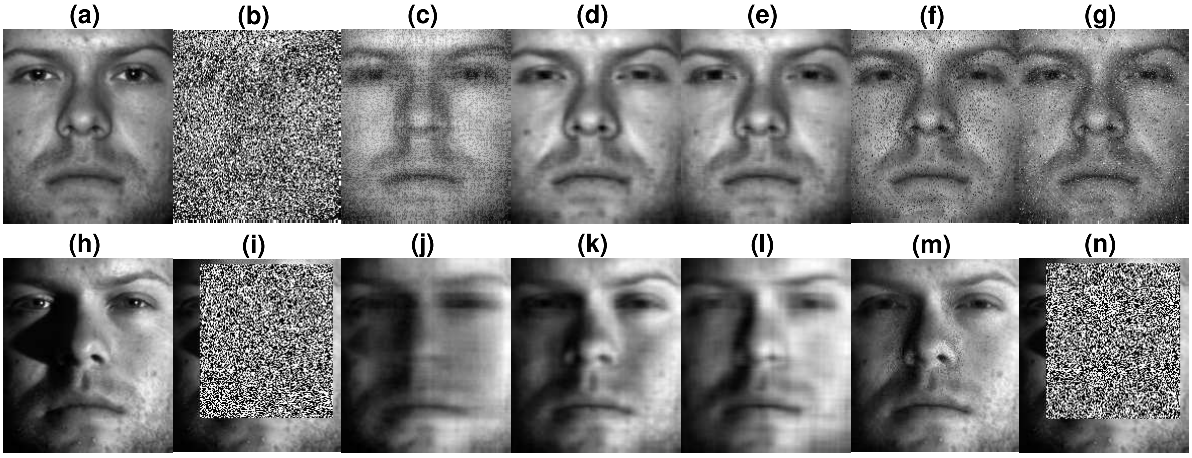


Figure 5.3: Image Denoising Experiments: (**a**) & (**h**) are original images from the sequence. Salt & pepper is introduced as shown in (**b**) and occlusion is demonstrated in (**i**). (**c**) & (**j**) present recovery results for RSTD. (**d**) & (**k**) for TORPCA. (**e**) & (**l**) for TRCPD. (**f**) & (**m**) for PCP. And (**g**) & (**n**) for ORCPA.

image in the sequence is shown. RSTD and matrix-based methods fail to remove the introduced noise, whereas TORPCA and TRCPD are extremely promising such that no trail of noise can be seen. Recovery from partial occlusion is displayed in the second row of Fig 5.3. ORPCA has little effect. The region where noise was introduced is severely distorted in the recovered image of RSTD. Both TORPCA and TRCPD mananged to denoise the occlusion though they have an additional smoothing effect. PCP achieves the highest quality of recovery but there is still unremoved noise left in the image. This may be attributed to the fact that the nature of the occlusion is inherently in a matrix form.

CHAPTER 6

Conclusion

6.1 Thesis summary

First, this dissertation revisits RPCA with features in the context of non-convex optimisation. Exact recovery can be guaranteed if the error sparsity is known to some degree and one of three incoherence assumptions for features and the data matrix is met. Extensive experiments on simulation data indicate that our algorithm can recover matrices with higher ranks and corrupted with more errors than previous state-of-the-art algorithms. Moreover, when the size of the observation matrix increases, our algorithm scales favorably. We further observe that when our algorithm is applied to real-world data, i.e. MNIST and Yale B, significant improvement can be achieved over the compared algorithms. Second, this work, for the first time, incorporates side information which has the same dimension as the observation matrix into the RPCA framework leveraging the format of trace norms. The proposed framework can also assimilate subspace features in the convex optimisation altogether, thus generalising previous approaches. Experiments demonstrate the greater applicability of our algorithm which not only surpasses RPCA in usual applications but also remains potent when RPCA is ineffective. Third, we extend the previous work of robust principal component analysis with features and side information to the case of missing values. Furthermore, for the application domain of UV completion, we propose the use of generative adversarial networks to extract side information and subspaces, which, to the best of our knowledge, is the first time RPCA and GAN have been combined. We also provide a proof of the convergence of the multi-block ADMM optimiser for our convex program. The effectiveness of side information can be seen through experiments on both simulated and real-world datasets. Face recognition benchmarks further justify our fusion approach on in-the-wild data. Last but not the least, we move on to propose two robust tensors models, TORPCA and TRCPD, to deal with low-rank recovery. These models not only outpace their matrix counterparts but also several recent tensor attempts. We firmly believe that our work has further reach in signal processing beyond experiments conducted in this dissertation.

6.2 Limitations and Future work

The derived bound for the non-convex RPCA with features is not optimal. Further efforts are needed to improve the convergence bound for the the proposed algorithm. The initialisation strategy in our non-convex program can still be considered primitive and crude. More efforts should be exerted to pursue better initialisation in order to improve the recovery bound. Only clean features are considered in the RPCA framework in this thesis. However, clean features are not trivial to find the in the real world and investigation to exploit noisy features is called for. We have only considered side information which is an approximation of the low-rank component. Still rigorous characterisation of the side information is demanded to facilitate the full understanding of the effect of side information. We have constrained ourselves to the Euclidean metric. More work is required to generalise the RPCA framework into other manifolds. Additionally, there remains an opportunity to consider graphical knowledge for side information. There are many schemes for tensor decomposition: the tensor-train format, hierarchical tensor decomposition, tensor average rank to name a few. Future works on RPCA could explore extensions to these frameworks. Our analysis on side information is confined within the matrix domain using convex relaxations. Generalising to the tensor domain using non-convex approaches are promising directions for future research. RPCA is essentially an unsupervised learning approach. Labels may also help recovery and form an interesting line of future research by incorporating them into the optimisation objective. Moreover, for complex tasks, it is also beneficial to embed the task-specific metric into the RPCA objective. An example is to consider features, side information, tensor generalisations for kernel RPCA. Given the recent progress of deep learning approaches, it is advantageous to combine the techniques of deep learning and RPCA to form an enhanced algorithm. We limit our application study only to facial recognition from videos, but new application domains, such as pose estimation and gender estimation [94] are also accessible with our proposed approaches after adaptation. Novel application domains where low-rank modelling is appropriate and RPCA is applicable should also be explored.

Bibliography

[1] Abubakar Abid, Martin J. Zhang, Vivek K. Bagaria, and James Zou. Exploring patterns enriched in a dataset with contrastive principal component analysis. *Nature Communications*, 9, 12 2018. 15

[2] Michal Aharon, Michael Elad, and Alfred Bruckstein. rmk-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *TSP*, 54(11):4311–4322, 2006. 126

[3] S. An, W. Liu, and S. Venkatesh. Exploiting side information in locality preserving projection. *CVPR*, 2008. 79

[4] A. Aravkin, S. Becker, V. Cevher, and P. Olsen. A variational approach to stable principal component pursuit. *UAI*, pages 32–41, 2014. 30, 113

[5] Ankan Bansal, Carlos Castillo, Rajeev Ranjan, and Rama Chellappa. The do's and don'ts for cnn-based face verification. In *arXiv:1705.07426*, 2017. 143

[6] B. Bao, G. Liu, C. Xu, and S. Yan. Inductive robust principal component analysis. *IEEE Transactions on Image Processing*, 21(8):3794 – 3800, 2012. 30, 113

[7] R. Basri and D. W. Jacobs. Lambertian reflectance and linear subspaces. *TPAMI*, 25:218–233, February 2003. 40, 125

[8] J Ross Beveridge, P Jonathon Phillips, David S Bolme, Bruce A Draper, Geof H Givens, Yui Man Lui, Mohammad Nayeem Teli, Hao Zhang, W Todd Scruggs, and Kevin W Bowyer. The challenge of face recognition from digital point-and-shoot cameras. In *BTAS*, pages 1–8, 2013. 133, 140

[9] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999. 113

[10] Volker Blanz and Thomas Vetter. Face recognition based on fitting a 3d morphable model. *TPAMI*, 25(9):1063–1074, 2003. 126

[11] Navaneeth Bodla, Jingxiao Zheng, Hongyu Xu, Jun-Cheng Chen, Carlos Castillo, and Rama Chellappa. Deep heterogeneous feature fusion for template-based face recognition. In *WACV*, pages 586–595, 2017. 136

[12] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou. 3d face morphable models in-the-wild. In *CVPR*, 2017. 113

[13] J. Booth and S. Zafeiriou. Optimal uv spaces for facial morphable model construction. In *ICIP*, 2014. 113

[14] James Booth, Epameinondas Antonakos, Stylianos Ploumpis, George Trigeorgis, Yannis Panagakis, and Stefanos Zafeiriou. 3d face morphable models in-the-wild. In *CVPR*, 2017. 13, 127, 133, 134

[15] James Booth, Anastasios Roussos, Evangelos Ververas, Epameinondas Antonakos, Stylianos Poumpis, Yannis Panagakis, and Stefanos P Zafeiriou. 3d reconstruction of" in-the-wild" faces in images and videos. *IEEE T-PAMI*, 2018. 127

[16] James Booth, Anastasios Roussos, Stefanos Zafeiriou, Allan Ponniah, and David Dunaway. A 3d morphable model learnt from 10,000 faces. In *CVPR*, 2016. 126, 127

[17] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. 26, 28, 83, 120

[18] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2d and 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, 2017. 127

[19] R. Cabral, F. De la Torre, J.P. Costeira, and A. Bernardino. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *IEEE International Conference on Computer Vision*, 2013. 30, 113

[20] E. Candès. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008. 30, 114

[21] E.J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), 2011. 27, 29, 30, 39, 40, 81, 83, 113, 118, 120

[22] Q. Cao, L. Shen, W. Xie, O. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose andage. In *arXiv:1710.08092*, 2017. 113

[23] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *FG*, 2018. 133, 136, 138, 151

[24] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM J. Optim.*, 21:572–596, June 2011. 29, 30, 113

[25] V. Chandrasekarana and M. I. Jordan. Computational and statistical tradeoffs via convex relaxation. *PNAS*, 110:E1181–E1190, March 2013. 29

[26] Feng-Ju Chang, Anh Tuan Tran, Tal Hassner, Iacopo Masi, Ram Nevatia, and Gerard Medioni. Faceposenet: Making a case for landmark-free face alignment. In *ICCVW*, 2017. 136

[27] Ching-Hui Chen, Vishal M. Patel, and Rama Chellappa. Learning from ambiguously labeled face images. *TPAMI*, 40(7):1653–1667, 2018. 113

[28] Jie Chen, Vishal Patel, Li Liu, Vili Kellokumpu, Guoying Zhao, Matti Pietikäinen, and Rama Chellappa. Robust local features for remote face recognition. *Image and Vision Computing*, 2017. 132

[29] Yudong Chen, Ali Jalali, Sujay Sanghavi, and Constantine Caramanis. Low-rank matrix recovery from errors and erasures. *TIT*, 59(7):4324–4337, 2013. 28

[30] Shiyang Cheng, Irene Kotsia, Maja Pantic, and Stefanos Zafeiriou. 4dfab: A large scale 4d facial expression database for biometric applications. In *CVPR*, 2018. 127, 143

[31] K. Chiang, C. Hseih, and I. Dhillon. Matrix completion with noisy side information. In *NIPS*, 2015. 29, 30, 114

[32] K. Chiang, C. Hsieh, and I. Dhillon. Robust principal component analysis with side information. In *ICML*, 2016. 29, 52

[33] K. Chiang, C. Hsieh, and I. Dhillon. Robust principal component analysis with side information. *ICML*, 2016. 30, 83, 115

[34] Forrester Cole, David Belanger, Dilip Krishnan, Aaron Sarna, Inbar Mosseri, and William T Freeman. Synthesizing normalized faces from facial identity features. In *CVPR*, pages 3386–3395, 2017. 152

[35] I. Cox, J. Kilian, F. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, 1997. 79

[36] Jiankang Deng, Shiyang Cheng, Niannan Xue, Yuxiang Zhou, and Stefanos Zafeiriou. Uv-gan: Adversarial facial uv map completion for pose-invariant face recognition. In *CVPR*, 2018. 13, 127, 133, 134, 143

[37] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *arXiv:1801.07698*, 2018. 133, 136, 138, 144

[38] Jiankang Deng, Yuxiang Zhou, and Stefanos Zafeiriou. Marginal loss for deep face recognition. In *CVPRW*, 2017. 139

[39] C. Ding, J. Choi, D. Tao, and L. Davis. Multi-directional multi-level dual-cross patterns for robust face recognition. *TPAMI*, 2015. 133

[40] C. Ding and D. Tao. A comprehensive survey on pose-invariant face recognition. *TIST*, 7(3):1–42, 2016. 133

[41] Changxing Ding, Jonghyun Choi, Dacheng Tao, and Larry S Davis. Multi-directional multi-level dual-cross patterns for robust face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 38(3):518–531, 2016. 133

[42] Ming Du, Aswin C Sankaranarayanan, and Rama Chellappa. Robust face recognition from multi-view videos. *IEEE transactions on image processing*, 23(3):1105–1117, 2014. 133

[43] R. Ge, J. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *NIPS*, 2016. 43

[44] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T Freeman. Unsupervised training for 3d morphable model regression. In *CVPR*, 2018. 152

[45] C. Georgakis, Y. Panagakis, and M. Pantic. Discriminant incoherent component analysis. *TIP*, 25(5):2021–2034, 2016. 91

[46] D. Goldfarb and Z. Qin. Robust low-rank tensor recovery: Models and algorithms. *SIAM Journal on Matrix Analysis and Applications*, 35(1), 2014. 39

[47] P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *ICML*, 2013. 43

[48] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014. 142

[49] J. Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4):644–654, 1990. 39

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 133

[51] Michael Hintermüller and Tao Wu. Robust principal component pursuit via inexact alternating minimization on matrix manifolds. *Journal of Mathematical Imaging and Vision*, 51(3):361–377, 2015. 122

[52] B. Huang, C. Mu, J. Wright, and D. Goldfarb. Provable models for robust low-rank tensor completion. *Pacific Journal of Optimisation*, 11(2):339–364, 2015. 39

[53] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *ICCV*, 2017. 142

[54] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *arXiv:1611.07004*, 2016. 129, 141, 142, 143

[55] J. Jiao, T. Courtade, K. Venkat, and T. Weissman. Justification of logarithmic loss via the benefit of side information. *IEEE Transactions on Information Theory*, 61(10):5357–5365, 2015. 30, 113

[56] M. Kan, S. Shan, H. Chang, and X. Chen. Stacked progressive auto-encoders (spae) for face recognition. In *CVPR*, 2014. 133

[57] W. Kang, D. Cao, and N. Liu. Deception with side information in biometric authentication systems. *IEEE Transactions on Information Theory*, 61(3):1344–1350, 2015. 79

[58] Brendan F Klare, Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *CVPR*, pages 1931–1939, 2015. 133, 135

[59] J. M. Kohler and A. Lucchi. Sub-sampled cubic regularization for non-convex optimization. In *ICML*, 2017. 43

[60] T.G. Kolda and B.W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009. 18, 39

[61] J.B. Kruskal. Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and its Applications*, 18(2):95–138, 1977. 39

[62] Amit Kumar and Rama Chellappa. Disentangling 3d pose in a dendritic cnn for unconstrained 2d face alignment. In *CVPR*, 2018. 127

[63] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4), 2000. 156

[64] Y. Li, S. Liu, J. Yang, and M. Yang. Generative face completion. In *CVPR*, 2017. 115, 116, 143

[65] Y. Li, J. Yan, Y. Zhou, and J. Yang. Optimum subspace learning and error correction for tensors. In *European Conference on Computer Vision*, 2010. 39

[66] Z. lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *UIUC Technical Report*, 2009. 120, 122

[67] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *TPAMI*, 35(1):171–184, 2013. 83, 115

[68] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *ICML*, 2010. 29, 115

[69] G. Liu, Q. Liu, and P. Li. Blessing of dimensionality: Recovering mixture data via dictionary pursuit. *TPAMI*, 39:47–60, January 2017. 28, 29, 59, 115

[70] G. Liu and S. Yan. Active subspace: Toward scalable low-rank learning. *Neural Computation*, 24(12):3371–3394, 2012. 155

[71] Guangcan Liu and Ping Li. Low-rank matrix completion in the presence of high coherence. *TSP*, 64(21):5623–5633, 2016. 113

[72] Guangcan Liu, Qingshan Liu, and Xiaotong Yuan. A new theory for matrix completion. In *NIPS*, 2017. 113

[73] Guangcan Liu, Huan Xu, Jinhui Tang, Qingshan Liu, and Shuicheng Yan. A deterministic analysis for lrr. *TPAMI*, 38(3):417–430, 2016. 115

[74] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208 – 220, 2013. 39

[75] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 139

[76] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *TIP*, 17(1):53–69, 2008. 10, 129, 144, 153

[77] R. Margolin, L. Zelnik-Manor, and A. Tal. How to evaluate foreground maps? *CVPR*, 2014. 89

[78] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, and Jordan Cheney. Iarpa janus benchmark–c: Face dataset and protocol. In *ICB*, 2018. 133, 135

[79] J. Mota, N. Deligiannis, and M. Rodrigues. Compressed sensing with side information: Geometrical interpretation and performance bounds. *GlobalSIP*, 2014. 30, 80

[80] J. Mota, N. Deligiannis, and M. Rodrigues. Compressed sensing with prior information: Strategies, geometry, and bounds. *TIT*, 2017. 115, 117

[81] P. Netrapalli, N. U. N, S. Sanghavi, A. Anandkumar, and P. Jain. Non-convex robust pca. In *NIPS*, 2014. 43, 45, 59

[82] M. Nielsen and I. Chuang. Quantum computation and quan-tum information. *Cambridge University Press*, 2010. 80

[83] U.N. Niranjan, A. Rajkumar, and T. Tulabandhula. Provable inductive robust pca via iterative hard thresholding. In *UAI*, 2017. 29, 43

[84] T. Oh, Y. Tai, J. Bazin, H. Kim, and I. Kweon. Partial sum minimization of singular values in robust PCA: Algorithm and applications. *TPAMI*, 38(4):744–758, 2016. 83, 122

[85] T. Oh, Y. Tai, J. Bazin, H. Kim, and I. S. Kweon. Partial sum minimization of singular values in robust pca: Algorithm and applications. *TPAMI*, 38:744–758, August 2015. 45

[86] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *BMVC*, volume 1, page 6, 2015. 132, 133, 136, 138, 139

[87] Ankur Patel and William A. P. Smith. 3d morphable face models revisited. In *CVPR*, 1999. 113

[88] V. Patel, T. Wu, S. Biswas, P. 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. 87, 126

[89] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 115, 116

[90] Julien Pérard, Cédric Leyrat, Florence Baudin, Emmanuel Drouet, and Marc Jamin. Structure of the full-length hcv ires in solution. *Nature communications*, 4, 2013. 15

[91] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *TOG*, 2003. 143

[92] Rajeev Ranjan, Ankan Bansal, Jingxiao Zheng, Hongyu Xu, Joshua Gleason, Boyu Lu, Anirudh Nanduri, Jun-Cheng Chen, Carlos D Castillo, and Rama Chellappa. A fast and accurate system for face detection, identification, and verification. *arXiv:1809.07586*, 2018. 136, 138

[93] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. In *arXiv:1703.09507*, 2017. 132

[94] Rajeev Ranjan, Vishal M. Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *TPAMI*, 2018. 166

[95] S. Gandyand B. Recht and I. Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):025010, 2011. 39

[96] S. Reich and A. Zaslavski. *Infinite Products of Operators and Their Applications*. the AMS and Bar-Ilan University, 2012. 57

[97] R Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM journal on control and optimization*, 14(5):877–898, 1976. 120

[98] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 143

[99] Janina S Ried, Janina Jeff M, Audrey Y Chu, Jennifer Bragg-Gresham, Jenny Dongen, Jennifer E Huffman, Tarunveer S Ahluwalia, Gemma Cadby, Niina Eklund, Joel Eriksson, Tõnu Esko, Mary F Feitosa, Anuj Goel, Mathias Gorski, Caroline Hayward, Nancy Heard-Costa, Anne U Jackson, Eero Jokinen, Stavroula Kanoni, and Ruth Loos. A principal component meta-analysis on multiple anthropometric traits identifies novel loci for body shape. 11 2016. 15

[100] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic. Raps: Robust and efficient automatic construction of person-specific deformable models. In *CVPR*, 2014. 29

[101] C. Sagonas, Y. Panagakis, S. Zafeiriou, and M. Pantic. RAPS: Robust and efficient automatic construction of person-specific deformable models. *CVPR*, 2014. 30, 83, 115, 122

[102] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 132, 133, 138, 139

[103] A.B. Sekuler and S.E. Palmer. Perception of partly occluded objects: A microgenetic analysis. *Journal of Experimental Psychology: General*, 121(1), 1992. 162

[104] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *WACV*, 2016. 133, 134

[105] A. Shabalin and A. Nobel. Reconstruction of a low-rank matrix in the presence of gaussian noise. *Journal of Multivariate Analysis*, 118:67–76, 2013. 85, 124

[106] N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Robust principal component analysis on graphs. *ICCV*, 2015. 30

[107] N. Shahid, N. Perraudin, V. Kalofolias, G. Puy, and P. Vandergheynst. Fast robust PCA on graphs. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):740–756, 2016. 30

[108] F. Shang, J. Cheng, Y. Liu, Z. Luo, and Z. Lin. Bilinear factor matrix norm minimization for robust pca: Algorithms and applications. *TPAMI*, PP:1–1, September 2017. 45

[109] F. Shang, Y. Liu, J. Cheng, and H. Cheng. Robust principal component analysis with missing data. *CIKM*, pages 1149–1158, 2014. 28, 30, 113

[110] A. Sharma, A. Kumar, H. Daume, and D. Jacobs. Generalized multiview analysis: A discriminative latent space. In *CVPR*, 2012. 133

[111] Sumit Shekhar, Vishal M Patel, and Rama Chellappa. Synthesis-based robust low resolution face recognition. In *arXiv:1707.02733*, 2017. 132

[112] J. Shen, S. Zafeiriou, G. G. Chrysos, J. Kossaifi, G. Tzimiropoulos, and M. Pantic. The first facial landmark tracking in-the-wild challenge: Benchmark and results. In *ICCVW*, pages 1003–1011, 2015. 130

[113] Daichi Shigemizu, Shintaro Akiyama, Yuya Asanomi, Keith A. Boroevich, Alok Sharma, Tatsuhiko Tsunoda, Kana Matsukuma, Makiko Ichikawa, Hiroko Sudo, Satoko Takizawa, Kouichi Ozaki, Takahiro Ochiya, and Shumpei Niida. Risk prediction models for dementia constructed by supervised principal component analysis using mirna expression data. *Communications Biology*, 2:77, 02 2019. 15

[114] V. Stanković, L. Stanković, and S. Cheng. Compressive image sampling with side information. *ICIP*, 2009. 79

[115] H. Sun, J. Wang, and T. Deng. On the global and linear convergence of direct extension of ADMM for 3-block separable convex minimization models. *Journal of Inequalities and Applications*, (227):227, 2016. 82, 121

[116] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *NIPS*, pages 1988–1996, 2014. 132, 139

[117] S. Taheri, V. Patel, and R. Chellappa. Component-based recognition of facesand facial expressions. *IEEE Transactions on Affective Computing*, 4:360–371, 2013. 40

[118] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014. 132, 139

[119] Kim-Chuan Toh and Sangwoon Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 6(615-640):15, 2010. 120

[120] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht. Low-rank solutions of linear matrix equations via procrustes flow. In *ICML*, 2016. 59

[121] G. Warnell, S. Bhattacharya, R. Chellappa, and T. Başar. Adaptive-rate compressive sensing using side information. *IEEE Transactions on Image Processing*, 24(11):3846–3857, 2015. 79

[122] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 136, 138, 139

[123] Cameron Whitelam, Emma Taborsky, Austin Blanton, Brianna Maze, Jocelyn C Adams, Tim Miller, Nathan D Kalka, Anil K Jain, James A Duncan, Kristen Allen, et al. Iarpa janus benchmark-b face dataset. In *CVPRW*, pages 592–600, 2017. 133, 135, 136, 138

[124] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, 2011. 133, 138

[125] J. Wright, A. Ganesh, S. Rao, and Y. Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *NIPS*, 2009. 91

[126] J. Wright, A. Ganesh, A. Yang, Z. Zhou, and Y. Ma. Sparsity and robustness in face recognition. *arXiv:1111.1014*, 2011. 54

[127] L. Wu, S. Hoi, R. Jin, J. Zhu, and N. Yu. Distance metric learning from uncertain side information with application to automated photo tagging. *ACMMM*, 2009. 79

[128] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *TIFS*, 2018. 136

[129] A. Wyner and J. Ziv. The rate-distortion function for source coding with side information at the decoder. *IEEE Transactions on Information Theory*, 22(1):1–10, 1976. 30, 114

[130] Weidi Xie, Shen Li, and Andrew Zisserman. Comparator networks. *ECCV*, 2018. 133, 136, 138

[131] Weidi Xie and Andrew Zisserman. Multicolumn networks for face recognition. *BMVC*, 2018. 133, 136, 138

[132] H. Xiong, T. Liu, and D. Tao. Diversified dynamical gaussian process latent variable model for video repair. In *AAAI*, 2016. 51

[133] H. Xu, C. Caramanis, and S. Sanghavi. Robust pca via outlier pursuit. *IEEE Transactions on Information Theory*, 58(5):3047–3064, 2012. 30, 113

[134] M. Xu, R. Jin, and Z. Zhou. Speedup matrix completion with side information: application to multi-label learning. In *NIPS*, 2013. 29, 30, 114

[135] N. Xue, Y. Panagakis, and S. Zafeiriou. Side information in robust principal component analysis: Algorithms and applications. In *ICCV*, 2017. 53, 115

[136] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *CVPR*, 2017. 115, 117

[137] L. Yang, Z. Huang, and X. Shi. A fixed point iterative method for low n-rank tensor pursuit. *IEEE Transactions on Signal Processing*, 61(11):2952 – 2962, 2013. 39

[138] X. Yi, D. Park, Y. Chen, and C. Caramanis. Fast algorithms for robust pca via gradient descent. In *NIPS*, 2016. 43, 45, 59, 66, 67, 68, 69

[139] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *SPL*, 23(10):1499–1503, 2016. 133

[140] L. Zhang, L. Wang, and W. Lin. Conjunctive patches subspace learning with side information for collaborative image retrieval. *IEEE Transactions on Image Processing*, 21(8):3707–3720, 2012. 79

[141] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tail. In *ICCV*, 2017. 139

[142] Jingjing Zheng and Zhuolin ; Rama Chellappa Jiang. Submodular attribute selection for visual recognition. *TPAMI*, 39(11):2242–2255, 2017. 126

[143] Z. Zhou, X. Li, J. Wright, E. Candès, and Y. Ma. Stable principal component pursuit. *ISIT*, 2010. 30, 113

[144] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2), 2006. 158