MATRIX COMPLETION: INFER MISSING ENTRIES

A DISSERTATION SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

2020

By
Wang Tingyue
Student id: 10439320

Department of Computer Science

Contents

Word Count: 18012

List of Tables

List of Figures

Abstract

MATRIX COMPLETION:
INFER MISSING ENTRIES
Wang Tingyue
A dissertation submitted to The University of Manchester
for the degree of Master of Science, 2020

In 2006, the Netflix company announced to launch a competition to find algorithm which could boost the accuracy of their movie recommendation system most, with one million dollars prize. As a typical low rank matrix completion problem, it has attracted increasing attention to machine learning area since then. Low rank matrix completion problem generally refers to estimate missing values in a matrix using a limited sample of entries and the recovered matrix should be consistent with the observed entries. More importantly, the recovered matrix should be low rank. In fact, this problem is of great importance in many other fields including signal processing, computer vision, machine learning, data mining. Over years, quantities of algorithms have been put forward to solve this problem, however, the majority of research papers for these algorithms focus more on specific algorithm rather than an overview of low rank matrix completion problem. As a review paper for low rank matrix completion problem, this essay attempts to enable readers to grasp the essence of the problem and different algorithms, so it analyses general low rank matrix completion problem and describes current low rank matrix completion algorithms including their pros and cons. Moreover, this paper classifies these algorithms into four types: **Naive Methods for missing values**, **Convex Relaxation Method**, **Direct Non-convex Optimization Method** and **Other Type LRMC Method**. It also implements the representatives of the last three type algorithms including **Singular Value Thresholding**, **Singular Value Projection**, **Alternating Steepest Descent** and **Schatten-p Norm Minimization** and evaluate these three algorithms' performance in terms of **efficiency**, **accuracy**, **robustness**. Through the experiment results, this paper summarizes the features of each type of low rank matrix completion algorithms. Convex Relaxation Method always has high accuracy by achieving global optimum but relative slow convergence speed caused by Singular Value Decomposition. Direct Non-convex Optimization Method may suffer from local optima, but by using Alternating Minimization technique it has extremely fast speed and it is resistant to noise. For Other Type method involves pseudo-inverse calculation has very slow speed when matrix size becomes larger. Based on the features, the paper

also provides guidelines when choosing algorithms under different circumstances by summarizing the characteristics of each algorithm in practical.

Keywords: low rank matrix completion, low rank matrix recovery, image recovery, recommendation system, rank minimization, nuclear norm minimization.

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

First of all, I would like to express my sincere thanks to supervisor Dr. Tingting Mu for her efforts and patience throughout my project. She gave me tremendous freedom and proper guidance towards this project.
Secondly, I also would like to thank the University of Manchester, my university provided me with strong support of relevant materials and professional advice about dissertation writing and information searching.
Finally, I want to thank my parents and friends for their company and encouragement during the outbreaks.

Chapter 1

Introduction

1.1 Project Description and Motivation

In the present time of information explosion, missing data is a common problem. This is evident in the case of carrying out surveys. Imaging that audience is required to rate some movies (score 1-5), some may leave many questions unanswered out of various reasons, for instance, they haven't watched it or they are reluctant to rate it. However, in this way, the uncompleted ratings are less representative and lead to difficulty in latter analysis and application such as advertisement campaigns. As a result, that motivates the research into inferring these missing values by existing ratings these days. Abstract the ratings from all interviewees for different movies to a two-dimension matrix with missing values, the row represents different movies, columns represent different users and the value in each position is rating, then recovering the missing values by small sample of the matrix is called Matrix Completion. However, the problem will be ill-posed with no constraint or expectation to the missing value, for there are numerous answer and no standard to evaluate their correctness, so it is essential to suppose that there is an underlying pattern or structure shared by all entries in the matrix. Fortunately, the matrix to be recovered is always low rank or approximately low rank in any applications, such as recommendation system and image recovery area. The reason is that in Recommendation System, studies proved that users' tastes towards different movies is mainly influenced by movies' type, decade, director and actor/actress, directors, and different products of the same type always share common characteristics, so users are very likely to share similar preferences and their ratings are possibly correlated with each other. And in Image Recovery field, there are always some big areas in a picture have the same colour or textures, which indicates the underlying low rank structure. Given low rank constraint, the Matrix Completion problem is largely simplified to Low Rank Matrix Completion(LRMC), for the recovered matrix with lowest rank should be the right one. Numerous problems can be formulated as low rank matrix problem, such as inferring 3D structure from motion[PD04] and sensor localization problem[KO10]. Nowadays, low rank matrix completion gains its popularity in many areas, including machine learning [YMNS07, ATM06], pattern recognition[ZL09], model reduction [L.07], control[MG97] and computer vision [CT92].

In order to guarantee the relative depth of this essay, I restrict the number of area covered in this project. So I only focus the low rank matrix completion in Recommendation System and Image

Recovery area, so before we proceed, we clarify the LRMC problem in these two areas in detail.

- **Recommendation system**
  Many e-commerce and information search websites utilize recommendation system to offer potential items to users based on their previous behavior. Although nowadays personalized recommendation is emphasized by more and more individuals especially in healthcare and education area, from the consideration about saving social resources and limited artificial intelligence, it is reasonable to assume that users always share similarities. Top-N recommendation and Rating prediction are two major scenarios in the application of recommendation system. The former Top-N recommendation is mainly used for shopping sites that do not receive explicit ratings, it focuses on recommending N columns of items that are similar to the user's previous preferences. While the goal of the latter problem (see Figure 1.1) is predicting the preference of users for unrated items based on their limited rankings collected so far and it is used in some websites with ratings, such as movie rating website and music rating websites. One famous example is the Netflix prize, The Netflix company launched a contest in 2006 to publicly solicit the best computer algorithms for its movie recommendation system, with a 1 million dollars prize for the first contestant to improve the accuracy of its existing recommendation system by 10 percent. More specifically, the task of this contest is completing the rating matrix with only 1.2% known entries. The main difficulty comes from the fact that the rating matrix involves 480K customers and 18K movies and each user only rated nearly 200 movies on average. It is this famous "Netflix prize" problem brought more and more attentions to recommendation system and matrix completion topic. Nowadays, Top-n recommendations are more in line with actual needs than rating predictions, but this project only focuses on low rank matrix completion algorithms recovering rating matrix, which is called collaborative filtering problem. Compared to normal low rank matrix completion problem, the obvious feature of recommendation system is large matrix size, limited observed entries which distributed randomly. More specifically, the rating matrix is always more than 1000×1000 dimensions and observed entries only take up less than 5% of the whole matrix. Currently, Methods designed for tackling this problem are mainly classified into memory based and model based algorithm. The main idea of memory based algorithm is calculating the similarity between items or users and recover the missing value with the rating of the most similar item or user. While the model based algorithm is using existing rating to train a machine learning model and use it to predict unknown ratings, which is more general and can also be used for other low rank matrix completion problems. In conclusion, model based rating prediction algorithms in recommendation system are within this essay's research content.

- **Image recovery**
  During the transmission and storage process of images, noise is inevitable added to original images. The noise can be uniformly distributed white Gaussian noise or text and logos superposed to the images (see Figure 1.2 Source: [XCHW14]). For the latter kind of noise, it is not uniformly distributed in the whole images, which is out of the scope of this project. To filter out the noise and recover the quality of images, one approach is replacing the noise pixel with

Ratings of Different Users for Different Movies						
Users \ Movies	The Godfather	Forrest Gump	Casablanca	Schindler's List	Star Wars	.....
Alice	5	1	?	3	?	
Bob	4	?	4	5	2	
Cecilia	?	3	2	?	5	
David	3	?	1	2	?	
Emily	?	4	1	3	?	
.....			.....			

Figure 1.1: Movie Ratings

zeros and use low rank matrix completion algorithm to estimate the missing values. What's more, the missing values caused by occlusions and tracking failures can also be recovered by low rank matrix completion method. It is widely acknowledged that every color-level image has its RGB value so it can be considered as a tensor and gray-scale image can be viewed as a matrix. In this project, I only deal with 2D gray-level image matrix. To recover original image matrix, algorithms can be divided into two types. One is naive methods to solve missing values, which replace the missing value by calculating the mean or median value of its neighbour pixels, which is not applicable to other low rank matrix completion problem. The other is training model using remaining pixel values to predict the missing values, which is within the scope of this essay.

Figure 1.2: Uniformly-distributed Noise and Block Noise on Lena Image

Over years, papers of various low rank matrix completion algorithms have been put forward. However, some algorithms are based on the knowledge from different disciplines such as compress sensing, graph analysis and convex optimization and some only concentrate on specific algorithm including the theory, convergence proof and implementation, which is not conducive to the beginners' overall understanding of low rank matrix completion problem. In fact, these are some overview essays, but they have limitations. For example, [DR16] only describe various low rank matrix completion algorithms in theory without implementation. [CC18] divides all LRMC algorithms into convex and

non-convex optimization-based algorithms, while [LJB19] divides them in to Nuclear Norm minimization and Frobenius Norm minimization algorithms. Compared to these existing overview papers on LRMC algorithms, this paper is distinguished for following reasons. First, I categorize current LRMC algorithms into four classes and describe characteristic of different types. Second, I not only clarify theorem of different algorithms but also implement representatives of each type to evaluate their performance in reality. Third, I compare the algorithms implemented and summarize how to choose LRMC algorithms under different circumstance.
In conclusion, this essay is a review paper that aims at classifying current prevalent LRMC algorithms within a standard framework and implement the representatives of different types to compare their performance in terms of efficiency, accuracy and robustness through noise. Based on the findings, this essay also provides guidelines about low rank matrix completion algorithm selection.

1.2 Project Aim and Objectives

This paper is a review paper for different low rank matrix completion algorithms, which can be used for both recommendation system and image recovery area. The ultimate goal of this project is to develop a classification system for these algorithms and summarize the feature of various types. By implementing some representative algorithm of each type and comparing their performance, this paper also aims to provide guidelines for choosing low rank matrix completion algorithm according to application scenario. To achieve this goal in an orderly manner, the following objectives need to be met:

a. Literature review on current low rank matrix completion algorithms, their features and application scenario.

b. Implement some algorithms of each type and develop a standard matrix completion system to evaluate their performance.

c. Generate random low rank matrix of different size to test the developed system over synthetic data.

d. Apply the developed system to recover both images and recommendation system data sets with missing-values to test the developed system over real data matrices.

e. Compare different algorithms in terms of efficiency, accuracy, and robustness through noise and summarize their pros and cons and suitable application scenarios.

f. Provide low rank matrix completion algorithm selection advice and future research direction based on the experiment results.

1.3 Project Scope

To achieve the aim and objectives declared in Section 1.2. The scope of this project is described as follows:

a. Create random low rank matrices with uniformly-distributed missing values. These matrices are generated to have different matrix size, rank, sampling rate to test their influence on algorithms' performance.

b. For recommendation system tasks, Movielens 1M is downloaded, which contains 6000 users' 1 million ratings for 4000 movies. For this project, only the first 500 users' ratings for the first 500 movies are used.

c. For image recovery tasks, both the standard image in computer vision "Lena" and "Yixiu Tower" photo I took in Shanghai are used. Similarly, Delete the entries randomly and uniformly to create matrix with missing values.

d. For robustness test, the white Gaussian noise is added to the image matrix, which is the random entries sampled from the Gaussian distribution $N(0, \sigma^2)$

e. As representatives of different types, Singular Value Thresholding, Singular Value Projection, Alternating Steepest Descent and Schatten-p Norm Minimization methods are implemented in the project.

f. The performance is evaluated in terms of efficiency, accuracy and robustness through noise. Specifically, the accuracy is measured by Relative Error for random matrix, Normalized Mean Absolute Error for Movielens data set and Peak Signal to Noise Ratio for image matrix and noisy image matrix. The running time is also recorded for efficiency test.

1.4 Mathematical Notations and Conceptions

This section describes mathematical notations and conceptions used in this paper.

1.4.1 Mathematical Notations

- The real numbers set is denoted by $\mathcal{R}$.
- The cardinality of a set S is denoted as $|S|$.
- Vectors are denoted as boldface lowercase letter such as $\mathbf{a}$ and a_i is its i-th coordinates.
- The l_0 norm of a vector $\mathbf{a}$ is denoted as $||\mathbf{a}||_0$, which is the number of nonzero entries in the vector.
- The l_1 norm of a vector $\mathbf{a}$ is denoted as $||\mathbf{a}||_1$, which is also known as Manhattan Distance or Taxicab norm. It equals to the magnitudes of entries in the vector.
- The l_2 norm of a vector $\mathbf{a}$ is denoted as $||\mathbf{a}||_2$, which is also known as the Euclidean norm. It equals to the squared root of the sum of squared entries in the vector.

- The matrix is denoted as the boldface uppercase letter such as $\mathbf{A}$. The ij-th entry of matrix $\mathbf{A}$ is denoted as A_{ij}. $\mathbf{A}^i$ and $\mathbf{A}_j$ represent the i-th row and j-th column of matrix $\mathbf{A}$ respectively. The transpose of matrix $\mathbf{A}$ is denoted as $\mathbf{A}^T$. More specifically, in this project. $\mathbf{M}$ represents the original matrix and $\mathbf{X}$ represents the recovered matrix.

- The identity matrix is donated as $\mathbf{I}$. If its order is p, the identity matrix can also be represented as $\mathbf{I}_{p\times p}$ or $\mathbf{I}_p$

- The singular values of a matrix $\mathbf{A} \in \mathcal{R}^{m\times n}$ is denoted as $\sigma_1(\mathbf{A}) \geq \sigma_2(\mathbf{A}) \geq ... \geq \sigma_{min(m,n)}(\mathbf{A})$ when they are listed in descending order.

- The trace of matrix $\mathbf{A}$ is denoted as $Tr(\mathbf{A})$, which is only defined on square matrix. For a square matrix $\mathbf{A} \in \mathcal{R}^{n\times n}$, $Tr(\mathbf{A})$ equals the sum of diagonal elements from upper left to the lower right of matrix $\mathbf{A}$. Mathematically,

$$Tr(\mathbf{A}) = \sum_{i=1}^{n} A_{ii}$$

  It has following properties:

$$Tr(\mathbf{A}) = Tr(\mathbf{A}^T)$$
$$Tr(\mathbf{A}^T\mathbf{B}) = Tr(\mathbf{A}\mathbf{B}^T) = Tr(\mathbf{B}^T\mathbf{A}) = Tr(\mathbf{B}\mathbf{A}^T)$$

- The Frobenius Norm of matrix is represented as $||\mathbf{A}||_F$. It is defined as the square root of the sum of squares of the the absolute value of all elements in a matrix. Sometimes, it is also known as Euclidean Norm. Mathematically, given a matrix $\mathbf{A} \in \mathcal{R}^{m\times n}$, the Frobenius Norm of matrix $\mathbf{A}$ is defined as

$$||\mathbf{A}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2}$$

  Besides, it also equal to the square root of the matrix trace of $\mathbf{A}\mathbf{A}^H$, where $\mathbf{A}^H$ is the conjugate transpose of matrix $\mathbf{A}$. Mathematically,

$$||\mathbf{A}||_F = \sqrt{Tr(\mathbf{A}\mathbf{A}^H)}$$

- The Spectral Norm of matrix $\mathbf{A}$ is denoted as $||\mathbf{A}||_2 = (\max_i \sigma_i(\mathbf{A}\mathbf{A}^H))^{1/2}$, where $\mathbf{A}^H$ is the conjugate transpose of matrix $\mathbf{A}$. It is also known as Operator Norm.

- The Schatten-p Norm of matrix $\mathbf{A} \in \mathcal{R}^{m\times n}$ is denoted as $||\mathbf{A}||_{Sp}$. Mathematically, it is defined as

$$||\mathbf{A}||_{S_p} = (\sum_{i=1}^{min(m,n)} \sigma_i^p)^{\frac{1}{p}} = (Tr((\mathbf{A}^T\mathbf{A})^{\frac{p}{2}})^{\frac{1}{p}}$$

  where σ_i is the i-th singular value of matrix $\mathbf{A}$

- The rank of matrix $\mathbf{A} \in \mathcal{R}^{m\times n}$ is denoted as $rank(\mathbf{A})$. It is the number of non-zero singular values of a matrix. Mathematically, rank is defined as

$$rank(\mathbf{A}) = ||\mathbf{A}||_{S_0} = \sum_{i=1}^{min(m,n)} \sigma_i^0$$

It is equal to Schatten-p Norm when $p = 0$. It represents the maximal number of linearly-independent columns in this matrix. If it is equal to the largest dimension of this matrix, then this matrix is said to be full-rank and can be inverted. If not, the matrix is called singular matrix and not able to be inverted.

- The Nuclear Norm of matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$ is denoted as $||\mathbf{A}||_*$. It is the sum of singular values of a matrix and often used to constrain the low rank property of a matrix. Mathematically, it is defined as

$$||\mathbf{A}||_* = ||\mathbf{A}||_{S_1} = \sum_{i=1}^{min(m,n)} \sigma_i = Tr((\mathbf{A}^T\mathbf{A})^{\frac{1}{2}}$$

It is equal to Schatten-p Norm when $p = 1$. It is also known as Schatten-1 Norm, Trace Norm, Ky-Fan r-Norm.

1.4.2 Mathematical Conceptions

- **Singular Value Decomposition**
  Singular Value Decomposition (SVD) is a factorization technique of matrix in order to reduce matrix dimension or extract features. The SVD of matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$ is

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^{\mathbf{T}}$$

Where $U \in \mathcal{R}^{m \times m}$, $V \in \mathcal{R}^{n \times n}$ are both unitary matrices, $\Sigma \in \mathcal{R}^{m \times n}$ is a rectangular diagonal matrix with non-negative singular values σ in descending order on its diagonal.

- **Convex Set**
  If for every vector $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ and $\lambda \in [0, 1]$, $(1-\lambda)\mathbf{x} + \lambda\mathbf{y} \in \mathcal{C}$ always holds, then the set $\mathcal{C} \in \mathcal{R}^p$ is a convex set (see Figure 1.3 Source:[PP17]).

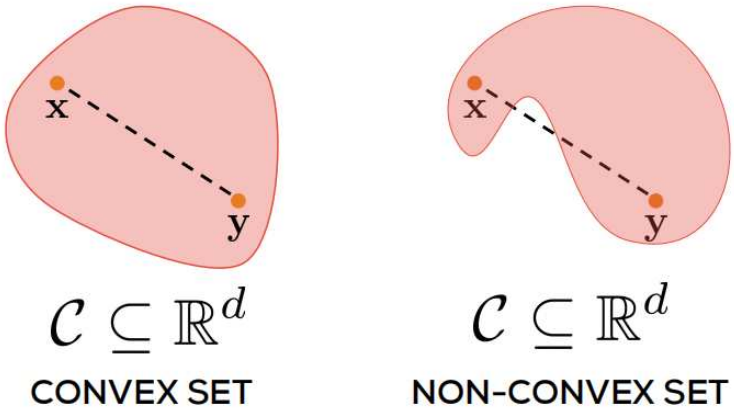

Figure 1.3: Convex Set and Non-convex Set

- **Convex Envelope**
  Given two functions $g(\bullet)$ and $f(\bullet)$, the largest convex function $f(x)$ is $g(x)$'s convex envelope

if $g(x) \leq f(x)$ always holds for all x. It is proved that the Nuclear Norm is the tightest convex envelope of rank on the unit ball of matrices with norm smaller than 1. Mathematically, given a matrix $\mathbf{X} \in \mathcal{R}^{m \times n}$ with $||\mathbf{X}||_2 \leq 1$, the Nuclear Norm is the rank.

- **Convex Function**
  For a continuously differentiable function f. For every $\mathbf{x}, \mathbf{y} \in \mathcal{R}^p$, if $f(\mathbf{y}) \geq f(\mathbf{x}) + < \bigtriangledown f(\mathbf{x}), \mathbf{y} - \mathbf{x} >$ always holds, where $\bigtriangledown f(\mathbf{x})$ is the gradient of $f(\mathbf{x})$, then the function $f(\bullet)$ is a convex function. A convex function is bounded by its tangent at all points(see Figure 1.4 Source:[PP17]).

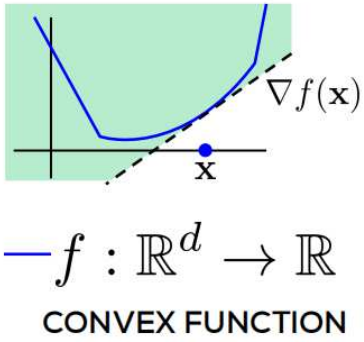

Figure 1.4: Convex Function

- **Convex Optimization Problem**
  If the objective function is a convex and the constraint set is a convex set, this optimization problem is said to be convex.

- **Non-convex Optimization Problem**
  If the objective function is non-convex and/or the constraint set is a non-convex set, this optimization problem is said to be non-convex.

1.5 Dissertation Structure

In following part of this essay, Chapter 2 is the literature review part and it attends to give an overview of low rank matrix completion problem. It includes the mathematical model of low rank matrix completion problem, the constraints on low rank matrix completion problem, the description of some typical low rank matrix completion methods and their classification according to the property of objective function and solving approaches. For each low rank matrix algorithm, I mainly introduce their main idea, advantages and disadvantages. In Chapter 3, I divide the whole project into four phases and the methodology used in each phase is introduced in this chapter. Moreover, the design of experiments is also described. Chapter 4 describes the implementation detail of each phase including the matrix generation, rank estimation method, algorithms outline, choice of parameters and evaluation result representation. Chapter 5 presents the experiment result of four matrix completion algorithms

implemented and discussions about the result. It gives the guidelines about choosing LRMC algorithms under different circumstances based on the experiment result. Conclusion and future research direction are discussed in Chapter 6.

Chapter 2

Background and Literature Review

2.1 Introduction

This chapter is a literature review chapter for low rank matrix completion problem. It starts with the introduction to low rank matrix completion mathematical model, then comes the constraints required for uniquely recovering the matrix. Next, it builds a classification system for current low rank matrix algorithms. It finally introduces typical low rank matrix completion algorithms of each type defined before.

In detail, the structure of this chapter is as follows. Section 2.2 introduces the derivation of mathematical model for low rank matrix completion problem. Section 2.3 gives the assumptions when performing low rank matrix completion. Section 2.4 outlines four main types of current low rank matrix completion algorithms according to their objective functions and solving techniques. Section 2.5 introduces the representative algorithm of each type defined before. Section 2.6 concludes this chapter.

2.2 Low Rank Matrix Completion Mathematical Model

The centric task for low rank matrix completion is recovering the matrix $\mathbf{M} \in \mathcal{R}^{m \times n}$ with missing values by using limited known entries. Significantly, the rank of recovered matrix $\mathbf{X}$ should be as small as possible, so a conventional approach is finding the matrix with lowest rank. Mathematically, the rank minimization problem for matrix can be written as

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t.\ \mathbf{X} \in L \tag{2.1}$$

Where $\mathbf{X} \in \mathcal{R}^{m \times n}$ and L is a convex set. This model has wide applicability, ranging from low-dimensional Euclidean embedding problem[NEY01] to minimum order linear system realization [MHS01]. If the constraint is an affine function, then the general affine rank minimization problem (ARMP) can be written as

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t.\ \mathcal{A}(\mathbf{X}) = b \tag{2.2}$$

where $\mathcal{A}$ is an affine transformation (linear map) from $\mathcal{R}^{m\times n}$ to $\mathcal{R}^{d}$. Many machine learning problems can be modelled as above problem. Some examples include low rank kernel learning, matrix completion and low-dimensional metric embedding. Considering the existence of noise, the constraint can be relaxed to

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t. \ ||A(\mathbf{X})-b||_2 \leq \theta \tag{2.3}$$

where θ is parameter. For Equation (2.2), if $\mathcal{A}$ is linear combinations of entries in matrix $\mathbf{X}$, this problem is low rank matrix sensing problem, which is not within the scope of this project. Else if $\mathcal{A}$ is constrained as random mapping, in other words, $\mathcal{A}$ is a sparse matrix with entries equal to 1 at observed location and 0 at missing values position, above affine rank minimization problem (ARMP) becomes low rank matrix completion problem (LRMC), which can be written as

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t. \ X_{ij} = M_{ij}, \ (i,j) \in \Omega \tag{2.4}$$

where Ω is the set of indices of sampled entries. Equation (2.4) is a special case of Equation (2.2). To solve affine rank minimization problem (ARMP), the affine transformation $\mathcal{A}$ must satisfy a restricted isometry property(RIP), while for low rank matrix completion problem, it must satisfy the restriction elucidated in Section 2.3. Here we define a sampling operator $\mathcal{P}_\Omega$ as

$$\mathcal{P}_\Omega(\mathbf{M}) = \begin{cases} M_{ij}, & if(i,j) \in \Omega \\ 0, & otherwise. \end{cases} \tag{2.5}$$

$\mathcal{P}_\Omega(\mathbf{M})$ represents the operation of projecting matrix into matrix space where all matrices in it all have the same value at non-zero positions. In other words, $\mathcal{P}_\Omega(\mathbf{M})$ has the same value with matrix $\mathbf{M}$ for sampled entries and zero value for unsampled entries. By using this sampling operator, the objective function becomes

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t. \ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.6}$$

The equality constraint in Equation (2.6) means the values at sampled entries in recovered matrix $\mathbf{X}$ should coincide with the original matrix $\mathbf{M}$. However, if the observed entries are corrupted by noise, the equality constraint is too strict, which may lead to overfitting problem. So some methods relax Equation (2.6) to

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t. \ ||\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(M)||_F \leq \varepsilon \tag{2.7}$$

In order to use optimization approach for objective function without constraint. Equation (2.7) can be synthesized to

$$Objective: \min_{X} \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{X}) - \mathcal{P}_\Omega(\mathbf{M})||_F^2 + rank(\mathbf{X}) \tag{2.8}$$

Actually, the low rank matrix $\mathbf{X}$ which minimizes the difference from original matrix $\mathbf{M}$ can be calculated by the leading singular components of the fully-observed matrix $\mathbf{M}$. However, for low rank matrix completion problem, the low rank matrix $\mathbf{X}$ can only be computed by limited observed entries in the original matrix $\mathbf{M}$ i.e. $\mathcal{P}_\Omega(\mathbf{M})$, so it is not able to directly calculate its singular components.

Then the low rank matrix completion problem becomes non-convex and can be easily be stuck in local minimum. What's more, the formulations involved in it always contains sum-squared loss equation, which is a non-convex optimization problem even when the matrix is fully known[SJ03]. Although in recommendation system and image recovery task, some other constraints such as non-negativity and sparsity also facilitate matrix completion, they also pose more challenges to this non-convex optimization problem.

2.3 Assumptions for Low Rank Matrix Completion

As discussed before, low rank matrix completion task is underdetermined and NP-hard without any constraint, so in order to solve it and get unique optimal solution, there are three more assumptions other than the low rank structure constraint. 1) Uniformly sampling of observed entries. 2) Lower bound on number of observed entries. 3) Incoherence. The details are covered in the section below.

- **Uniformly Sampling of Observed Entries**
  The low rank matrix completion problem always assumes that the observed entries are uniformly distributed over the matrix. In other words, the nonzero elements are not clustered in a certain area, so that a few entries are enough to recover the matrix. To achieve that, Bernoulli sampling can be used here, where every entry is determined to be samples or not through independent Bernoulli trial. This process ensures that all entries have equal possibility of being included in the sample. A simpler approach is sampling entries independently with replacement[EB08]. Actually, there are some algorithms able to solve uneven-distributed observed entries, such as the images corrupted by text or logos mentioned in Chapter 1, however, that is outside the scope of this project.

- **Lower Bound on Number of Observed Entries**
  Although only a small number observed entries needed to recover the matrix $\mathbf{M} \in \mathcal{R}^{m\times n}$ with rank r, there is still a lower bound. [EB08] proved that if $r \leq n^{\frac{1}{5}}$, there exists constant C and c that

$$Sampling\ number \geq Cd^{\frac{6}{5}} r \log d \tag{2.9}$$

  where $d = \max(m,n)$. With this minimum sampling number of entries, it is able to recover the original matrix with possibility at least $1 - cd^{-3}$. [Zhi15] calculated a more detailed expression. The author introduced a concept "Degree of Freedom" (DOF), which is the number of freely chosen variables in the matrix given the rank. For a matrix $\mathbf{M} \in \mathcal{R}^{m\times n}$ with rank less or equal to r, DOF equals to $(m+n)r - r^2$, then it can be proved that at least $4nr - 4r^2$ entries should be observed to uniquely recover the matrix $\mathbf{M} \in \mathcal{R}^{n\times n}$ with rank $r \leq \frac{n}{2}$. What's more, at least one entry should be observed in every row and column of matrix. The reason is that given the SVD of matrix $\mathbf{M}$ is $\mathbf{M} = \mathbf{U\Sigma V^T}$, if all entries in the i-th row are totally unsampled, then the i-th right singular vector can be any value, which in turn yields different recovered matrices all having the same sampled entries as original matrix. Analogously, if no entry is sampled in j-th column then the j-th left singular value can be arbitrary. So according to the Coupon Collector Effect,

the order for sampled entries should be more than $O(nlogn)$. Summarize the requirements, to recover the matrix uniquely, the lower bound on number of sampled entries should be on the order of $O(nrlogn)$.

- **Incoherence**
  The singular vectors of matrix should not be too sparse. In other words, all singular values should have similar magnitude, rather than a few singular values are much larger than others. Otherwise, more observed entries are needed to recover the matrix. Coherence in Compress Sensing area can be used to measure it, which is defined as

$$\mu(U) = \frac{n}{r} \max_{i<n} ||P_U e_i||^2 \tag{2.10}$$

  where $\mathbf{U}$ is a subspace of $\mathcal{R}^n$ of dimension r, e_i is the standard basis and $P_U$is the orthogonal projection onto matrix $\mathbf{M}$. Here the incoherence is always used to measure the difficulty of recovering the low rank matrix with limited sampled entries. High incoherence(μ is small) ensures that the sampled entries are not concentrated in a small area, so it is minimized when the observed entries are spread out in the matrix, which is consistent with assumption (1) above. Given SVD for matrix $\mathbf{M} \in \mathcal{R}^{m\times n}$ is $\mathbf{M} = \mathbf{U}\Sigma\mathbf{V}^{\mathbf{T}}$, [EB08] proved that $\mu(\mathbf{U})$, $\mu(\mathbf{V})$ should be no more than some positive μ_0 and $\Sigma_k u_k v_k^T$ is no more than $\mu_1\sqrt{\frac{r}{mn}}$ for some positive μ_1.

2.4 Classification for Current LRMC Algorithms

As the development of low rank matrix completion problem, an increasing number of algorithms are put forward to tackles different limitations of previous algorithms. As a result, a clear classification system for them is required in order to analyse them better. In this way, beginners are more likely to gain deeper and general insight to low rank matrix completion problem. There are various criteria to classify these algorithms. For example, [XCHW14] divided current algorithms into rank minimization method and matrix factorization method. In this essay, I divide all algorithms based on the property of its objective function and optimization approach to solve the objective function. As illustrated in Section 2.2, the original objective function of low rank matrix completion problem tries to find the recovered matrix with minimum rank and the known entries should be the same with original matrix. Even though considering the problem as a non-convex optimization problem facilitates algorithm designer to accurately model this learning problem, the direct minimization of rank is NP-hard and pose formidable challenge to latter solving stage. This is because compared to convex optimization, there lacks efficient tools to solve non-convex problems. What's worse, not only obtaining optimal solution is NP-hard, but also obtaining approximate solution is NP-hard[RPCI08].
Generally, for all non-convex optimization problems, there are two main methods to deal with it. The first type is **Convex Relaxation Method**, where the non-convex rank minimization problem is relaxed into convex object function[Dhr20]. One typical convex surrogate for rank minimization is Nuclear Norm minimization, and numerous methods intend to solve Nuclear Norm minimization, Nuclear

Norm regularized or Nuclear Norm penalized convex problem. In this way, traditional methods designed for convex optimization problems such as **Gradient Descent**, **Newton Method**, **Conjugate Gradient** can be used here [PP17].
The other approach is using various techniques to directly deal with the **Direct Non-convex Optimization** problem, which has attracted increasing interests in the area recently. Because it always produces comparative or even better performance than convex relaxation experimentally. As for the techniques for general non-convex optimization problem, there are mainly four types including **Non-convex Projected Gradient Descent**, **Alternating Minimization**, **Expectation Maximization** and **Stochastic Non-convex Optimization** [PP17]. Note that expectation maximization is a problem-specific variant of Alternating Minimization method to some extent. Among these four techniques, the first two can also be used for convex optimization problem. For our specific low rank matrix completion problem discussed in this project, non-convex optimization type algorithms can be mainly categorized to using non-convex Projected Gradient Descent technique and Alternating Minimization technique. Another technique for low rank matrix completion problem is **Manifold Optimization Method**, where the non-convex objective function is solved in manifolds, such as Riemannian manifold and Grassmannian manifold, for some non-convex functions become convex in manifold and different properties of manifold can be utilized to solve the problem. Moreover, **Basis Pursuit Method** borrowed from Compress Sensing area is also a choice, because both Compress Sensing and low rank matrix completion problem attempts to find low-rank solution.
For other LRMC algorithms which can not be simply classified to the above two type, they belongs to **Other Type LRMC Method**.

2.5 Current Low Rank Matrix Completion Algorithms

Before moving to introduce current low rank matrix completion algorithms, I would like to restate the LRMC mathematical model and introduce some troublesome approaches for it. As illustrated in Section 2.2, the most direct method is minimizing the rank of the matrix recovered with the constraint that the observed entries are same with the original matrix. The objective function is defined as

$$Objective: \min_{\mathbf{X}} rank(\mathbf{X}) \qquad s.t.\ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.11}$$

This formulation has both non-convex objective function and non-convex constraint, so the main difficulty to solve it mainly comes from the non-convexity and discontinuity of above model and therefore this is a NP-hard problem [LJB19], which can be solved by troublesome and naive combinatorial search[C.R10]. More specifically, This method first assumes the rank of the matrix is 1, which means all the columns of matrix **M** are linearly dependent. Then use this property to calculate the missing values. If the system has no solution to this assumption, then move to assume the rank of this matrix is 2 and check whether there is a solution again. Repeat this process until a solution is found. Obviously, this method does not require rank information in advance, for it assumes different rank every iteration. However, this combinatorial search method is not suitable for most low rank matrix

completion problem owing to its exponentially increasing computational complexity when matrix becomes larger. As clarified in Section 2.4 that to overcome the difficulty of rank minimization model, numerous methods have been put forward, which can be mainly divided into two main categories: **1) Convex Relaxation Method 2) Direct Non-convex Optimization Method**. For the algorithms which are hard to be classified, they belong to **3)Other Type LRMC Method**. However, I would like to mention some naive methods first when dealing with missing values briefly in order to broaden our horizon. Some of them even perform better than algorithms tailored for low rank matrix completion problem under some circumstances, but they are not emphasized in this essay.

2.5.1 Naive Methods for Missing Values

There are various methods to solve missing values in data processing area, which can be used for the problem described in this project. Despite they totally ignore the low rank structure of recovered matrix, they may outperform other methods under certain circumstance. Here various methods are introduced briefly as a quick review. First, the missing value can be replaced by the mean or mode value. Second, Hot deck imputation and K-nearest neighbour can be used here, which intends to find the most "similar" object and impute the missing entry with the same value. The similarity can be measured by Euclidean distance for image recovery problem, Pearson correlation coefficient or Cosine distance for recommendation system. Other methods such as Regression, Expectation maximization, Multiple Imputation, Sequential KNN can also be used here.

2.5.2 Convex Relaxation Method for LRMC Problem

By relaxing the non-convex problem into convex formulation, existing convex optimization tools can be used to solve the problem. However, if the problem is not relaxed properly, the solution produced by the relaxed formulation is not optimal for the original problem. What's worse, although the relaxed convex optimization problem can be solved in polynomial time, it is still not scalable for large scale matrix.

- **NNM: Nuclear Norm Minimization Model**
  One prevalent approach of rank relaxation is using Nuclear Norm as a convex surrogate for rank to make the problem tractable, which is the tightest convex relaxation of rank(see Figure 2.1 Source: *http://www.mit.edu/ parrilo/pubs/talkfiles/ISMP2009.pdf*). The objective function then becomes

$$Objective: \min_{\mathbf{X}} ||\mathbf{X}||_* \qquad s.t.\ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.12}$$

  The relaxation process is analogous to the process of l_0-norm to l_1-norm relaxation in Compress Sensing[JKB11, JBY+15, SKJB13]. The Nuclear Norm is the sum of singular values of a matrix, in other words, Nuclear Norm is the l_1-norm of singular values. In Compress Sensing, minimizing the l_1-norm of a vector leads to sparse solutions and a sparse singular value vector $\sigma(X)$ means that the most of singular values are 0. Because rank is the number of non-zero singular values, so the matrix $\mathbf{X}$ is consequently low rank. It has been proven that, provided the

singular vectors are weakly correlated with the canonical basis, Nuclear Norm Minimization can provide comparable answer to rank minimization model[MHR10, EB08]. In this way, it is able to find the global optimum and there are many convex optimization solvers can be used, such as semi-definite programming solvers (SDP3[TKCM99], SeDuMi[SJ98]) using interior point method. Because the Nuclear Norm minimization is convex, it is feasible to achieve the global optimum efficiently. Moreover, [EB08] proved that the solution of Nuclear Norm minimization can recover the low rank matrix with a high probability in theory as long as the uncompleted matrix satisfies the constraints described in Section 2.3. However, these methods have high computational complexity when matrix getting larger due to the computation of Newton Direction. Actually, SPD3 can only deal with matrix less than 100×100 dimension, which is barely enough for images with low resolution, but deemed insufficient for large rating matrix in recommendation system. Furthermore, these tool packages are not designed specifically for low rank matrix completion problem, so they do not make use of the low rank property of recovered matrix, which makes algorithms inefficient.

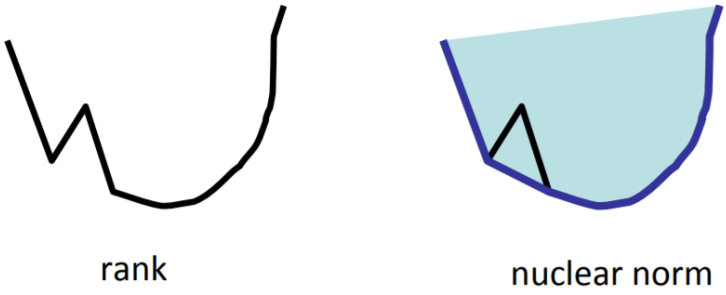

Figure 2.1: Nuclear Norm

- **SVT: Singular Value Thresholding**
  According to the limitations of semi-definite programming solver, [JFEZ10] put forward Singular Value Thresholding Method based on Uzawa's algorithm[AHU63]. Its objective function is still Nuclear Norm minimization as above, so it belongs to rank relaxation methods. The iterative updating approach to solve the nuclear minimization problem is defined below.

$$\begin{cases} \mathbf{X}^k = D_\tau(\mathbf{Y}^{k-1}) \\ \mathbf{Y}^k = \mathbf{Y}^{k-1} + \delta_k \mathcal{P}_\Omega(\mathbf{M} - \mathbf{X}^k) \end{cases} \tag{2.13}$$

Where $\mathbf{X} \in \mathcal{R}^{m \times n}$, $\tau > 0$, $\{\delta_k\}_{k \geq 1}$ of scalar step sizes and $\mathbf{Y}^0 = 0 \in \mathcal{R}^{m \times n}$ is an auxiliary matrix. $D_\tau(\mathbf{Y})$ is a nonlinear singular value shrinkage function using τ as threshold to cut off part of singular values of input matrix. More specifically, $D_\tau(\mathbf{X})$ involves Singular Value Decomposition of matrix $\mathbf{X} \in \mathcal{R}^{m \times n}$ with rank r and it is defined as

$$\begin{cases} D_\tau(\mathbf{X}) = \mathbf{U} D_\tau(\Sigma) \mathbf{V}^T \\ D_\tau(\Sigma) = diag(\{(\sigma_i - \tau)_+\}) \end{cases} \tag{2.14}$$

Where $t_+ = max(0,t)$. By doing this, $\{\mathbf{X}^k\}$ can coverage to nuclear minimization solution by selecting a large τ. The reason is that Lagrange multiplier approach proves that the upper iterative process is the solution to

$$Objective: \ \min \ \tau||\mathbf{X}||_* + \frac{1}{2}||\mathbf{X}||_F^2 \qquad s.t. \ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.15}$$

If a very large threshold τ is select, the first term dominates the objective function, then the objective function is equivalent with Nuclear Norm minimization. The process iterates until a stop condition is met, such are pre-defined iteration number or desirable error between recovered matrix and original matrix. By shrinking the singular values towards zero, the Nuclear Norm and rank of matrix $\mathbf{X}$ becomes lower and lower during the iterations. Moreover, the threshold used here can be calculated based on Stein's unbiased risk estimate, which facilitate the running process[CSLT12]. The main advantages of SVT include that first the soft-thresholding operation is applied to sparse matrix, which is true for typical matrix completion problem. Second, the rank of matrix during iteration is non-decreasing. These two merits in turn contributes to low computational complexity, small storage requirement and fast convergence speed. Moreover, the algorithm can reach convergence in polynomial time. However, It suffers from following weaknesses as well. First, Singular Value Decomposition involved in this algorithm results in slow convergence speed for large scale matrices and the selection of threshold is fixed. Second, it only works well for matrix with very low rank, for higher rank, it is slow and even fails. Third, despite it achieves low rank matrix in the end, the rank of intermediate matrix can be large during the iteration, which cause computation burden on this algorithm.

- **FPCA: Fixed Point Continuation with Approximate SVD**
  FPCA method is also designed to solve Nuclear Norm minimization problem and the constraint is relaxed to tolerate the noise in Affine rank minimization problem so the objective function is

$$Objective: \ \min_{\mathbf{X}} \ ||X||_* \qquad s.t. \ ||A(\mathbf{X}) - b||_2 \leq \varepsilon \tag{2.16}$$

its Lagrangian version is

$$Objective: \ \min_{\mathbf{X}} \ \mu||\mathbf{X}||_* + \frac{1}{2}||A(\mathbf{X}) - b||_2^2 \tag{2.17}$$

Where μ is a parameter commonly used in Lagrangian function. Similar to Singular Value Threaholding Method, it also make use of matrix shrinkage operator to solve the objective function. FPCA is based on Fixed point continuation method and it also uses a Monte Carlo approximate SVD to avoid computational complexity. The Fixed point continuation method is an extension for Fixed point iterative algorithm to accelerate the convergence, where the recovered matrix is updated as

$$\begin{cases} \mathbf{Y}^k = \mathbf{X}^k - \tau g(\mathbf{X}^k) \\ \mathbf{X}^{k+1} = D_{\tau\mu}(\mathbf{Y}^k) \end{cases} \tag{2.18}$$

Where $\mathbf{Y}$ is an auxiliary matrix, $D_{\tau\mu}(\bullet)$ is the same matrix shrinkage operator with the one used in the Singular Value Thresholding, $g(\mathbf{X}^k)$ is the gradient of function $\frac{1}{2}||A(\mathbf{X})-b||_2^2$ at point $\mathbf{X}^k$, which is equal to $A\times(A(\mathbf{X}^k)-b))$ and τ is the threshold. For Fixed point continuation technique, it improves Fixed point iterative by simply not performing SVD in stop condition. Other technique can also boost the algorithm, such as debiasing or Bregman iterative algorithm [SDL09]. When performing the Matrix shrinkage operation, Singular Value Decomposition for whole matrix is of high computational complexity, so FPCA utilizes fast Monte Carlo algorithm for approximate SVD. Its main idea is performing Singular Value Decomposition on a much smaller matrix which is not computational expensive and the small matrix is derived from the original large matrix. The main advantage of this approach is that compared to other Nuclear Norm minimization methods, it can deal with large-scale problem which can not be solved by Semidefinite programming solver. And it can deal with matrix with higher rank, which can not be solved by Singular Value Thresholding. Furthermore, it produces much faster and more robust result than SDP3.

- **IRLS: Iterative Reweighted Least Squares Minimization**
  The objective for Iterative Reweighted Least Squares algorithm is also Nuclear Norm minimization and it is designed for overcoming the high computational complexity of Nuclear Norm minimization Method as well. Assume all the singular values of matrix $\mathbf{X}$ are nonzero, this method rewrites the Nuclear Norm as

$$\begin{aligned} &||\mathbf{X}||_* = ||\mathbf{X}||_{S_1} = Tr[(\mathbf{X}\mathbf{X}^T)^{-\frac{1}{2}}(\mathbf{X}\mathbf{X}^T)] = ||\mathbf{W}\frac{1}{2}\mathbf{X}||_F^2 \\ &Where \mathbf{W} = (\mathbf{X}\mathbf{X}^T)^{-\frac{1}{2}} \end{aligned} \tag{2.19}$$

  Then this approach updates matrix $\mathbf{X}$ and weight matrix $\mathbf{W}$ in an iterative way.

$$\begin{cases} \mathbf{X}^k = arg\ min||(\mathbf{W}^{k-1})^{\frac{1}{2}}\mathbf{X}||_F^2 \quad s.t. \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \\ \mathbf{W}^k = (\mathbf{X}^k(\mathbf{X}^k)^T)^{-\frac{1}{2}} \end{cases} \tag{2.20}$$

  The initial weight matrix $\mathbf{W}_0 \in \mathcal{R}^{n\times n}$ is initialized randomly and the updating process of matrix $\mathbf{X}$ can be reformulated as a weighted least squares problem with linear constraint. However, the updating process of weight matrix $\mathbf{W}$ involves the calculation of inverse matrix, which can only be generated when the matrix is full-rank. For low rank matrix completion matrix, we always aim to find low rank solution, which makes the computation of weight matrix $\mathbf{W}$ ill-posed. To solve this, this algorithm sets a lower bound ε for all singular values to prevent them approaching zero. The main advantages of this method are its stable and robust performance through noise and its simplicity both in implementation and computation.[KM12, AMQ+18]

- **Truncated NNM: Truncated Nuclear Norm Minimization**
  Different from previous Nuclear Norm Minimization Method, which minimize all singular values together and may not approximate rank minimization well in reality, Truncated Nuclear

Norm Minimization Method only minimizes the sum of small singular values. More specifically, given the rank r, the top-r largest singular values are ignored and the rest small singular values are summed and minimized. The reason is that the rank of a matrix is the number of its nonzero singular values, while Nuclear Norm refers to the sum of its singular values. That means the calculation of rank treats all nonzero singular values equally, while for Nuclear Norm minimization calculation, large singular values dominate the result. Furthermore, [EB08, CT09] declare that the Incoherence property required by Nuclear Norm minimization is hard to satisfied in reality. The objective function for matrix $\mathbf{X} \in \mathcal{R}^{m \times n}$ is

$$Objective: \ min_X \ ||\mathbf{X}||_r \qquad s.t. \ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.21}$$

Where $||\mathbf{X}||_r = \sum_{i=r+1}^{min(m,n)} \sigma_i(\mathbf{X})$ However, unlike the Nuclear Norm minimization, the objective function above is non-convex. So we recast it into

$$Objective: \ min_X \ ||\mathbf{X}||_* - Tr(\mathbf{A}_l \mathbf{W} \mathbf{B}_l^T) \qquad s.t. \ \mathbf{X} = \mathbf{W} \quad \mathcal{P}_\Omega(\mathbf{W}) = \mathcal{P}_\Omega(\mathbf{M}) \tag{2.22}$$

Where $\mathbf{W}$ is a randomly-generated auxiliary matrix, $\mathbf{A}, \mathbf{B}$ are based on the Singular Value Decomposition of matrix $\mathbf{X}$. [YDJ+13] put forward three methods to solve it: Alternating Direction Method of Multipliers(ADMM), Accelerated Proximal Gradient Line search (APGL), Alternating Direction Method of Multipliers with Adaptive Penalty(ADMMAP). They share similar reconstruction error while ADMMAP has faster convergence rate. The reason for that is that ADMMAP considers two constraints simultaneously and uses adaptive penalty. Then the objective function now is

$$Objective: \ min_X \ ||\mathbf{X}||_* - Tr(\mathbf{A}_l \mathbf{W} \mathbf{B}_l^T) \qquad s.t. \ \mathcal{A}(\mathbf{X}) + \mathcal{B}(\mathbf{W}) = \mathcal{C} \tag{2.23}$$

Where $\mathcal{A}$ and $\mathcal{B}$ are linear operators defined as

$$\mathcal{A}(\mathbf{X}) = \begin{pmatrix} \mathbf{X} & 0 \\ 0 & 0 \end{pmatrix}, \mathcal{B}(\mathbf{W}) = \begin{pmatrix} -\mathbf{W} & 0 \\ 0 & \mathcal{P}_\Omega(\mathbf{W}) \end{pmatrix} \tag{2.24}$$

and

$$\mathcal{C} = \begin{pmatrix} 0 & 0 \\ 0 & \mathcal{P}_\Omega(\mathbf{M}) \end{pmatrix} \tag{2.25}$$

Then the updating rule of ADMMAP for above objective function is described briefly here. In iteration k:

$$\begin{cases} \mathbf{X}^{k+1} = \mathcal{D}_{\frac{1}{\beta_k}}(\mathbf{W}_k - \frac{1}{\beta_k}(Y_k)_{11}) \\ \mathbf{W}^{k+1} = \frac{1}{2\beta_k}\mathcal{P}_\Omega[\beta_k(\mathbf{M} - \mathbf{X}_{k+1}) - (\mathbf{A}_l^T \mathbf{B}_l + (\mathbf{Y}_k)_{11} + (\mathbf{Y}_k)_{22})] + \mathbf{X}_{k+1} + \frac{1}{\beta_k}(\mathbf{A}_l^T \mathbf{B}_l + (\mathbf{Y}_k)_{11}) \\ \mathbf{Y}_{k+1} = \mathbf{Y}_k + \beta_k(\mathcal{A}(\mathbf{X}_{k+1}) + \mathcal{B}(\mathbf{W}_{k+1}) - \mathbf{C})) \end{cases} \tag{2.26}$$

Where $\mathcal{D}$ represents the singular value shrinkage operator introduced in Singular Value Thresholding Method. $(\mathbf{Y}_k)_{11}$ represents the value in the value lying on the first row and first column in matrix $\mathbf{Y}_k$. β is the penalty parameter and it is updated adaptively in ADMMAP to avoid being too large or too small, which results in high computational complexity. The updating rule is

$$\beta_{k+1} = min(\beta_{max}, \rho\beta_k) \tag{2.27}$$

Where β_{max} is the upper bound of β_k. The value p is defined as

$$\rho = \begin{cases} \rho_0, & if \dfrac{\beta_k\{||\mathbf{X}_{k+1} - \mathbf{X}_k||_F, \mathbf{W}_{k+1} - \mathbf{W}_k||_F\}}{||C||_F} < \mathcal{K} \\ 1, & otherwise \end{cases} \tag{2.28}$$

Where $\rho_0 > 1$ is a constant and $\mathcal{K}$ is a threshold predefined. The matrix will be updated iteratively until the recovered matrix meets the standard. The main advantage for this method is its robustness through noise, faster convergence speed and low reconstruction error. What's more, they can even solve low rank matrix completion when the sampled entries are not uniformly distributed.

2.5.3 Direct Non-convex Optimization Method

Instead of relaxing the original problem into its convex surrogate, the non-convex problem can be solved directly by using some techniques. By using proper technique, the NP-hardness can be overcome and the algorithm can even yield optimal solution. In practical, this kind of methods always show better scalability and speed compared to Convex Relaxation Methods. For low rank matrix completion task, Typical techniques are Projected Gradient Descent, Alternating Minimization, Manifold Optimization and Basis Pursuit Method. In the sections bellow, I would like to further classify algorithms according to the technique they used.

Non-convex Optimization with Projected Gradient Descent Technique

- **SVP: Singular Value Projection**
  This method extends the Iterative Hard Thresholding algorithm in Compress Sensing[BD09]. It not only can solve low rank matrix completion problem, but also the general Affine Rank Minimization Problem (ARMP). For ARMP, It rewrites the Equation (2.2) to robust version as

$$Objective: Min \quad \psi(\mathbf{X}) = \frac{1}{2}||\mathcal{A}(\mathbf{X} - b)||_2^2 \quad Subjectto: \mathbf{X} \in \mathcal{C}(r) = \mathbf{X}: rank(\mathbf{X}) \leq r \tag{2.29}$$

  Where $\mathcal{C}(r)$ denotes the set of low rank matrices. Because using Singular Value Decomposition can compute the Euclidean projection on non-covex set $\mathcal{C}(r)$, Singular Value Projection algorithm use Projected Gradient Descent schema to update matrix $\mathbf{X}$ iteratively. More specifically, Gradient Descent is performed in each iteration and the intermediate result is projected on the

set of matrices with rank-r. The updating rule is

$$\mathbf{X}^k = \mathcal{P}_r(\mathbf{X}^{k-1} - \eta_{k-1} \bigtriangledown \psi(\mathbf{X}^{k-1})) = \mathcal{P}_r(\mathbf{X}^{k-1} - \eta_{k-1}\mathcal{A}^T(\mathcal{A}(\mathbf{X}^{k-1}) - b)) \tag{2.30}$$

Where k is the iteration number, η is a predefined parameter, $\mathcal{P}_r$ denotes the orthogonal projection on set $\mathcal{C}(r)$. By Eckart-Young-Mirsky theorem, it is well-known that $\mathcal{P}_r(\mathbf{X})$ can be calculated by using hard thresholding on the top-r singular values and singular vectors. In other words, only the first r singular vectors and singular values are retained.

$$\mathbf{X}^t = \mathbf{U}_r \Sigma_r \mathbf{V}_r^T \tag{2.31}$$

For specific low rank matrix completion. The objective is the same as Equation (2.7). Then for the updating rule, it becomes

$$\mathbf{X}^t = \mathcal{P}_r(\mathbf{X}^{t-1} - \frac{1}{(1+\delta)p}(\mathcal{P}_\Omega(\mathbf{X}^{t-1}) - \mathcal{P}_\Omega(\mathbf{M}))) \tag{2.32}$$

Now the step-size η_{t-1} in Equation (2.22) equals to $\frac{1}{(1+\delta)p}$ now, where p is the sampling rate and $0 \leq \delta \leq \frac{1}{3}$ is a parameter relevant to the sampling rate. The process will iterate until the stop condition is met. During the iterations, the rank of intermediate matrix is always low, which relief the computational burden. The main advantages for this approach include its robustness through both uniformly-distributed Gaussian noise and outlier noise, simplicity of comprehension and implementation and linear convergence rate. What's more, The Projected Gradient Descent can be effortlessly scaled to large matrix completion problem. All these merits makes SVP an outstanding method to solve low rank matrix completion problem. However, the rank information must be provided for this algorithm, which may not available sometimes and has to be estimated. And to guarantee the optimal solution, it should be initialized properly within the region of global optimum.

Non-convex Optimization with Alternating Minimization Technique

The basic idea of this technique is factoring the matrix into two or more variables and the low rank structure is eliminated. More specifically, the low rank matrix completion problem always presents the rating matrix by two factor matrices for users and items respectively. The premise behind the factor matrix with low dimension is that we suppose only a few factors influence the users' choice over items. Then it utilizes the fact that the non-convex matrix completion function is marginally convex(see Figure 2.2 Source: [PP17]), so the original single NP-hard problem is converted into intermediate margin optimization sub-problems, which is a simple least squares problem and can be solved by existing efficient solvers. However, Alternating Minimization type method always requires proper initialization tuning. Gradient Descent method can also be used, which is easy to carry out with slow convergence rate. Even though the descent version algorithm has hyper-parameters parameter to tune, it enables the users to control the progress of algorithm to some extent.

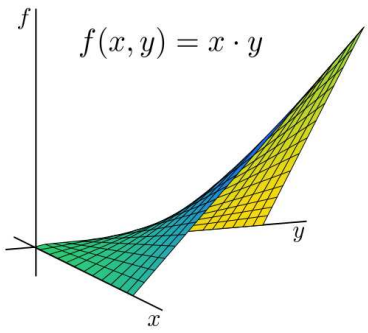

Figure 2.2: A Margin Convex Function But Not Jointly Convex

- **AltMinComplete: Alternating Minimization for matrix completion**
  Unlike the object function of above function, which aims to minimize the rank of recovered matrix. Alternating Minimization Algorithm directly finds rank-r matrix $\mathbf{X}$, with the same constraint that the known samples are the same. The general idea of this approach is that a matrix with missing value can be decomposed into two matrices with low rank r, and the object function is minimizing the Frobenius norm of difference between their product and original matrix, which is non-convex. Mathematically,

$$Objective : \min \ ||\mathcal{P}_\Omega(\mathbf{AB}) - \mathcal{P}_\Omega(\mathbf{M})||_F^2 \tag{2.33}$$

  Where $\mathbf{A} \in \mathcal{R}^{m\times r}$ and $\mathbf{B} \in \mathcal{R}^{r\times n}$ are factor matrices of recovered matrix. In this way, there is no constraint in the formulation. However, both matrices $\mathbf{A}, \mathbf{B}$ should be optimised rather than a single matrix $\mathbf{X}$. Based on this theory, quantities of methods were published, such as [Yeh09, CBSX12, YDRR08, YRC09]. [PPS12] improved these algorithms by introducing partition on sample set Ω. In detail, this approach first divides mask matrix Ω into $2k+1$ subsets, with each entry has equal possibility to belong to each subset. Then perform SVD on matrix $\frac{1}{p}\mathcal{P}_\Omega(\mathbf{M})$ to find its top-r left singular vector $\mathbf{A}^0$. Then clip all elements greater than $\frac{2\mu\sqrt{k}}{\sqrt{n}}$ to zero and orthonormalize the columns of $\mathbf{A}^0$. Then every time it fixes one factor matrix and update the other one, then switches them and repeats the process until the stop condition is met. In this way, each sub problem is a convex least-squares problem and admits a closed-form solution, which can be solved efficiently.

$$\begin{cases} \mathbf{B}_k = arg\ min||\mathcal{P}_\Omega(\mathbf{A}_{k-1}\mathbf{B}) - \mathcal{P}_\Omega(\mathbf{M})||_F^2 \\ \mathbf{A}_k = arg\ min||\mathcal{P}_\Omega(\mathbf{A}\mathbf{B}_{k-1}) - \mathcal{P}_\Omega(\mathbf{M})||_F^2 \end{cases} \tag{2.34}$$

  Finally, the recovered matrix is $\mathbf{X} = \mathbf{AB}$. Because this approach minimizes the smaller matrices rather than their product, its main advantage is that the system only needs to store smaller matrix, which relief the burden on storage and computation. And it is scalable and guarantees faster convergence. However, number of samples required by our analysis depend on the

condition number of the underlying matrix $\mathbf{M}$. The matrix factorization and Alternating Minimization technique is especially useful in recommendation system tasks, because two factor matrices can be interpreted as matrix for users and items respectively, then each factor matrix can be used to analyse users' behavior and items' popularity. [PPS12]

- **ASD : Alternating Steepest Descent Method**
  Actually, this method is an improvement to Alternating Minimization problem, so the objective function is similar.

$$Objective : \min\ f(\mathbf{A},\mathbf{B}) = \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{AB})||_F^2 \tag{2.35}$$

  Where $\mathbf{A} \in \mathcal{R}^{m\times r}$ and $\mathbf{B} \in \mathcal{R}^{r\times n}$ are the factor matrices of recovered matrix $\mathbf{X}$. However for alternating minimization step, it uses simple line-search along the Gradient Descent directions instead of minimizing least square problem in AltMinComplete Method above, which results in low computational complexity especially when the recover accuracy is moderate for all approaches. More specifically, It first initializes two random factor matrices $\mathbf{A} \in \mathcal{R}^{m\times r}$ and $\mathbf{B} \in \mathcal{R}^{r\times n}$. Using notation $f_B(\mathbf{A})$ donates $f(\mathbf{A},\mathbf{B})$ when matrix $\mathbf{B}$ is fixed and $f_A(\mathbf{B})$ donates $f(\mathbf{A},\mathbf{B})$ when matrix $\mathbf{A}$ is fixed, then the directions of gradient ascent are

$$\begin{cases} \bigtriangledown f_B(\mathbf{A}) = -(\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{AB}))\mathbf{B}^T \\ \bigtriangledown f_A(\mathbf{B}) = -\mathbf{A}^T(\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{AB})) \end{cases} \tag{2.36}$$

  If t_A, t_B are used for the steepest descent stepsizes for descent directions $-\bigtriangledown f_B(\mathbf{A})$ and $-\bigtriangledown f_A(\mathbf{B})$, then

$$\begin{cases} t_A = \dfrac{||\bigtriangledown f_B(\mathbf{A}||_F^2}{||\mathcal{P}_\Omega(\bigtriangledown f_B(\mathbf{A})\mathbf{B}||_F^2} \\ t_B = \dfrac{||\bigtriangledown f_A(\mathbf{B}||_F^2}{||\mathcal{P}_\Omega(\mathbf{A}\bigtriangledown f_A(\mathbf{B}))||_F^2} \end{cases} \tag{2.37}$$

  Once got the gradient ascent direction and steepest descent stepsize of factor matrices $\mathbf{A}$ and $\mathbf{B}$, the updating rule for them is

$$\begin{cases} \mathbf{A}_k = \mathbf{A}_{k-1} - t_{A_{k-1}} \bigtriangledown f_{B_{k-1}}(\mathbf{A}_{k-1}) \\ \mathbf{B}_k = \mathbf{B}_{k-1} - t_{B_{k-1}} \bigtriangledown f_{A_{k-1}}(\mathbf{B}_{k-1}) \end{cases} \tag{2.38}$$

  The above algorithm is repeated until the stop condition is met. Finally the recovered matrix is calculated as $\mathbf{X} = \mathbf{AB}$. In terms of recovered rank and running time, this method is competitive with other state-of-art matrix. Moreover, because it has low computational complexity in each iteration, it is also suitable for large scale matrix. It should be also noticed that it can solve both low rank and high-rank matrix completion problem[JK15]

- **MMMF: Maximum Margin Matrix Factorization**
  Maximum Margin Matrix Factorization Method is designed for recovering rating matrix. It also

follows the matrix factorization idea and attempts to minimize the Nuclear Norm, but unlike consider it as the tightest convex of rank, it interprets the Nuclear Norm as the equivalent of the norm of its factor matrices $\mathbf{A} \in \mathcal{R}^{m\times r}$ and $\mathbf{B} \in \mathcal{R}^{r\times n}$. The reason why Nuclear Norm is replaced by the norm of its factor matrices is that the Nuclear Norm is not differentiable which hinder the computation of descent direction in latter steps. This time the dimension of factor matrix is not constrained, that means in principle infinite number of factors is allowed, but actually only a few of them dominates. The objective function of MMMF algorithm also constrains the error between recovered matrix and original matrix. It is formulated as

$$Objective: \min\ f(\mathbf{A},\mathbf{B},\theta) = \frac{1}{2}(||\mathbf{A}||_F^2 + ||\mathbf{B}||_F^2) + C\sum_{r=1}^{R-1}\sum_{ij\in\Omega} h(T_i^r j(\theta_{ir} - \mathbf{A}_i\mathbf{B}_j)) \tag{2.39}$$

$$T_{ij}^r = \begin{cases} +1 & for \quad r > \mathbf{M}_{ij} \\ -1 & for \quad r < \mathbf{M}_{ij} \end{cases}$$

Where C is the penalty parameter, Ω is the sample set, R is the upper bound of the ratings, $h(\bullet)$ is the hinge loss that $h(z) = max(0, 1-z)$, θ is threshold for hard-margin setting and can be learned from the data. Because the first term is the norm of the factor matrices and the second term is the error, so the object function is a balance between the Nuclear Norm and the recover error. Given the objective function and ignore the non-differentiable hinge loss at one, the partial derivative of $\mathbf{A},\mathbf{B},\theta$ can be easy to compute and Gradient Descent technique can be perform. Note that the objective function is non-convex, so global minimum is not guaranteed. To overcome the non-differentiability of hinge function at one, [SRJ04] put forward Smooth Hinge function (see Figure 2.3 Source: [SRJ04]) , which facilitate the Gradient Descent step and it is not sensitive to outliers like Hinge loss. The main advantage of this approach is its scalability over large-scale with comparative recover accuracy. However, the objective function is non-convex, which may suffer from local optima.

This approach is mainly used for collaborative filtering in recommendation system. The main

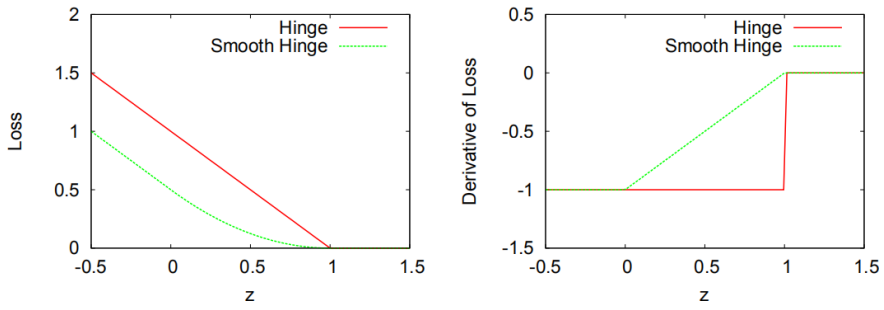

Figure 2.3: Hinge Loss and Smooth Hinge Loss Function(left) and Their Gradients(right)

feature of this problem is its high sparsity. In other words, the majority of its entries are unknown. An approach to address it penalizes the squared Frobenius norms of factor matrices.

The main advantages of this approach are its fast convergence rate and scalability when dealing with large-scale problems.

- **LMaFit: Low Rank Matrix Fitting**
  LMaFit also adopts the matrix factorization idea and it uses a nonlinear successive over-relaxation (SOR) technique to accelerate the algorithm that only deals with a simple linear least squares problem in every iteration. To avoid troublesome Singular Value Decomposition, this approach only minimizes the Frobenius norm of the difference between the recovered matrix and original matrix. By represents the recovered matrix $\mathbf{X} \in \mathcal{R}^{m\times n}$ by its factor matrices $\mathbf{A} \in \mathcal{R}^{m\times r}$ and $\mathbf{B} \in \mathcal{R}^{n\times r}$. The objective function is defined as

$$Objective : \min \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{Z}) - \mathcal{P}_\Omega(\mathbf{A}\mathbf{B}^T)||_F^2 \qquad s.t.\ \mathcal{P}_\Omega(\mathbf{Z}) = \mathcal{P}_\Omega(\mathbf{M}) \tag{2.40}$$

  Where $\mathbf{Z} \in \mathcal{R}^{m\times n}$ is an auxiliary variable. In this algorithm, $\mathbf{A}, \mathbf{B}, \mathbf{Z}, r$ are all dynamically adjust. Then this approach adopts a nonlinear SOR-like schema, which has faster convergence rate compared to block Gauss-Seidel. Because it uses a weighted average of previous result and current Gauss-Seidel result. Mathematically, matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{Z}$ are updated as

$$\begin{cases} \mathbf{A}_{k+1} = \mathbf{Z}_k\mathbf{B}_k^T(\mathbf{B}_k\mathbf{B}_k^T)^\dagger \\ \mathbf{A}_{k+1}(\omega) = \omega\mathbf{A}_{k+1} + (1-\omega)\mathbf{A}_k \\ \mathbf{B}_k = (\mathbf{Z}_k(\omega)^T\mathbf{A}_{k+1}(\omega))^\dagger(\mathbf{A}_{k+1}(\omega)^T\mathbf{Z}_k) \\ \mathbf{B}_{k+1}(\omega) = \omega\mathbf{B}_k + (1-\omega)\mathbf{B}_k \\ \mathbf{Z}_{k+1}(\omega) = \mathbf{A}_{k+1}(\omega)\mathbf{B}_{k+1}(\omega) + \mathcal{P}_\Omega(\mathbf{M} - \mathbf{A}_{k+1}(\omega)\mathbf{B}_{k+1}(\omega)) \end{cases} \tag{2.41}$$

  Where k is the iteration number, $\mathbf{A}^\dagger$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{A}$, ω denotes the weight. Considering updating the weight during the iterations can boost the efficiency, so [ZWY12] defined a residual ratio as

$$\gamma(\omega) = \frac{||\mathcal{P}_\Omega(\mathbf{M} - \mathbf{A}_{k+1}(\omega)\mathbf{B}_{k+1}(\omega))||_F}{||\mathcal{P}_\Omega(\mathbf{M} - \mathbf{A}_k\mathbf{B}_k)||_F} \tag{2.42}$$

  If $\gamma(\omega) < 1$, it means the factor matrices produced by current weight reduce the difference between the recovered matrix and original matrix, so the weight does not need to change, otherwise, the weight should be increase to $min(\omega + \delta, \widetilde{\omega})$, where δ is the increment and $\widetilde{\omega}$ is the upper bound.
  The main advantages of this method are outstanding convergence speed and comparative accuracy compared to many Nuclear Norm minimization algorithms. Because Nuclear Norm Minimization Method always involves Singular Value Decomposition, which requires high computational complexity especially when the rank and the size of the matrix is increasing. However, the objective function is non-convex, which may result in local minima problem. To avoid this dilemma, the initial guess of parameters should be within the neighbour of global minimum where there is highly possible no other stationary point. By doing this, despite of the

non-convexity, the Gradient Descent Method only moves in a small neighbourhood and is very likely to achieve global optimum. Moreover, the initial rank r estimation is required. Fortunately, experiments prove not exact rank estimation does not influence the final performance a lot and it has comparative performance with Nuclear Norm Minimization Methods.

Non-convex Optimization with Manifold technique

Considering that some non-convex optimization problem becomes convex in manifold, this technique is also prevalent in non-convex optimization area. What's more, some special properties of manifold such as its gradient facilitate the recovery process. Riemannian manifold and Grassmannian manifold are used in many methods.

- **LRGeomCG:Smooth Riemannian Manifold Optimization**
  LRGeomCG takes the presence of noise into consideration, so it exploits the relaxed objective Equation (2.6). it requires rank information as a prior, so above objective function can be rewritten as

$$Objective: \min\ f(\mathbf{X}) = \frac{1}{2}||\mathcal{P}_\Omega(X) - \mathcal{P}_\Omega(M)||_F \quad s.t.\ \mathbf{X} \in \mathcal{M}_k := \{rank(\mathbf{X}) = k\} \tag{2.43}$$

  It is well known that $\mathcal{M}_k$ is a smooth manifold, which can be considered as the smooth part of the determinantal variety of matrices of rank less than or equal to k. What's more, the objective function f is also smooth, so the optimization problem above can be solved by the methods for Riemannian optimization problem. [Bar13] put forward Conjugate Gradient Descent Method on smooth Riemannian manifold to solve it. Its main idea is to iteratively search for the matrix closest to the original matrix in the matrix space with fixed rank. In detail, in every iteration, it computes the Riemannian gradient ξ_i at current point, which is the steepest ascent direction of $f(\mathbf{X}_i)$ restricted in the tangent space $T_{X_i}\mathbf{M}_k$

$$\xi_i = grad\ f(\mathbf{X_i}) \tag{2.44}$$

  Next in order to obtain search direction η_i, which is a linear combination of the Riemannian gradient and previous search direction η_{i-1}. This comes from Polak-Ribiere updating rule in nonlinear Conjugate Gradient.

$$\eta_i = -\xi_i + \beta_i \mathcal{T}_{X_{i-1}\to X_i}(\eta_{i-1}) \tag{2.45}$$

  Where $\mathcal{T}_{X_{i-1}\to X_i} : \mathcal{T}_{X_{i-1}}\mathcal{M}_k \to \mathcal{T}_{X_i}\mathcal{M}_k$ denotes transport previous Riemannian direction to current tangent space $\mathcal{M}_k$ Then use straight-line search to calculate the stepsize t_i.

$$t_i = arg\min_t f(\mathbf{X}_i + t\eta_i) \tag{2.46}$$

  Now the iteration direction and stepsize enable the calculation of next iteration point, however, the new point may not lie in the smooth manifold, so Retraction is needed, which maps the

tangent vector into manifold.

$$\mathbf{X}_{i+1} := R_{\mathbf{X}_i}(0.5^m t_i \eta_i) \tag{2.47}$$

Where $R_{\mathbf{X}_i} : \mathcal{T}_{X_{i-1}}\mathcal{M}_k \to \mathcal{M}_k$ is the retraction operation. Smart selection of parameter m facilitates convergence process, and one approach is using Armijo backtracking to find the smallest positive integer such that

$$f(\mathbf{X}_i) - f(R_{\mathbf{X}_i}(0.5^m t_i \eta_i)) \geq -0.0001 \times 0.5^m t_i < \xi_i, \eta_i > \tag{2.48}$$

The implementation detail is described in [Bar13]. The advantage of using Riemannian manifold structure is that some objective functions are non-convex in Euclidean space, but they are convex in Riemannian manifold. Moreover, the property of Riemannian manifold facilitate more efficient approach for solving specific problems. The reason why Conjugate Gradient Method is used is that compared to Steepest Gradient Descent Method with slow convergence rate and Newton Method with high computational complexity, Conjugate Gradient Descent Method requires less storage and no external tunable parameter with high stability. The main advantage of Smooth Riemannian Manifold Optimization Method is that it is simple to implement and fast to coverage with high accuracy. However, the starting point based on Riemannian manifolds is random and this algorithm requires rank information as a prior, but rank estimation is still an open question in this area.

- **OPTSPACE: Gradient Descent Algorithm on the Grassmannian Manifold**
  This approach uses spectral method involving Singular Value Decomposition and Grassmannian manifold's property to complete the optimization task. There are four main steps in the method and the first step is trimming the redundant elements in every row and column. More specifically, given the total number of known entries is $|\Omega|$, if the known entries in matrix $\mathbf{M} \in \mathcal{R}^{m \times n}$ is more than $\frac{2|\Omega|}{m}$ in a row or more than $\frac{2|\Omega|}{n}$ in a column, which means the row or column is over-represented, then the redundant entries should be deleted randomly and the trimmed matrix is denoted as $\mathbf{M}_t$. Because otherwise the singular vectors will be highly concentrated on these over-represented rows and columns in latter steps, which does not help to reveal the formation of unknown entries. The second step is estimating the rank of the recovered matrix using Singular Value Decomposition technique. Mathematically, it finds the index i that minimizes $R(i)$

$$R(i) = \frac{\sigma_{i+1} + \sigma_1 \sqrt{\frac{i}{\varepsilon}}}{\sigma_i} \tag{2.49}$$

  Where σ_i denotes the i-th singular values of trimmed matrix from last step in descending order, and ε is the average degree of the square matrix, defined as $\varepsilon = \frac{|\Omega|}{\sqrt{mn}}$. Then the idex i is the estimated rank $\hat{r}$. The theory supporting this approach is that the first r largest singular vectors are enough to reveal the main structure of the matrix, where r is the rank. As a result, there will be a distinct difference between the r^{th} singular value and the next singular value.The third step

is rank-$\hat{r}$ projection, which is defined as

$$\mathbf{P}_{\hat{r}}(\mathbf{M}_t) = \mathbf{U}_0\Sigma_0\mathbf{V}_0^T \tag{2.50}$$

Where $\mathbf{U}_0 \in \mathcal{R}^{m\times\hat{r}}$,$\mathbf{V}_0 \in \mathcal{R}^{n\times\hat{r}}$ and Σ_0 is a $\hat{r}\times\hat{r}$ diagonal matrix. They can be calculated directly using the singular vectors of trimmed matrix as

$$\begin{cases} \mathbf{U}_0 = \sqrt{m}[u_1, u_2, ..., u_{\hat{r}}] \\ \mathbf{V}_0 = \sqrt{n}[v_1, v_2, ..., v_{\hat{r}}] \\ \Sigma_0 = (\frac{1}{\varepsilon})diag(\sigma_1, \sigma_2, ..., \sigma_{\hat{r}}) \end{cases} \tag{2.51}$$

The forth step is performing Gradient Descent on the Grassmannian manifold. The objective function is defined as

$$\mathbf{F}(\mathbf{U},\mathbf{V},\Sigma) = \frac{1}{2}||\mathcal{P}_\Omega(\mathbf{M} - \mathbf{U}\Sigma\mathbf{V}^T) + \lambda\frac{1}{2}||P_{\Omega\perp}(\mathbf{U}\Sigma\mathbf{V}^T||_F^2 \tag{2.52}$$

Where $\Omega\perp$ is the complementary set of Ω.The second term helps when the matrix suffers from noise. For simplicity, $\lambda = 0$ is considered here. For factor matrices $\mathbf{U},\mathbf{V}$, they are updated by Gradient Descent over Grassmannian manifold and Σ is updated by least squares. The process is repeated until the the predefined stop criterion is met. Finally, the recovered matrix is represented as $\mathbf{X} = \mathbf{U}_k\Sigma_k\mathbf{V}_k^T$.

The main advantages of this approach are its robustness through noise and only a very small subset of entries are required to recover the original matrix. [NS12] put forward improved version which performs scaled gradients on Grassmannian manifold to speed up the convergence rate[RS09].

- **SET:Subspace Evolution and Transfer**

  This method is similar to OPTSPACE method, but it only searches for a column space (or row space) rather than searches both column and row spaces at the same time in the Grassimannian manifold. Which space is used depends on the dimensions of matrix and the sampling pattern. The objective function of the approach is defined as follows:

$$Objective : Min||\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{UW})||_F^2 \tag{2.53}$$

  Where $\mathbf{U} \in \mathcal{R}^{m\times r}$ and $\mathbf{W} \in \mathcal{R}^{r\times n}$. To solve it, this method perform Gradient Descent in the Grassmanain manifold. The whole algorithm can be divided into two parts: subspace evolution and subspace transfer. Especially, to boost the performance, this algorithm designs a mechanism to detect the existence of barriers and transfer the result column space across the barriers in subspace transfer step. More specifically, by decomposing the objective function into a sum of atomic functions, it is easy to found the discontinuities Gradient Descent process, which has not been dealt with before. To achieve better result, this approach redirects the search path

to overcome them. However, this method lacks detailed analysis about the number of iterations required to achieve convergence. Sometimes, it takes a large number of iterations which consumes time.[WOE11]

Non-convex Optimization with Basis Pursuit Technique

Considering the close relationship between low rank matrix completion problem and Compress Sensing, some methods can be borrowed and modified to solve our problem. Basis Pursuit Method is one of them, which also adopts matrix factorization idea. Its main idea is that a low rank matrix can be represented as a linear combination of many rank-one matrices.

- **OR1MP:Orthogonal Rank-one Matrix Pursuit**
  This method borrows the Orthogonal Matching Pursuit Method from vector case to solve matrix-level problem, which adopts Coordinate Gradient Descent idea for sparse learning problems. Its key idea is that any matrix can be considered as a linear combination of several rank-one matrices with unit Frobenius norm, so each iteration only solve a simple one-dimension problem. More specifically, the recovered matrix $\mathbf{X} \in \mathcal{R}^{m\times n}$ can be expresses as

$$\mathbf{X} = \mathbf{B}(\theta) = \sum_{i\in I} \theta_i \mathbf{B}_i \tag{2.54}$$

  Where $\mathbf{B}_i : i \in I$ is the set of all $m \times n$ rank-one basis matrix with unit Frobenius norm. Our aim is finding low rank matrix, so the number of nonzero item in θ should be minimized. Then the problem becomes

$$Objective : \min \ ||\theta||_0 \qquad s.t.\ \mathcal{P}_\Omega(\mathbf{B}(\theta)) = \mathcal{P}_\Omega(\mathbf{M}) \tag{2.55}$$

  Where $||\theta||_0$ represents the number of nonzero elements of vector θ. It is proved in [ZMJZJ14] that the rank-one matrix in iteration k can be calculated as $\mathbf{B}_k = \mathbf{u}_* \mathbf{v}_*^T$, where $\mathbf{u}_*$ and $\mathbf{v}_*$ are the top left and right singular vector of residual matrix, which is the difference between recovered matrix and original matrix. In every iteration k, we are now able to update the weights θ^k once we got the new basis matrix by solving the following least squares regression problem:

$$Objective : \min \ ||\sum_{i=1}^{k} \theta_i \mathbf{B}_i - \mathbf{M}||^2 \qquad s.t.\ \mathcal{P}_\Omega(\mathbf{B}(\theta)) = \mathcal{P}_\Omega(\mathbf{M}) \tag{2.56}$$

  By concatenating all observed entries in $\mathcal{P}_\Omega(\mathbf{M})$ and $\mathcal{P}_\Omega(\mathbf{B}_i)$ as vectors $\dot{\mathbf{y}}$ and $\dot{\mathbf{b}_i}$, the optimal solution for above objective function is

$$\theta^k = (\bar{\mathbf{B}}_k^T \bar{\mathbf{B}}_k)^{-1} \bar{\mathbf{B}}_k^T \dot{\mathbf{y}} \tag{2.57}$$

  Where $\bar{\mathbf{B}}_k = [\dot{\mathbf{b}_1}, \dot{\mathbf{b}_2}, ..., \dot{\mathbf{b}_k}]$. The updating process of rank-one basis matrix and weight matrix run alternatively until the predefined stop condition is met. The main advantages of this approach are inexpensive computation in every iteration and the rank is the only parameter

to adjust, so it is efficient and scalable for large matrix for comparative recover performance. Furthermore, this method can achieve linear convergence rate. [ZMJZJ14]

- **ADMiRA: Atomic Decomposition for Minimum Rank Approximation**
  This approach is a pursuit-type and greedy method. It extends the CoSaMP algorithm for l_0-norm minimization of vectors in Compress Sensing to rank minimization of matrix. Equation (2.3) is recast to below as its objective function.

$$Objective: \min \ ||\mathcal{A}(\mathbf{X}) - b||_2 \qquad s.t.\ rank(\mathbf{X}) \leq r \tag{2.58}$$

Experiment result reveals that the function above yields same result as Equation (2.3). The general idea of this method is that given a matrix $\mathbf{X} \in \mathcal{R}^{m \times n}$, it can be represented as linear combination of atoms $\mathbf{X} = \sum_j \alpha_j \psi_j$, which is called atomic decomposition. Every atom ψ_i has unit norm and every pair of atoms are orthogonal to each other. To calculate all the atoms of matrix $\mathbf{X}$, the matrix $\mathbf{X}$ can be represented by Singular Value Decomposition as $\mathbf{X} = \sum_{k=1}^{r} \sigma_k u_k v_k^T$, where r is the rank of matrix $\mathbf{X}$. Because a few largest singular values and corresponding singular vectors are enough to describe the main feature of this matrix. Then for each k, there exists ρ_k with unit norm. Mathematically,

$$atoms(\mathbf{X}) = \{\rho_k u_k v_k^T\}_{k=1}^{r} \tag{2.59}$$

This algorithm adopts the subadditivity property of the rank. In detail, if matrix $rank(\mathbf{X}) = r$, then $\mathbf{X}$ can be represented by a linear combination of rank-one matrices and there are r atoms that span X. In terms of its algorithm, it first performs Singular Value Decomposition on the residual matrix to find first $2r$ largest components, where the residual matrix is the difference between original matrix and recovered matrix $\mathbf{X}_{k-1}$ in last iteration(initial recovered matrix $\mathbf{X} = 0$).

$$\psi' = SVD(\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{X}_{k-1}), 2r) \tag{2.60}$$

Then concatenate this $2r$ atom set ψ' with r atom set $\hat{\psi}$ in last iteration(initial atom set $\hat{\psi} = \phi$). Then atom span will be $3r$.

$$\tilde{\psi} = \psi' \bigcup \hat{\psi} \tag{2.61}$$

Then calculate the intermediate matrix with $3r$ rank by using typical approach for standard least-square problem.

$$\tilde{\mathbf{X}} = argmin_X ||\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{X}_k)||_F, X \in span(\tilde{\psi}) \tag{2.62}$$

According to projection maximum principle, $2r$ atoms are pruned by performing Singular Value Decomposition on $\tilde{\mathbf{X}}$ and getting the first r components

$$\hat{\psi} = SVD(\tilde{\mathbf{X}}, r) \tag{2.63}$$

Finally, the recovered matrix $\mathbf{X_k}$ in this iteration is calculated by solving another least square

problem.

$$\mathbf{X}_k = argmin_X ||\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{X}_{k-1})||_F, X \in span(\hat{\psi}) \tag{2.64}$$

The process is repeated until the stop condition is met. The main advantages of this method are its efficiency and performance guarantee. Compared to those semidefinite solver(SDP3), it can solve large scale low rank matrix completion problem. However, this algorithm is relative complicated and slow.

2.5.4 Other Low Rank Matrix Completion Method

There are some methods which can hardly be classified into two types described above. Scahtten-p Norm Minimization Method is one of them, as introduced in Section 1.4.1, when $p = 1$, Schatten-p Norm is equal to Nuclear Norm, so the objective function is convex, while when $p = 0$, it is equal to rank, which results in a non-convex objective function. So I simply classify it as "Other Type" low rank matrix completion method.

- **Schatten-p Norm Minimization(non-convex)**
  As illustrated before in Section 1.4.1, Schatten-p Norm is the general form for both rank and Nuclear Norm. To bridge the gap between them, more attention is paid to the performance of this approach when $0 < p < 1$. The objective function of these method for matrix $\mathbf{X}$ is

$$Objective: \min_{\mathbf{X}} ||X||_{S_p}^p \qquad s.t.\ \mathcal{P}_\Omega(X) = \mathcal{P}_\Omega(M) \tag{2.65}$$

  Define a mask matrix $\mathbf{H}$, where $H_{ij} = 1$ if $(i, j) \in \Omega$ and $H_{ij} = 0$ if $(i, j) \notin \Omega$. Operator $\circ$ is the Hadamard product, which is an element-wise production, then the objective function becomes

$$Objective: \min_{\mathbf{X}} ||X||_{S_p}^p \qquad s.t.\ \mathbf{X} \circ \mathbf{H} = \mathbf{M} \tag{2.66}$$

  To solve this problem, Lagrangian function is derived for above objective function.

$$\mathcal{L}(\mathbf{X}, \wedge) = Tr(\mathbf{X}^T\mathbf{X})^{\frac{p}{2}} - Tr\wedge^T(\mathbf{X} \circ \mathbf{H} - \mathbf{M}) \tag{2.67}$$

  Taking the derivative of above function to zero with respect to $\mathbf{X}$ in order to obtain the expression of matrix $\mathbf{X}$, we get

$$\begin{cases} \mathbf{X} = \dfrac{1}{2}(\mathbf{H} \circ \wedge)\mathbf{D}^{-1} \\ \mathbf{D} = \dfrac{p}{2}(\mathbf{X}^T\mathbf{X})^{\frac{p-2}{2}} \end{cases} \tag{2.68}$$

  Define $\mathbf{H}^i$ is a diagonal matrix which takes i-th row of mask matrix $\mathbf{H}$ as its diagonal elements. Every row of matrix $\wedge$ is defined as

$$\wedge^i = 2m^i(\mathbf{H}^i\mathbf{D}^{-1}\mathbf{H}^i)^{-1} \tag{2.69}$$

  In the initialization step, matrix $\mathbf{D}$ is calculated by matrix $\mathbf{M}$. Then after using matrix $\mathbf{D}$ to

updating matrix $\mathbf{X}$ and matrix $\wedge$, matrix $\mathbf{X}$ is used to update $\mathbf{D}$ again. The process is repeated until convergence. The main advantage of this algorithm is that it aims to minimize the general form of rank and Nuclear Norm of the matrix, which is the Schatten-p Norm, so when $0 < p < 1$, it generates better result than above Nuclear Norm minimization algorithms. What's more, this approach does not require performing Singular Value Decomposition on matrix, which is always the main cause for high computational complexity, especially for large matrix. However, the pseudo-inverse calculation is also troublesome when matrix becomes larger.

2.6 Summary

Chapter 2 first clarifies the origin of low rank matrix completion by describing how it developed from general rank minimization to affine rank minimization and then to its final version step by step. Besides, consider the existence of noise, its mathematical model has a relaxed version. The mathematical model is of great importance for it is the prototype of the objective function of various algorithms. Based on the property of objective function, current low rank matrix completion methods are divided into four categories **Naive Method**, **Convex Relaxation Method** and **Direct Non-convex Optimization Method**, **Other Type LRMC Method** in Chapter 2, and the latter three are emphasized in this project. Furthermore, the algorithms of Non-convex Optimization type can be divided in to four types based on the technique used, which are **Projected Gradient Descent**, **Alternating Minimization**, **Manifold Optimization** and **Basis Pursuit Method**. Based on the classification system defined, numerous algorithms are introduced in terms of their main idea, performance, advantages and disadvantages. Among these algorithms, besides some **Naive Method**s for missing values such as KNN, **NNM, SVT, FPCA, IRLS, Truncated NNM** belong to Convex Relaxation Method, **SVP** belong to Direct Non-convex Optimization using Project Gradient Descent technique, **AltMinComplete, ASD, MMMF,LMaFit** belong to Direct Non-convex Optimization using Alternating Minimization technique, **LRGeomCG, OPTSPACE, SET** belong to Direct Non-convex Optimization using Manifold Optimization technique, **OR1MP, ADMiRA** belong to Direct Non-convex Optimization using Basis Pursuit Technique. **Schatten-p Minimization** is classified as Other Type LRMC Method.

Chapter 3

Research Methodology and Experimental Design

3.1 Introduction

According to the project objective declared in Section 1.2, this project aims to provide an overview of low rank matrix completion problem. Bearing this purpose in mind, this chapter will describe the research methodology used in this project and the reasons behind the choices. More specifically, I divide the project into four phases including **Background Research**, **Data Loading**, **Algorithms Implementation** and **Evaluation** (see Figure 3.1). The Background research phase is completed in Chapter 2, and the methodology of the rest three phases will be introduced in this chapter respectively. What' more, the design of experiments to compare the performance of different types of low rank matrix completion algorithms will also be discussed in this chapter.
In detail, the structure of this chapter is as follows. Section 3.2 describes the methodology used for Data Loading phase including the random matrix, image matrix, rating matrix and their sampling approach. Section 3.3 gives the methodology used for Algorithms Implementation phase including the choice of representative algorithms and programming language. Section 3.4 introduces the methodology for Evaluation phase including the criteria for their performance comparison. Section 3.5 describes the design of experiments, which is used for comparing the performance of algorithms selected. They are evaluated in terms of efficiency, accuracy and robustness through noise.

3.2 Data Loading

This project attempts to test various algorithms' performance on both synthetic matrix and real-word matrix from recommendation system and image recovery area. Besides, the matrix size, rank, sampling rate also play an important role in performance, which worth exploring in this project. As a result, synthetic matrix with different size, rank and sampling rate need to be generated and real-word data matrices should be downloaded. It is worth noting that the original images may not be low rank owning to the noise, so its low representation is desired for this project. Then as the random matrix generated and image matrix are original without missing values, manually sampling is also required

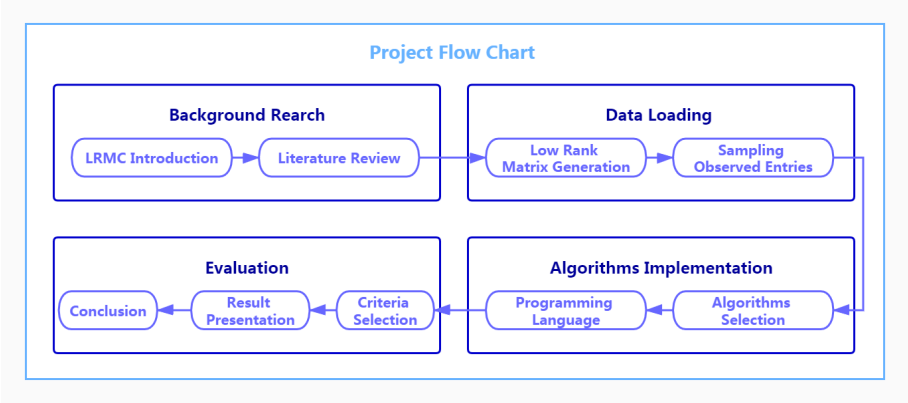

Figure 3.1: Project Flow Chart

to create missing values.

3.2.1 Synthetic and Real Data Loading

- **Random Low Rank Matrix**
  Given the task to create a random matrix $\mathbf{M} \in \mathcal{R}^{m\times n}$ with rank r, an approach is generating two random factor matrices $\mathbf{X} \in \mathcal{R}^{m\times r}$ and $\mathbf{Y} \in \mathcal{R}^{r\times n}$. Then the dot production of these two factor matrices produces the desired matrix. The reason why this approach generates desired dimension is easy to understand, for the dot production of a $m \times r$ matrix and a $r \times n$ matrix will definitely create a a $m \times n$ matrix. Then the reason why it generates desired rank is that the intermediate dimension is r for two factor matrices, after production, the new matrix is generated by r kinds of linear combination of entries in two matrices. As introduced in Section 1.4.1, the rank of a matrix represents the maximal number of linearly-independent columns in this matrix, so the rank of new matrix should be r. To test the algorithms' performance on matrices with different size and rank, I simply adjust the value of parameters m, n and r.

- **Image Matrix**
  For image recovery task, experiments are carried on both standard image and self-generated photo. First, for standard image, the gray-level "Lena" image (see Figure 3.2 left) is used[len], which is a standard and famous image in image processing area. It not only has high-rank part(feather on the hat) and low rank part(smooth skin), which is suitable for verifying various algorithms. Specifically, the gray-level Lena image matrix is 512×512 dimension and each entry is a value within $0 - 255$. However, when directly calculate the rank of this image, its rank is up to 511, which is caused by noise. One method to find its low rank representation is to perform Singular Value Decomposition on this matrix and then delete the majority small singular components. Finally use the remaining 50 largest singular values and corresponding vectors to calculate the low rank image matrix. The theorem supporting this approach is that some largest singular values contain the major information of an image and they are enough to represent the original data, so it is not very different with the original image(see Figure 3.2 right). Second, for self-generated image, the "Yixiu Tower" photo taken by me last summer is

used here. I first use "Format Factory" to shrink it and convert it to gray-level (see Figure 3.3). It is 640×480 dimension and I also use its low rank representation when $r = 50$.

Figure 3.2: Lena and Low Rank Representation with r=50

Figure 3.3: Yixiu Tower photo and its gray version

- **Rating Matrix**
  For recommendation system task, the project uses Movielens 1M data set [mov]. Movielens is a collection of movie ratings provided by Movielens users from the late 1990s to the early 2000s. The data includes movie ratings, movie metadata (genre and era) and demographic data about users (age, zip code, gender and occupation). This project only uses its u.ratings data set, which contains 6000 users' 1 million ratings(score 1-5) for 4000 movies. In detail, it contains four column data including user_id, movie_id, rating, timestamp. To create the required matrix with missing values, the first step is reshaping the data set into 2 dimension matrix, where each row number represents user_id, each column represents movie_id and each entry is the corresponding rating. The next step is using zero value to fill in the positions without rating.

3.2.2 Sampling Method

According to constraints described in Section 2.3, the observed entries and missing values should be uniformly-distributed in the matrix. However, for recommendation system task, the Movielens-1M

matrix already has missing value and does not need sampling, but the missing values are randomly distributed and pose challenges for low rank matrix completion algorithms. For random matrix and image matrix, they do not have missing values originally, so I need to uniformly sample he observed values manually. The sampling is carried out as follows, first generate a mask matrix with uniformly-distributed 1s and 0s according to the desired sampling rate, where 1s are in the position of the observed values and 0s are in the position of the missing value. Then multiply the mask matrix to the original full matrix $\mathbf{M}$ and the result $\mathcal{P}_\Omega(\mathbf{M})$ is the matrix with missing values to be recovered, where Ω is the Sample set. In this way, all known entries in $\mathcal{P}_\Omega(\mathbf{M})$ are the same as the original matrix $\mathbf{M}$ and missing values are all set to zero.

3.3 Implementation Methodology

In Implementation Phases, the methodology involves the how to choose algorithms to be implemented the programming language.

3.3.1 Algorithm Selection

As the objective declared in Section 1.2, numerical experiments are designed to evaluate the performance of different types of the low rank matrix completion algorithms, but it does not aim at carrying out comprehensive comparison among all algorithms mentioned, only some representatives of each type are implemented in the project. Moreover, current low rank matrix completion algorithms are classified into Convex Relaxation Method, Direct Non-convex Optimization Method and Other Type LRMC Method with various techniques in Section 2.4. Typical techniques involve Projected Gradient Descent, Alternating Minimization and so on. Therefore, in this project, one representative algorithm of each type is selected. For Convex Relaxation Method, Singular Value Thresholding is chosen and for Direct Non-convex Optimization Method, Singular Value Projection and Alternating Steepest Descent algorithm are selected, which use Projected Gradient Descent and Alternating Minimization technique respectively. For Other Type of low rank matrix completion algorithms, Schatten-p Norm Minimization Method is selected to implement.

3.3.2 Programming Language

PYTHON and MATLAB are two prevalent languages used for low rank matrix completion problem. Their pros and cons are described as follows.

- **PYTHON**
  PYTHON is a cross-platform computer programming language. Because of its conciseness, interpretability and extensibility, it gains an increasing popularity among programmers nowadays. An outstanding feature of PYTHON is that numerous packages provide PYTHON interface and some even designed especially for PYTHON, such as NumPy, SciPy and matplotlib. These scientific computing libraries provide fast matrix processing, numerical calculation and plotting

function for evaluation, which are adequate and essential for low rank matrix completion problem in this project. What's more, compared to MATLAB, it has interface to many complex libraries, which enables program to deal with more advanced tasks such as document management, interface design and network communication, although they are not used in this project. Another merit is its simplicity for learning and writing. That allows users to write codes that are easier to read and maintain. However, for low rank matrix completion task, its main drawback is that it can not deal with extremely large matrix, which related to its memory release schema.

- **MATLAB**
  MATLAB is an abbreviation of "matrix laboratory", which indicates that it is a professional programming language for matrix manipulation. Despite it is original created for numerical computing, many useful toolboxes are developed to extend its application such as MuPAD symbolic engine. Compared to PYTHON, it provides faster computation speed and its language is more closed to mathematical expression. Moreover, its core framework is written in C, so some functions and packages written in C can fit in MATLAB perfectly. However, MATLAB is not open source with high price to access, while PYTHON is totally free. Moreover, its code is not as easy as PYTHON, which causes difficulty in programming and maintaining.

According to their advantages and disadvantages listed above, I choose PYTHON as my programming language, because the matrix used in this project will not larger than PYTHON can bear and I have better understanding of PYTHON language.

3.4 Evaluation Methodology

As the objective declared in Section 1.2, the project attempts to evaluate the performance of different types of the low rank matrix completion algorithms in terms of their efficiency, accuracy and robustness. Consider that the performance of an algorithm is affected by various reasons, such as the matrix size, the rank of recovered matrix, the distribution of missing values, noise level and even the shape of matrix, the experiment schema just simply uses typical settings and compare their performance, regardless of their outstanding performance under different or extreme condition. The efficiency is measured in seconds, which is easy to understand. For accuracy, relative error between the recovered matrix and original matrix is used to measure it in random matrix completion task. Mathematically, Relative Error is defined as

$$\text{Relative Error} = \frac{||\mathbf{X}-\mathbf{M}||_F}{||\mathbf{M}||_F}$$

Where $\mathbf{X}$ and $\mathbf{M}$ represents the recovered matrix and the original matrix respectively. Note that the result is highly correlated with the stopping condition, so I not only compare the value of Relative Error for each algorithm, but also care about the changing pattern and trend of Relative Error as the program runs. However, for real data matrix in recommendation system task, there are missing values in the matrix I never know, so Relative Error can not be calculated. Instead, I compute the Normalized Mean Absolute Error(NMAE) which is calculated as

$$\text{NMAE}= \frac{||\mathcal{P}_\Omega(\mathbf{X})-\mathcal{P}_\Omega(\mathbf{M})||_F}{|\Omega|(v_{max}-v_{min})}$$

Where $|\Omega|$ donates the number of observed entries and v_{max}, v_{min} are the upper bound and lower bound of values in the matrix. While for image recovery task, Peak Signal to Noise Ratio(PSNR) is a typical criterion to measure the image quality. Mathematically, it is defined as

$$\text{PSNR} = 10\log_{10}(\frac{v_{max}^2}{MSE})$$

Where v_{max} represents the largest value in matrix $\mathbf{X}$, for gray-level image, it equals to 255. For matrix $\mathbf{X} \in \mathcal{R}^{m\times n}$, MSE is defined as

$$\text{MSE} = \frac{1}{mn}||\mathbf{X}-\mathbf{M}||_F^2$$

Finally, for robustness, it uses the same standard as accuracy, but the low rank matrix completion algorithms are carried out under noisy environment this time.

3.5 Experiment Design

This project aims to evaluate and compare the performance of different algorithms on synthetic random matrix with different matrix size, rank and sampling rate, real-word image matrix and noisy image matrix, real-word rating matrix. Moreover, their performance are measured by criteria introduced in Section 3.4. Based on this general idea, experiments are designed. Note that for simplicity, only square random matrix is used in this project. Experiment 1-4 are performed on synthetic matrix and Experiment 5-7 are performed on real-world matrix.

- **Experiment 1**: Experiment 1 is designed to evaluate the performance of these four algorithms on synthetic data, which is a small-scale random noiseless matrix with low rank and low sampling rate. So a random matrix $\mathbf{M} \in \mathcal{R}^{100\times 100}$ with rank 5 and sampling rate 0.5. In other words, there are 50% entries are observed. The change of the Relative Error over time is recorded for latter comparison.

- **Experiment 2**: Experiment 2 is designed to test the performance of these four algorithms on synthetic data. But this time it is a large-scale matrix. So a random matrix $\mathbf{M} \in \mathcal{R}^{1000\times 1000}$ with rank 5 and sampling rate 0.5 is generated.The change of the Relative Error over time is recorded. By comparing the result with experiment 1, the difference of four algorithms' performance on small matrix and large matrix can be explored.

- **Experiment 3**: Experiment 3 is designed to test the performance of these four algorithms on synthetic data. But this time it has higher rank. So a random matrix $\mathbf{M} \in \mathcal{R}^{100\times 100}$ with rank 100 and sampling rate 0.5 is generated.The change of the Relative Error over time is recorded. By comparing the result with Experiment 1, the difference of four algorithms' performance on low rank matrix and high rank matrix can be explored.

- **Experiment 4**: Experiment 4 is designed to evaluate the performance of these four algorithms on synthetic data. But this time it is a matrix with higher sampling rate. So a random matrix $\mathbf{M} \in$

$\mathcal{R}^{100\times100}$ with rank 5 and sampling rate 0.8 is generated. In other words, there are 80% entries are observed. The change of the Relative Error over time is recorded for latter comparison. By comparing the result with Experiment 1, the difference of four algorithms' performance on matrix with different number of missing value can be explored.

- **Experiment 5**: Experiment 5 is designed to explore the performance of these four algorithms on real-word data matrix in image recovery task, so the Lena image and Yixiu Tower photo is used, which is famous standard image and self-generated photo respectively. As explained in Section 3.2.1, the low rank representation of images is used as input. The change of Peak Signal to Noise Ratio over time is recorded for latter comparison.

- **Experiment 6**: Experiment 6 is designed to explore the performance of these four algorithms on real-world data matrix in image recovery task, but this time the images in Experiment 5 are corrupted by Gaussian-distributed noise. To generate noisy image matrix, a Gaussian noise matrix is generated, which is the same size with the Lena image matrix. Then the white Gaussian noise is added to the original matrix, mathematically, $\mathbf{M}' = \mathbf{M} + \mathbf{W}$. $\mathbf{M}'$ is then used as the input for four algorithms. The change of the Peak Signal to Noise Ratio over time is recorded for latter comparison. By comparing the result with Experiment 5, the difference of four algorithms' performance on original image matrix and noisy image matrix can be explored.

- **Experiment 7**: Experiment 7 is designed to evaluate the performance of these four algorithms on real-word data matrix in recommendation task, so the Movielens rating matrix is used. For simplicity, only 500×500 truncated matrix is used here. The change of Normalized Mean Absolute Error over time is recorded for latter comparison.

The purposes of 7 experiments are summarized in Table 3.1 below.

Experiment	Factor
1	Synthetic matrix
2	Matrix size
3	Rank
4	Sampling Rate
5	Image matrix
6	Noise
7	Rating matrix

Table 3.1: Influence Factors on Performance in 7 Experiments

3.6 Summary

Chapter 3 introduced the methodology used in the latter three phases of this project. In Data Loading phase, low-rank random matrix generation, low-rank representation calculation for image matrix,

Movielens dataset conversion and sampling method are described in Section 3.2. In Algorithm Implementation phase, the reasons for choosing algorithms and programming language are illustrated in Section 3.3. In the final Evaluation phase, the criteria used to evaluate the performance of each algorithm on different matrices are introduced in Section 3.4. What's more, Chapter 3 also described the design of 7 experiments in order to test the performance of four algorithms in terms of efficiency, accuracy and robustness on different matrices, including synthetic matrices with different matrix size, rank, sampling rate, real-word image matrix, noisy image matrix and real-word rating matrix.

Chapter 4

Experiment Implementation

4.1 Introduction

As mentioned in Section 3.1, the whole project is divided into four phases including **Background Research**, **Data Loading**, **Algorithms Implementation** and **Evaluation**. This chapter will explain the implementation detail of the latter three phases based on the experiments designed in Section 3.5. In detail, Section 4.2 explains how to create different matrix with missing values as the input of different algorithms. Section 4.3 describes each low rank matrix completion method including their algorithms and parameter values selected. Moreover, the rank information should be provided in advanced, so rank estimation approach is also introduced here. Section 4.4 presents the evaluation process using the criteria introduced in Section 3.4.

4.2 Data Loading Implementation

- **Random Matrix**
  Experiment 1, 2, 3, 4 are carried out on random matrices. Based on the matrix generation method and sampling method described in Section 3.2, the details about random matrix generation process are as follows. Note that I only consider square matrix in this project for simplicity. First, identify the size $n \times n$, rank r and sampling rate p, which depends on the experiment purpose in different experiment. More specifically, for Experiment 1, 3, 4, the matrix size is set to be 100×100, while for Experiment 2, it is set to be 1000×1000. For Experiment 1, 2, 4, the rank is set to be 5, while it is set to be 100 in Experiment 3. The sampling rate is set to be 0.5 in Experiment 1, 2, 3, but it is equal to 0.8 in Experiment 4. Second, Using *numpy.random.randn*() function to create two factor random matrices of dimension $n \times r$ and $r \times n$ respectively, the dot production of these two matrices is the original matrix without missing values. Then using *numpy.random.choice*() function to create a matrix of the same size with original matrix, where p of its entries are 1s and the rest are 0s. Finally, Multiply the mask matrix to the original matrix to create the random matrix with missing values.

- **Image Matrix**

Experiment 5 and Experiment 6 tests the performance of low rank matrix algorithms on image recovery task. For Lena image, it is read using the function $PIL.Image.open()$. However it is not *ndarray* type required by some algorithms, so its value is obtained by function $getdata()$ and reshaped back to its original size. The sampling process is the same as that for random matrix above. However, because of the potential noise, the image matrix is not low rank originally, so the approach described in Section 3.2.1 is used to find its low rank representation. For Yixiu Tower photo, the process is similar, except that $PIL.Image.convert('L')$ function is performed first to obtain gray-level photo. For robustness evaluation in Experiment 6, $numpy.random.randn()$ function should be used to create a noise matrix of the same size with the original matrix. Add it to the original matrix and then perform sampling, the result matrix is used as input of algorithms.

- **Rating Matrix**
  Experiment 7 tests the performance of low rank matrix algorithms on real-word rating prediction task. The Movielens dataset is read into system line by line and the users ID, movie ID and ratings are stored in three lists respectively. These three lists in turn constructs a sparse matrix, which can be converted into *ndarray* latter if required. Note that the rating matrix is originally with missing values, so sampling step is omitted.

4.3 Algorithms Implementation Detail

4.3.1 Rank Estimation

As described in Section 2.5, quantities of algorithms for low rank matrix completion problem require rank information in advance. In the experiments designed for this project, random matrix is generated according to a fixed rank, so the rank is already known. While for image recovery task and ratings prediction task, we have to guess the rank. One method is using a random small value r as rank and increasing it by a fixed number(e.g. 5) in every iteration to run the algorithm, repeat the process that until the recovered matrix does not change violently, then the current r is the estimated rank. Similarly, another method is using a large rank at first, and then decrease it by a fixed number. For these two methods, [ZWY12] proved that the decreasing rank strategy performs better for matrix completion problem. Another method is counting the number of singular vectors with large singular value. Because rank is the number of singular values and [KMO10] proved that the top-r (rank = r) largest singular values are enough to represent the main structure of the matrix. That means that if the singular values are listed in descending order, there will be large gap between the r^{th} singular value and the next. Therefore, for image used in this project, we directly cut off small singular values and use its low rank representation. For Movielens data set, it is originally with missing values, so Singular Value Decomposition can not be performed, so I choose to the decreasing rank method.

4.3.2 Algorithm and Parameter Value Selection

Singular Value Thresholding, Singular Value Projection, Alternating Steepest Descent Method and Schatten-p Norm Minimization Method are illustrated in Section 2.5, hence this part focuses on their algorithms outline and parameter value selection. Moreover, this project is not designed for finding optimal parameter values used in every method, so empirically optimal values mentioned in relevant papers are directly used. For the number of iteration, it is defined differently in each algorithm, so I used the minimum iteration number for them to converge for each algorithm.

- **Singular Value Thresholding Method**
  The general idea of Singular Value Thresholding Method is using Singular Value Shrinkage Operator to minimize the Nuclear Norm of the recovered matrix. Other details are described in Section 2.5. The algorithm of Singular Value Thresholding Method for low rank matrix completion is outlined in Algorithm 1 below.

Algorithm 1 SVT: Singular Value Thresholding (Source: [JFEZ10])

Input: Sampled Matrix $\mathcal{P}_\Omega(\mathbf{M}) \in \mathcal{R}^{m\times n}$, Sample Set Ω, Step size δ, Tolerance ε, Threshold τ, Increment l and Iteration number k_{max}

Output: $\mathbf{X}^{opt}$

Set $\mathbf{Y}^0 = k_0\delta\mathcal{P}_\Omega(\mathbf{M})$
Set $r_0 = 0$
for $k = 1$ to k_{max} **do**
  Set $s_k = r_{k-1} + 1$
  repeat
    Compute $[\mathbf{U}^{k-1}, \Sigma^{k-1}, \mathbf{V}^{k-1}]_{s_k}$
    Set $s_k = s_k + l$
  until $\sigma^{k-1}_{s_k-l} \leq \tau$
  Set $r_k = max\{j : \sigma^{k-1}_j > \tau\}$
  Set $\mathbf{X}^k = \sum_{j=1}^{r_k}(\sigma^{k-1}_j - \tau)\mathbf{u}^{k-1}_j\mathbf{v}^{k-1}_j$
  if $||\mathcal{P}_\Omega(\mathbf{X}^k - \mathbf{M})||/||\mathcal{P}_\Omega(\mathbf{M})||_F \leq \varepsilon$ **then break**
  end if
  Set $\mathbf{Y}^k_{ij} = \begin{cases} 0 & if(i,j) \notin \Omega \\ \mathbf{Y}^{k-1}_{ij} + \delta(M_{ij} - X^k_{ij}) & if(i.j) \in \Omega \end{cases}$
end for
Set $\mathbf{X}^{opt} = \mathbf{X}^k$
return $\mathbf{X}^{opt}$

Where parameter k_0 is an integer which obey

$$\frac{\tau}{\delta||\mathcal{P}_\Omega(\mathbf{M})||_2} \in (k_0 - 1, k_0] \tag{4.1}$$

The values selected for Singular Value Thresholding Method are list in the Table 4.1 below.

- **Singular Value Projection**
  The general idea of Singular Value Projection Method is using Singular Value Decomposition to compute the Euclidean projection on non-convex set. Then it uses projected gradient schema

Parameters	Step size δ	Tolerance ε	Threshold τ	Increment l	Iteration number k_{max}
Values	2	10^{-4}	20000	5	100

Table 4.1: Parameter Value for Singular Value Thresholding Method

to update the matrices alternatively. Other details are described in Section 2.5. The algorithm of Singular Value Projection Method for low rank matrix completion is outlined in Algorithm 2 below.

Algorithm 2 SVP: Singular Value Projection (Source: [RPI09])

Input: Sampled Matrix $\mathcal{P}_\Omega(\mathbf{M}) \in \mathcal{R}^{m\times n}$, Sample Set Ω, Estimated rank r, Parameter δ, Sampling rate p, Tolerance ε, Iteration number k_{max}.

Output: $\mathbf{X}^{opt}$

1: Initialize all-zero matrix $\mathbf{X}_0 \in \mathcal{R}^{m\times n}$
2: **for** $k = 1$ to k_{max} **do**
3: $\quad \mathbf{Y}_{k+1} = \mathbf{X}_k - \frac{1}{(1+\delta)p}(\mathcal{P}_\Omega(\mathbf{X}_k) - \mathcal{P}_\Omega(M))$
4: $\quad$ Compute top r singular vectors of $\mathbf{Y}_{k+1}$: $\mathbf{U}_r, \Sigma_r, \mathbf{V}_r$
5: $\quad \mathbf{X}_{k+1} = \mathbf{U}_r\Sigma_r\mathbf{V}_r$
6: $\quad$ **if** $||\mathcal{P}_\Omega(\mathbf{X}_{k+1} - \mathbf{M})||/||\mathcal{P}_\Omega(\mathbf{M})||_F \leq \varepsilon$ **then break**
7: $\quad$ **end if**
8: **end for**
9: $\mathbf{X}^{opt} = \mathbf{X}_{k+1}$
10: **return** $\mathbf{X}^{opt}$

The values selected for Singular Value Projection Method are list in the Table 4.2 below. Note that the rank r is estimated using the method describe in Section 4.3.1, which varies over different matrices and the optimal value for δ must satisfy $0 < \delta < \frac{1}{3}$.

Parameters	δ	Tolerance ε	Iteration number k_{max}
Values	0.2	10^{-4}	1000

Table 4.2: Parameter Value for Singular Value Projection Method

- **Alternating Steepest Descent Method**

  Alternating Descent Method adopts matrix factorization idea, so it initializes two factor matrices. In every iteration, fix one matrix and the optimization function of the other factor matrix is now convex, and Steepest Gradient Descent Technique can be used. Other details are described in Section 2.5. The algorithm of Alternating Steepest Descent Method for low rank matrix completion is outlined in Algorithm 3 below.

  The values selected for Alternating Minimization Method are list in the Table 4.3 below. Note that the rank r is estimated using the method describe in Section 4.3.1, which varies over different matrices.

Algorithm 3 ASD: Alternating Steepest Descent (Source: [JK15])

Input: Sampled Matrix $\mathcal{P}_\Omega(\mathbf{M}) \in \mathcal{R}^{m\times n}$, Sample Set Ω, Estimated rank r, Tolerance ε.
Output: $\mathbf{X}^{opt}$

Initialize two random factor matrices $\mathbf{A}_0 \in \mathcal{R}^{m\times r}$ and $\mathbf{B}_0 \in \mathcal{R}^{r\times n}$
for $k = 1$ to k_{max} **do**
  $\bigtriangledown f_{B_i}(\mathbf{A}_i) = -(\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{A}_k\mathbf{B}_k))\mathbf{B}_k^T$
  $t_{A_k} = \frac{||\bigtriangledown f_{B_k}(\mathbf{A}_k)||_F^2}{||\mathcal{P}_\Omega(\bigtriangledown f_{B_k}(\mathbf{A}_k)\mathbf{B}_k)||_F^2}$
  $\mathbf{A}_{k+1} = \mathbf{A}_k - t_{A_k} \bigtriangledown f_{B_k}(\mathbf{A}_k)$
  $\bigtriangledown f_{A_{k+1}}(\mathbf{B}_k) = -\mathbf{A}_{k+1}^T(\mathcal{P}_\Omega(\mathbf{M}) - \mathcal{P}_\Omega(\mathbf{A}_{k+1}\mathbf{B}_k))$
  $t_{B_k} = \frac{||\bigtriangledown f_{A_{k+1}}(\mathbf{B}_k)||_F^2}{||\mathcal{P}_\Omega(\mathbf{A}_{k+1}\bigtriangledown f_{A_{k+1}}(\mathbf{B}_k))||_F^2}$
  $\mathbf{B}_{k+1} = \mathbf{B}_k - t_{B_k} \bigtriangledown f_{A_{k+1}}(\mathbf{B}_k)$
  $\mathbf{X}_{k+1} = \mathbf{A}_{k+1}\mathbf{B}_{k+1}$
  if $||\mathcal{P}_\Omega(\mathbf{X}_{k+1} - \mathbf{M})||/||\mathcal{P}_\Omega(\mathbf{M})||_F \leq \varepsilon$ **then break**
  end if
end for
$\mathbf{X}^{opt} = \mathbf{X}_{k+1}$
return $\mathbf{X}^{opt}$

Parameters	Tolerance ε	Iteration number k_{max}
Values	10^{-4}	1000

Table 4.3: Parameter Value for Alternating Minimization Method

- **Schatten-p Norm Minimization Method**
  Schatten-p Norm Minimization Method can be considered as a general method for non-convex rank minimization when $p = 0$ and convex Nuclear Norm minimization when $p = 1$. For optimization technique, it uses the Lagrangian version of the objective function and set the derivative of it to zero to calculate recovered matrix $\mathbf{X}$. It also defined an auxiliary matrix $\mathbf{D}$ and alternating updating matrices $\mathbf{X}$ and $\mathbf{D}$ during the iteration. Other details are described in Section 2.5. The algorithm of Alternating Steepest Descent Method for low rank matrix completion is outlined in Algorithm 4 below.

Algorithm 4 Schatten-p Norm Minimization (Source: [FHC12])

Input: Sampled Matrix $\mathcal{P}_\Omega(\mathbf{M}) \in \mathcal{R}^{m\times n}$, Sample Set Ω, Parameter$p$, Iteration number k_{max}.
Output: $\mathbf{X}^{opt}$

Define $\mathbf{H} \in \mathcal{R}^{m\times n}$ where $H_{ij} = 1$ for $(i,j) \in \Omega$ and $H_{ij} = 0$ for $(i,j) \notin \Omega$
Set $\mathbf{D}_0 = \frac{p}{2}(\mathcal{P}_\Omega(\mathbf{M})^T\mathcal{P}_\Omega(\mathbf{M}))^{\frac{p-2}{2}}$
for $k = 1$ to k_{max} **do**
  $\mathbf{X}_{k+1} = \frac{1}{2}(\mathbf{H}\circ\wedge_k)\mathbf{D}_k^{-1}$, where the i^{th} row of $\wedge_k$ is calculated as $\wedge_k^i = 2m^i(\mathbf{H}^i\mathbf{D}_k^{-1}\mathbf{H}^i)^-1$
  $\mathbf{D}_{k+1} = \frac{p}{2}(\mathbf{X}_{k+1}^T\mathbf{X}_{k+1})^{\frac{p-2}{2}}$
end for
$\mathbf{X}^{opt} = \mathbf{X}_{k+1}$
return $\mathbf{X}^{opt}$

The values selected for Schatten-p Norm Minimization Method are list in the Table 4.4 below.

Parameters	p	iteration number k_{max}
Values	0.2	10

Table 4.4: Parameter Value for Schatten-p Norm Minimization Method

4.4 Evaluation Implementation Detail

Based on the 7 experiments designed in Section 3.5 and the Evaluation criteria selected in Section 3.4, four algorithms are performed on different matrices and their performances are measured by different metric. In order to compare their results intuitively and visually, their statistic is depicted in a figure for every experiment. Especially, for image recovery task in Experiment 3 and Experiment 6, the recovered images are shown for readers to evaluate the results by their eyes. Although in this way the judgement is influenced by some objective factors, it is more direct and intuitive. The matrix information and axes of result figure of each experiment are shown in the Table 4.5 below.

Experiment	Matrix Type	Dimension	Rank	Sampling Rate	X-axis	Y-axis
1	Random	100×100	5	0.5	Time	Relative Error
2	Random	1000×1000	5	0.5	Time	Relative Error
3	Random	100×100	100	0.5	Time	Relative Error
4	Random	100×100	5	0.8	Time	Relative Error
5.1	Lena Image	512×512	50	0.5	Time	PSNR
5.2	Yixiu Tower Photo	640×480	50	0.5	Time	PSNR
6.1	Noisy & Lena Image	512×512	50	0.5	Time	PSNR
6.2	Noisy & Tower Photo	640×480	50	0.5	Time	PSNR
7	Movielens Rating	100×100	20	/	Time	NMAE

Table 4.5: Matrix and Criteria Used in Each Experiment

4.5 Summary

Chapter 4 describes the implementation details in Data Loading phase, Algorithm Implementation and Evaluation Phase in this project. In data loading phase, the steps for generating random matrix, loading image matrix, finding low rank representation and Converting rating dataset are explained in detail. In Method Implementation phase, many methods require rank information as input, so some rank estimation methods are introduced. Moreover, the algorithms, parameter values are also provided. In Evaluation phase, the result of each algorithm is plotted in order to compare their performance intuitively. The matrix information and axes setting of the figure in each experiment are listed at the end of Chapter 4. The code for six experiments has been uploaded to **GitHub**, and the link is *https://github.com/Jane115/pycharmproject/tree/master/venv/WTYprogram*

Chapter 5

Experiment Result and Discussion

5.1 Introduction

Chapter 5 represents the results of 7 experiments designed in Section 3.5. Moreover, Discussions on the results and guidelines for choosing various low rank matrix algorithms are also provided in this section.
In detail, Section 5.2 provides the equipment and software version used in this project. Section 5.3 presents the results of 7 experiments. More specifically, their performance curves are plotted respectively and the discussion towards the result is also given at the end of each result image. Then Section 5.3 provides the general rules for choosing low rank matrix completion under various situation.

5.2 Equipment and Software

Before presenting the experiments result, the equipment information and software version is provided below for readers to reproduce the result.

- CPU: Intel Core i5-4210H CPU 2.90GHz
- Memory: 8GB
- Operating system: Windows 8, 64 bit
- Python Version: 3.7.4

5.3 Experiment Result Representation and Discussion

In order to compare the result more directly, the tendency of Relative Error, NMAE and PSNR vary over time are drawn and the final optimal result of each algorithm is recorded for comparison. However, only Experiment 1 drew curves of four algorithms in one image, because the time consumed and performance are quite different among these four algorithms in the rest experiments and present their results in one image is not clear, their curves are plotted respectively. Experiment 1-4 are performed on Synthetic Matrix, while Experiment 5-7 are performed on real-word matrix.

5.3.1 Experiment 1: Performance on Synthetic Matrix

- **Result**

  In Experiment 1, four algorithms are performed on a 100×100 noiseless random matrix with rank 5 and sampling rate 0.5. The Relative Error changes against Time is shown in Figure 5.1 and their final minimum Relative Error is shown in Table 5.1

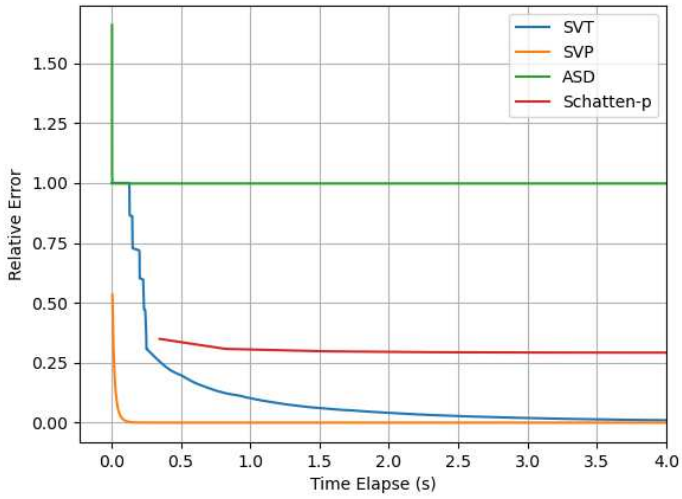

Figure 5.1: Performance of Four Algorithms on Synthetic Matrix

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
Relative Error	0.010576	0.000136	0.998972	0.284584

Table 5.1: Optimal Relative Error Achieved by Four Algorithms

- **Discussion**

  It can be observed that all algorithms are able to achieve low Relative Error, which indicates that they are all effective algorithms for low rank matrix completion problem. In terms of accuracy, which is measured by their final Relative Error in Table 5.1, Singular Value Projection Method achieves least Relative error, then comes Singular Value Thresholding Method and Schatten-p Norm Minimization Method, while Alternating Steepest Descent Method performs worst. However, in terms of efficiency, which is measured by time spent for reaching convergence, the curve of Alternating Steepest Decent Method requires least time to be stable, and the second fastest is Singular Value Projection Method, but Singular Value Thresholding Method needs much more time.

  The result is reasonable and consistent with result presented in [RPI09] and [JK15]. Because as a Convex Relaxation Method, SVT is always able to achieve global optimum, but it needs computing an increasing number of singular components during the iterations, which is time-consuming. However, for convex optimization type methods SVP and ASD, they may suffer

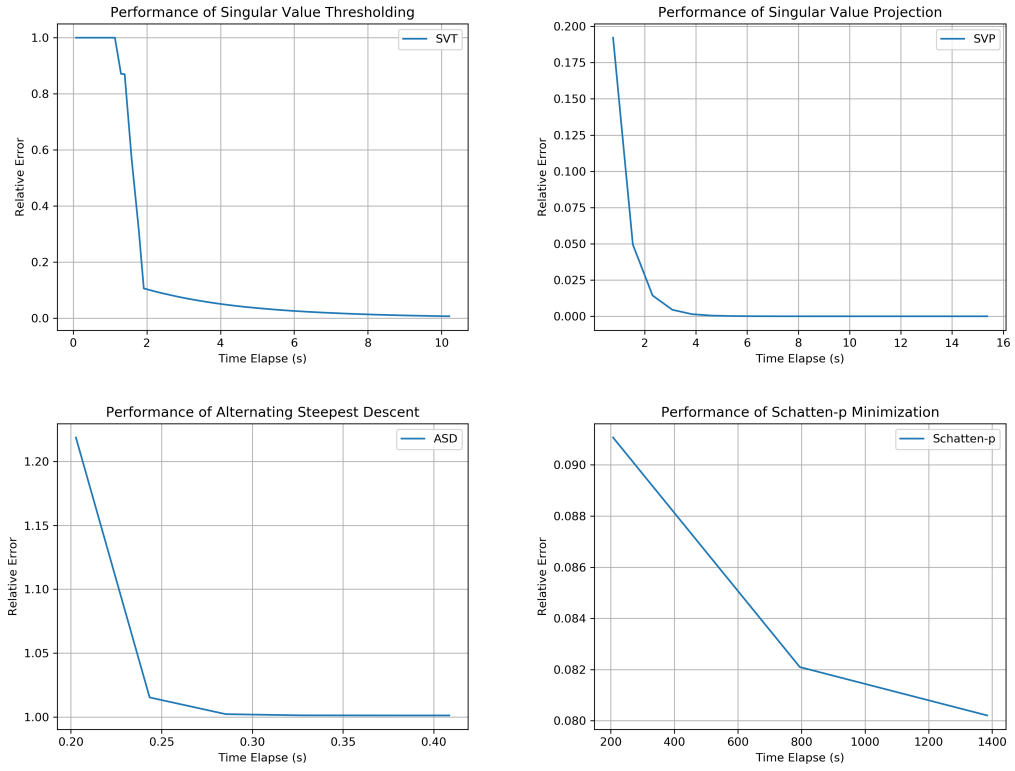

Figure 5.2: Performance of Four Algorithms on Large Synthetic Matrix

from local optima, that's why ASD has poor performance, but ASD does not perform SVD and only deal with small-size factor matrix in every iteration. Moreover, it has technique to simplify its algorithms in order to reduce complex operations, so ASD is extremely fast. For SVP, it is also able to achieve global optimum and it performs fixed number truncated SVD, which saves time. The reason why SVP outperforms SVT may be that SVT minimizes the convex envelope of rank i.e. Nuclear Norm, while SVP directly minimizes non-convex rank function. For Schatten-p Norm Minimization Method, it performs pseudo-inverse calculation, which is also of high computational complexity, so it is relatively slow.

5.3.2 Experiment 2: Performance on Large Synthetic Matrix

- **Result**

  In Experiment 2, four algorithms are performed on a 1000×1000 noiseless random matrix with rank 5 and sampling rate 0.5. The purpose of this experiment is illustrating the performance of four algorithms on different-size matrices. As a result, the result of Experiment 1 is considered as control group, and the result of Experiment 2 is compared with it. The Relative Error changes against Time is shown in Figure 5.2 and their final minimum Relative Error is shown in Table 5.2.

- **Discussion**

  When the matrix becomes larger, the relative status of these four algorithms does not change.

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
Relative Error	0.006831	$4.999629 \times e^{-10}$	1.001163	0.080207

Table 5.2: Optimal Relative Error Achieved by Four Algorithms

That means SVP still achieves the highest accuracy and ASD still has extremely fast speed but relative high Relative Error. In order to evaluate the influence of matrix size on algorithms performance, the result should be compared to Experiment 1's result, it can be concluded that when matrix gets bigger, all algorithms required more time to converge, especially Schatten-p Norm Minimization Method, because the calculation of pseudo-inverse for large matrix is much more expensive. However, the increased matrix size does not harm the accuracy of four algorithms, in other words, all these four algorithms have similar or slightly lower Relative Error than Experiment 1 and SVP even performs much better, which may due to the increased training time.

5.3.3 Experiment 3: Performance on High Rank Synthetic Matrix

- **Result**

  In Experiment 3, four algorithms are performed on a 100×100 noiseless random matrix with rank 100 and sampling rate 0.5. The purpose of this experiment is illustrating the performance of four algorithms on different-rank matrices. As a result, the result of Experiment 1 is considered as control group, and the result of Experiment 3 is compared with it. The Relative Error changes against Time is shown in Figure 5.3 and their final minimum Relative Error is shown in Table 5.3

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
Relative Error	0.913841	0.712304	4236.655722	0.778393

Table 5.3: Optimal Relative Error Achieved by Four Algorithms

- **Discussion**

  Compared to the result in Experiment 1, it can be seen that when the rank of matrix becomes higher, SVP still achieves the highest accuracy and ASD still has extremely fast speed but very high Relative Error. However, Schatten-p Norm Minimization Method fails in this situation. For SVT and SVP, albeit less time required for convergence than Experiment 1, the accuracy is worse than Experiment 1. Furthermore, for ASD, it even need more time to achieve a much worse result. It can be concluded that all four algorithms perform better in low rank matrix completion problem than high rank matrix completion task.

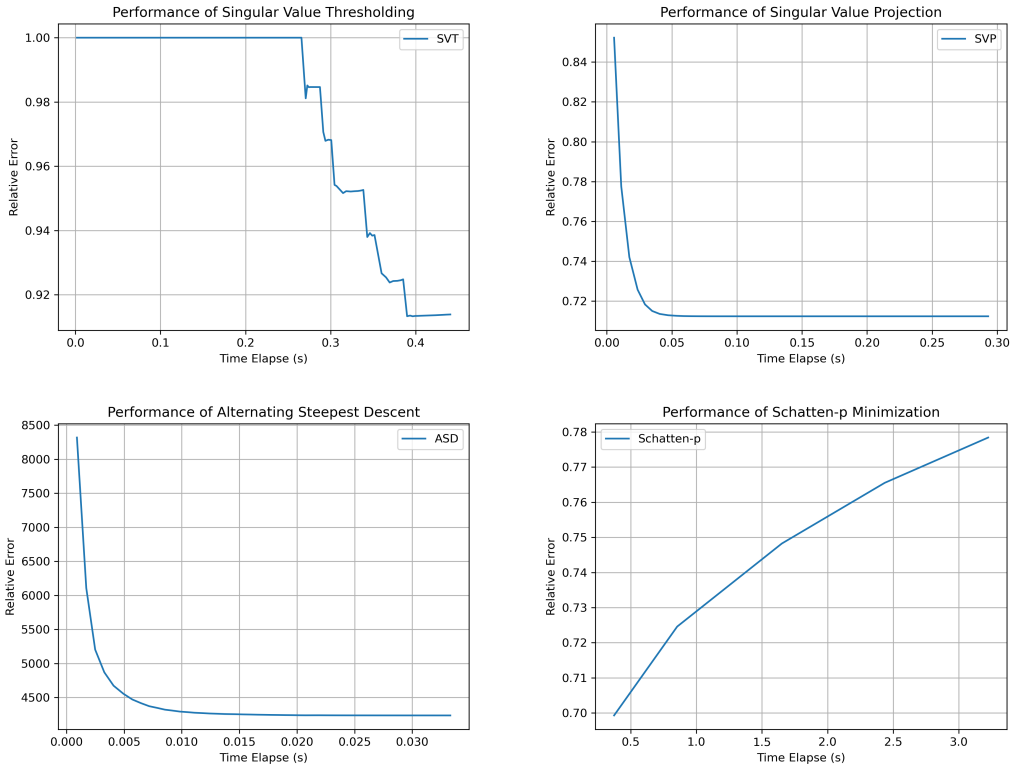

Figure 5.3: Performance of Four Algorithms on High Rank Synthetic Matrix

5.3.4 Experiment 4: Performance on High Sampling Rate Synthetic Matrix

- **Result**

  In Experiment 4, four algorithms are performed on a 100×100 matrix with rank 5 and sampling rate 0.8. The purpose of this experiment is illustrating the performance of four algorithms on different-sampling rate matrices. As a result, the result of Experiment 1 is considered as control group, and the result of Experiment 4 is compared with it. The Relative Error changes against Time is shown in Figure 5.4 and their final minimum Relative Error is shown in Table 5.4

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
Relative Error	0.002265	$1.832432 \times e^{-6}$	0.978610	0.165538

Table 5.4: Optimal Relative Error Achieved by Four Algorithms

- **Discussion**

  Compared to the result in Experiment 1, it can be seen that when there are more observed entries, all curves shift to the left which means they require less time to converge and they can achieve better accuracy. The reason behind this phenomenon is apparent, when there are more observed entries, the missing values to be estimated are less and the relation between entries is easier to infer. Similar to previous experiments the relative status of four algorithms remain stable, which means SVP still achieves the highest accuracy and ASD still has extremely fast speed but relative high Relative Error.

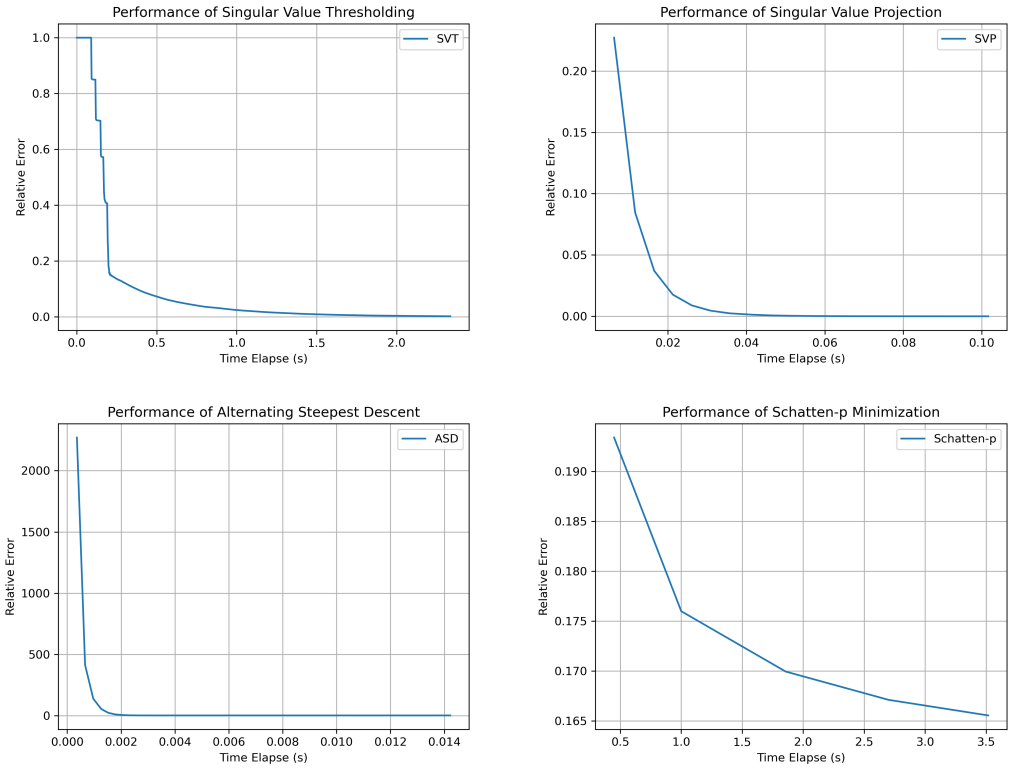

Figure 5.4: Performance of Four Algorithms on High Rank Synthetic Matrix

5.3.5 Experiment 5: Performance on Real-world Image Matrix

- **Result**

  In Experiment 5, four algorithms are applied on 512×512 Lena image matrix and 640×480 Yixiu Tower photo with rank 50 and sampling rate 0.5. The purpose of this experiment is illustrating the performance of four algorithms on real-world image recovery task, for both standard image and self-generated photo. The Peak Signal to Noise Ratio(PSNR) changes against Time for Lena image is shown in Figure 5.5 and their final maximal PSNR for Lena image is shown in Table 5.5. In order to view the result intuitively, the recovered Lena images are also shown in Figure 5.6. Similarly, the results about Yixiu Tower photo recovery are shown in Figure 5.7, Table 5.6 and Figure 5.8

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
PSNR	62.162982	298.695463	13.319062	20.255373

Table 5.5: Optimal PSNR Achieved by Four Algorithms For Lena Image

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
PSNR	39.667938	138.427914	14.144146	14.816584

Table 5.6: Optimal PSNR Achieved by Four Algorithms for Yixiu Tower Photo

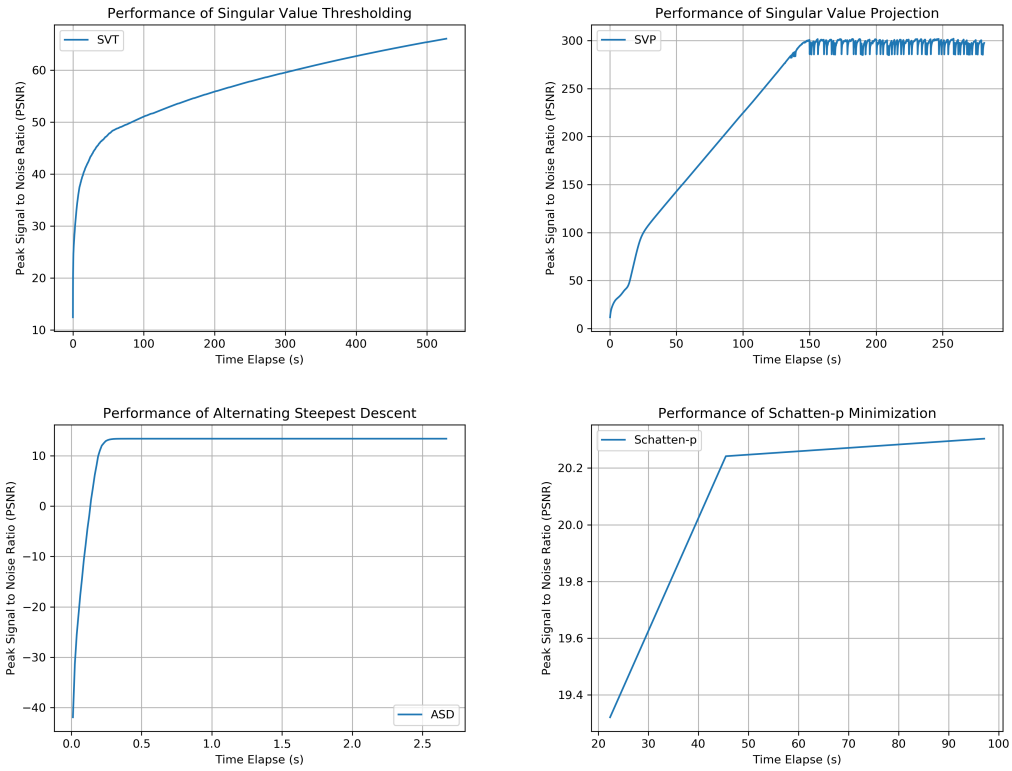

Figure 5.5: Performance of Four Algorithms on Lena Image Matrix

Figure 5.6: Recovered Lena Images of Four Algorithms

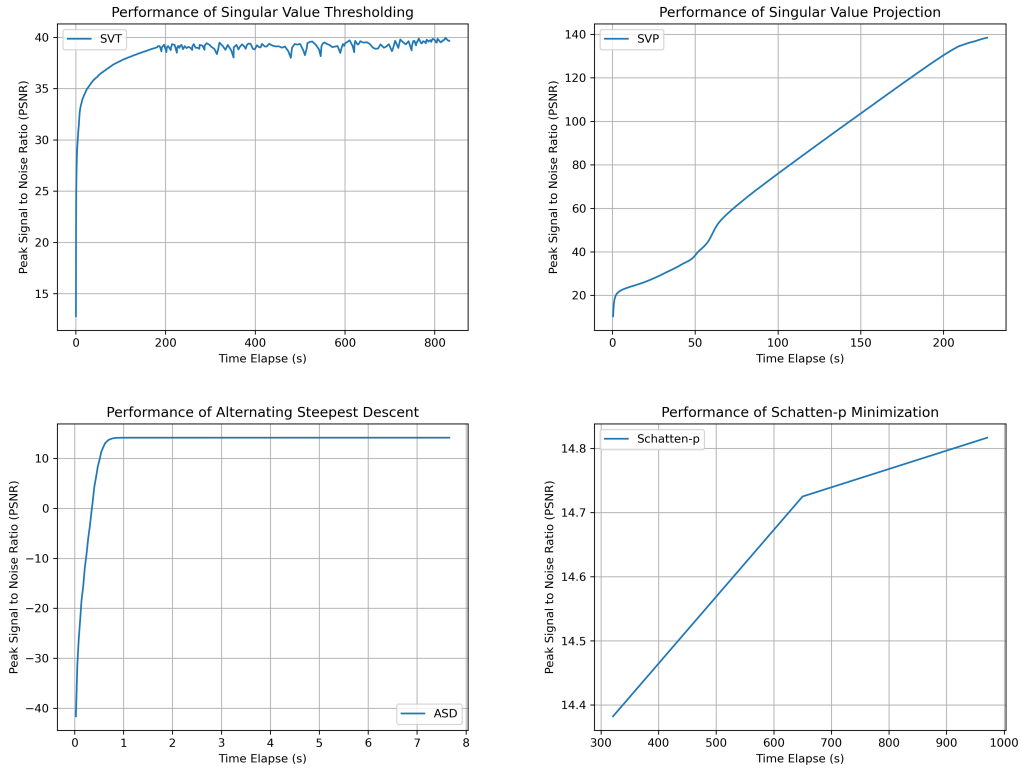

Figure 5.7: Performance of Four Algorithms on Yixiu Tower Photo Matrix

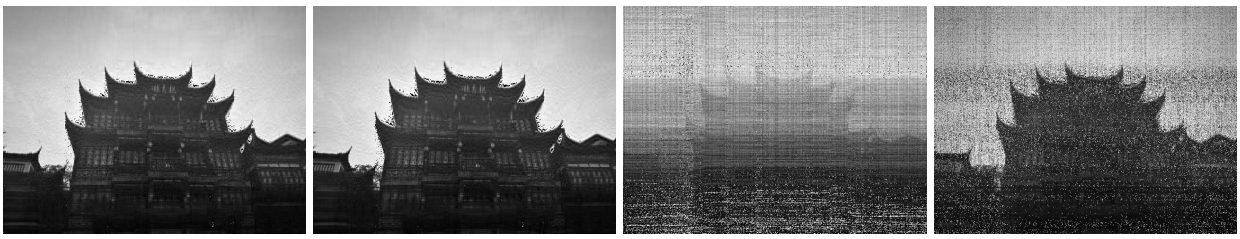

Figure 5.8: Recovered Yixiu Tower Photos of Four Algorithms

- **Discussion**

  When four algorithms applied to real-word image recovery task, It takes four algorithms much more time to converge for Lena image than Experiment 2, even though the Lena image matrix is indeed smaller than Experiment 2. The reason is that the pixel values are within the range of 0-255, which is wider than randomly-generated values in Experiment 2. Compared to Lena image, Yixiu Tower photo recovery takes more time, because of the larger matrix size. The photo also achieves lower PSNR except the result from ASD, which may caused by noise and blur when I took it. However, the relative status of these four algorithms is still unchanged for both Lena image and Yixiu Tower photo. That means SVP still achieves the highest accuracy and the recovered image is of high quality and ASD still has extremely fast speed but relative low PSNR. According to Figure 5.6, the result is more apparent. SVT and SVP Method both produce satisfying result, while ASD Method performs really bad, we can barely make out the girls outline. However, for the sake of fast speed, ASD's poor accuracy can be tolerated. Schatten-p Norm Minimization Method has moderate accuracy and running time.

5.3.6 Experiment 6: Performance on Noisy Image Matrix

- **Result**

  In Experiment 6, four algorithms are performed on a 512×512 Lena image and 640×480 Yixiu Tower photo with rank 50 and sampling rate 0.5. However, different from Experiment 5, Gaussian-distributed white noise is add to image matrix. The purpose of this experiment is illustrating the performance of four algorithms on noisy matrix, which indicates their robustness through noise. As a result, the result of Experiment 5 is considered as control group, and the result of Experiment 6 is compared with it. For Lena image, The Peak Signal to Noise Ratio(PSNR) changes against Time is shown in Figure 5.9 and their final maximal PSNR is shown in Table 5.7. In order to view the result intuitively, the recovered images are also shown in Figure 5.10. Similarly, the results about Yixiu Tower photo recovery are shown in Figure 5.11, Table 5.8 and Figure 5.12

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
PSNR	43.331384	51.078878	13.400392	19.948200

Table 5.7: Optimal PSNR Achieved by Four Algorithms for Lena Image Matrix

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
PSNR	36.872917	50.291370	14.108097	14.825491

Table 5.8: Optimal PSNR Achieved by Four Algorithms for Noisy Yixiu Tower Photo

- **Discussion**

  Results for Lena image and Yixiu Tower photo share similarities. Compared to the result in

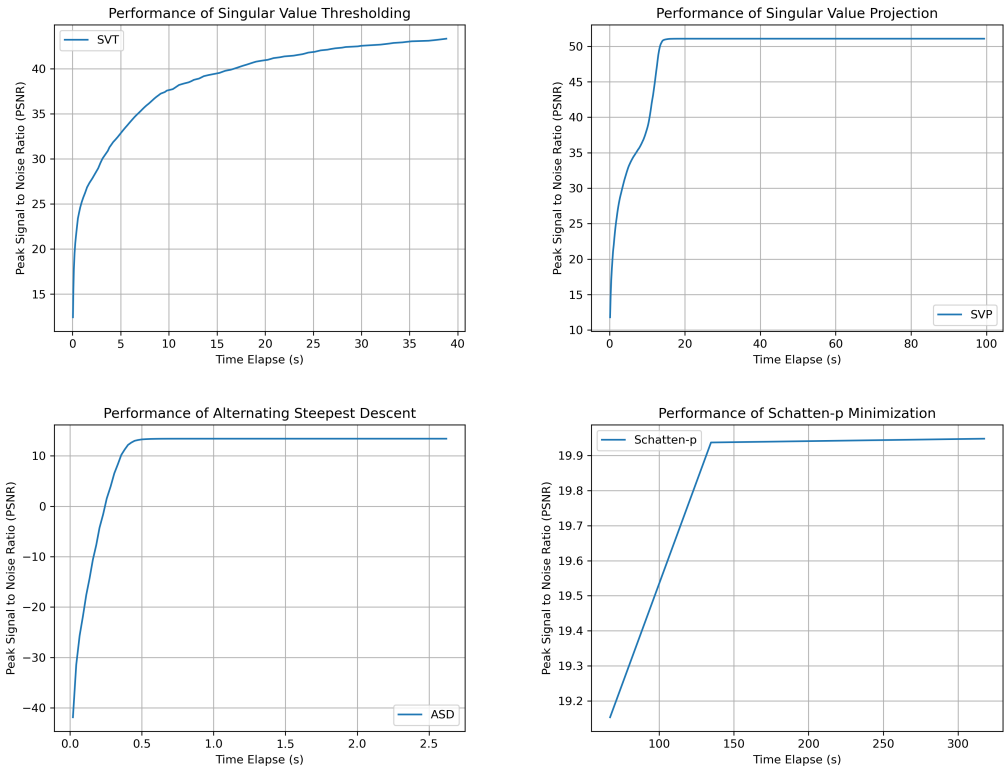

Figure 5.9: Performance of Four Algorithms on Noisy Lena Image Matrix

Figure 5.10: Recovered Noisy Lena Images of Four Algorithms

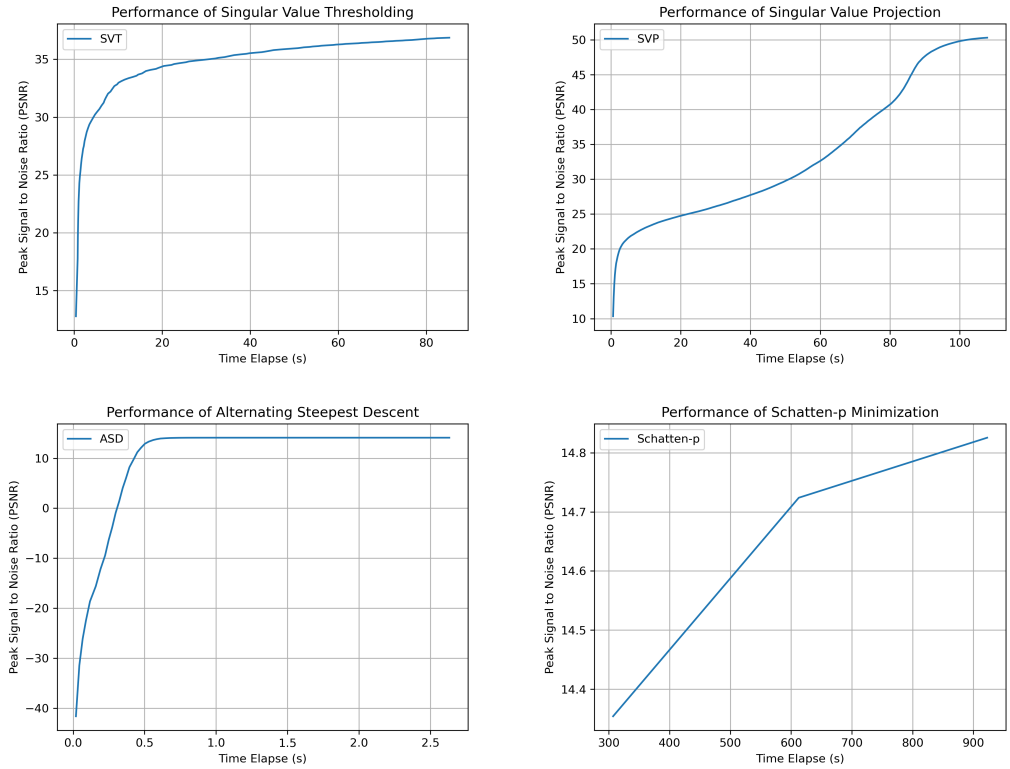

Figure 5.11: Performance of Four Algorithms on Noisy Yixiu Tower Photo Matrix

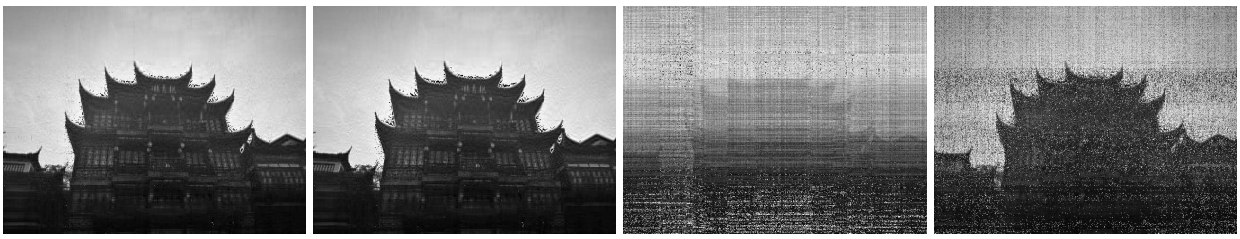

Figure 5.12: Recovered Noisy Yixiu Tower Photos of Four Algorithms

Experiment 5, it can be seen that when the noise is added to original image, all algorithms suffer from the noise and achieve lower PSNR. Among them, SVP is affected most and then is SVT Method. While due to the lower optimal PSNR, the time consumed is less for these two algorithms. The reason for SVT's poor result is that the convex relaxation algorithms always attempt to find the global optimum regardless of the presence of noise, which may in turn leads to bias to the model. Moreover, both SVT and SVP Method shrink the singular values to zero in each iterations. However, when the noise level is high, the real singular values will be too small after shrinkage to perform well in latter steps. To alleviate this situation, some postprocessing methods should follow the convex relaxation algorithm. For ASD and Schatten-p Norm Minimization Method, their accuracy are not affected by the noise a lot, but Schatten-p Norm Minimization Method requires more time to converge. As a result, ASD is the most robust algorithm through noise while SVP is the least robust one.

5.3.7 Experiment 7: Performance on Real-world Rating Matrix

- **Result**

  In Experiment 7, four algorithms are performed on a Movielens rating matrix. For simplicity, I only use the truncated 500×500 matrix and the rank is estimated to be 20 using the technique introduced in Section 4.3.1. The purpose of this experiment is illustrating the performance of four algorithms on real-world rating prediction task. As illustrated before, the main difficulties of this task are its sparsity, large matrix size and randomly-distributed missing values. The Normalized Mean Absolute Error(NMAE) changes against Time is shown in Figure 5.13 and their final minimal Normalized Mean Absolute Error(NMAE) is shown in Table 5.9.

Algorithm	SVT	SVP	ASD	Schatten-p Minimization
NMAE	0.001342	39.158673	3.432742	$2.299315 \times e^{-14}$

Table 5.9: Optimal NMAE Achieved by Four Algorithms

- **Discussion**

  When four algorithms applied to real-word rating prediction task, SVP and Schatten-p Minimization Method fail because the missing values in truncated rating matrix do not uniformly distributed, which does not meet the constraints for low rank matrix completion problem described in Section 2.3. For SVT and ASD Method, which survive from recommendation system task, their features are still obvious. SVT achieves relative higher accuracy but with slow convergence rate, while ASD has relative bad accuracy but extremely high speed. Compared to Experiment 3 image recovery task, despite the matrix size is similar, the rating prediction task consumes more time owing to the fact that the rating matrix has many randomly-distributed missing values.

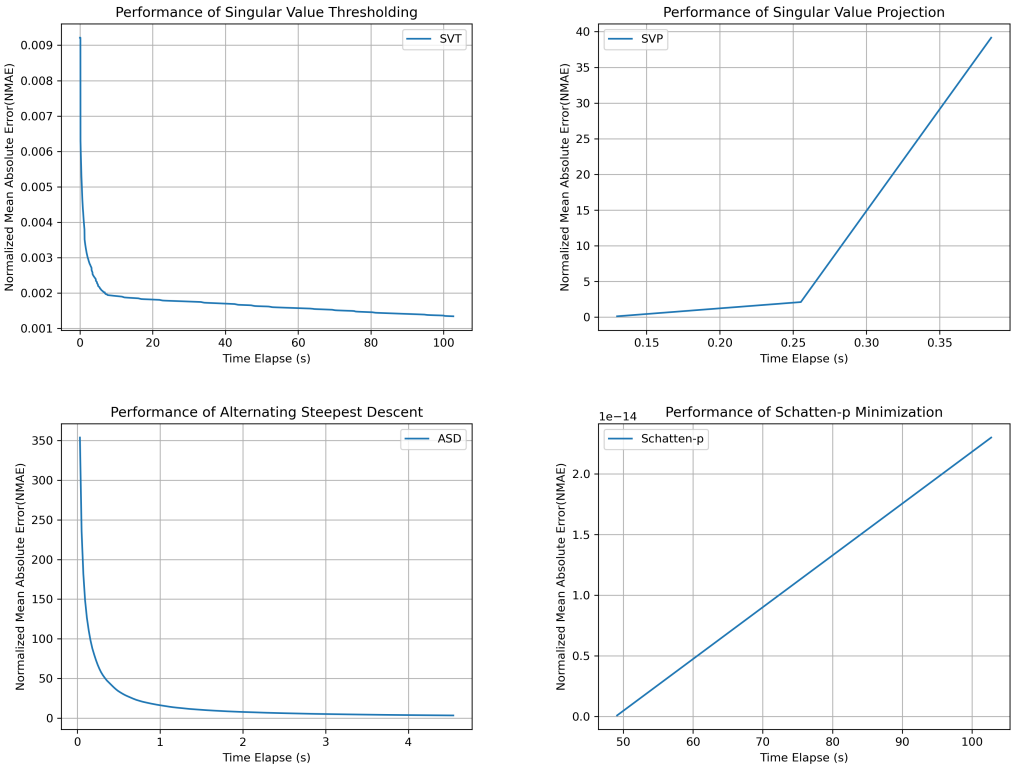

Figure 5.13: Performance of Four Algorithms on Movielens Rating Matrix

5.4 Experiment Summary

According the discussion about the result of 7 experiments, it is undoubted that each algorithm has its merits and drawbacks. The general findings and features of different types of low rank matrix can be concluded as follows.

- Small matrix size, low rank and high sampling rate facilitate all algorithms in terms of accuracy and running time. All algorithms suffer from noise more or less. The results of real-word matrix prove that small range of value and uniformly-distributed entries also contribute to the convergence.

- As a Convex Relaxation Method, Singular Value Thresholding Method is able to achieve global optimum with excellent accuracy. Moreover, it can handle various type matrix completion problem, regardless of the matrix size, rank, sampling rate. For real word image recovery and rating prediction task, it also has outstanding result.

- As a Direct Non-convex Optimization Method, Singular Value Projection is also able to reach global optimum, which always has the highest recovery accuracy. However it fails in rating prediction. So in order to use it, the missing value should be distributed uniformly and more than the minimal number illustrated in Section 2.3. Moreover, it is vulnerable to noise, so the original matrix should be free of noise to achieve better result.

- As a Direct Non-convex Optimization Method, Alternating Steepest Descent Method has outstanding convergence speed. It always consumes least time and storage in all experiments, which owes to the small size of factor matrix stored and processed in each iteration. Furthermore, it is the robustest algorithm among all these four algorithms. However, it always produces least accuracy, especially when the rank of matrix is high. So it is not suitable for high rank matrix completion problem.

- For Schatten-p Norm Minimization Method, it always generates moderate result, with medium accuracy and medium speed. However, because of the pseudo-inverse calculation, it consumes much more time to converge for large matrix. Moreover, it fails in ratings prediction tasks and high rank matrix completion. As a result, to use it, the property of uncompleted matrix should be examined carefully.

5.5 Guidelines for LRMC Algorithms Selection

In order to achieve optimal result in low rank matrix completion task, the algorithms should be selected wisely according to the tasks confronted. Based on the feature summarized above, the guidelines for algorithms selection are listed below.

- If you pursue fast convergence rate, accepting to sacrifice accuracy, Then Direct Non-convex Optimization Method with Alternating Minimization technique is a preferable choice. What's more, because of the independence of subroutines, it is suitable for parallel computing in reality, which further saves time. While the algorithms involve Singular Value Decomposition and pseudo-inverse calculation should be avoided.

- If you pursue high recover accuracy, both Direct Non-convex Optimization with Project Gradient Technique and Convex Relaxation Method can be selected, for they are able to reach global optimum. More specifically, in this project, the former has higher accuracy, while the latter can be applied to various types of matrices.

- For some LRMC algorithms, their application area is really narrow, so before use them, the property of original uncompleted matrix ought to be examined carefully. Schatten-p Norm Minimization Method is an example, which fails in rating prediction task and high rank matrix completion task.

- If the environment is vulnerable to noise, robustness is main criterion for algorithms selection. A robust algorithm remains stable performance even when the original matrix is corrupted by noise. In this project, Alternating Steepest Descent is one of robust algorithms.

- In terms of the algorithm implementation, the number of tunable parameter is also one of the considerations. Note that it is a double-edged sword, because although less parameters relieves the algorithm designer from troublesome parameter tuning job, it also leads to less control over the running process of the algorithm. In this project, Alternating Steepest Descent Method does not involve any hyper-parameters, which prevent the user to improve its relative poor accuracy.

5.6 Summary

Chapter 5 represents the results of 7 experiments carried out in Chapter 4. Their accuracy is measured by Relative Error for synthetic matrix, PSNR for image recovery task and NMAE for rating prediction task. Their efficiency is measured by time required for convergence. Robustness is presented by the degree of decline in accuracy caused by noise. After comparing their result, the pros and cons of each algorithm is summarized. Convex Relaxation Method SVT has relative high accuracy and wider application scenario but relative slow convergence rate. While Direct Non-convex Optimization Method ASD has extremely fast speed to converge and high robustness but relative low accuracy. SVP always achieves best accuracy but similar to Schatten-p Norm Minimization, it has narrower application area. Schatten-p Norm Minimization is relatively slow because of the calculation of pseudo-inverse. Based on their features, guidelines for algorithms selection under different circumstances are also provided in this chapter.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this essay, I attempt to present an overview of low rank matrix completion problem. First, the low rank matrix completion model is described and the constraints for this problem are illustrated. Second, some current low rank matrix completion methods are described including their main idea, pros and cons. Then based on the property of their objective function, I classify these methods into four categories: **Naive Method for Missing Values, Convex Relaxation Method, Direct Non-convex Optimization Method, Other Type LRMC Method**. The first type is not emphasized in this project. Furthermore, the Direct Non-convex Optimization Method can be divided by its optimization techniques, such as Projected Gradient Descent, Alternating Minimization, Manifold Optimization and Basis Pursuit. To capture the feature of methods in each type. Their representatives are selected to implement for latter comparison. Singular Value Thresholding Method is Convex Relaxation Method. Singular Value Projection Method and Alternating Steepest Descent Method are Direct Non-convex Optimization Method using Projected Gradient Descent technique and Alternating Minimization technique respectively. Schatten-p Norm Minimization is Other Type LRMC Method. Their performance is evaluated in terms of efficiency, accuracy and robustness. According to the experiment result, it can be concluded that the convex relaxation type algorithms are more stable, for they seek global minimum solution all the time, while non-convex methods using various techniques are more likely to be stuck in local minimum. However, the convex relaxation algorithms always required singular value decomposition, which is time-consuming especially when the matrix is large. To deal with it, efficient SVD packages should be used here. For Direct Non-convex Optimization Methods, in addition to local minimum problem, the majority of methods require rank information in advance, which require efficient and accurate rank estimation technique to solve real problem. According to the experiment result, Using project gradient technique for non-convex problem is able to achieve better result than Convex Relaxation Method but fails for high rank problem. While using Alternating Minimization technique is always fast, scalable and efficient. Moreover, for large matrix, it require less storage and resistant to noise. In reality, parallel programming can be used to accelerate the process. For Schatten-p Norm Minimization Method, it has moderate performance but the matrix should be low rank and the missing values should distribute uniformly. Finally, based on the features summarized,

Guidelines for algorithm selection for low rank matrix completion problem under various circumstances are provided. The selection should be consistent with the objective and refer to the efficiency, accuracy, robustness and the number of tunable parameters of algorithms to be selected.

6.2 Limitations and Future Work

Because of the restricted time and equipment, there are many topics which are not covered in this essay, some of them are listed below to provide future research direction.

- When carrying out literature review, it can be found that he constraints for Convex Relaxation Method to find proper convex replacement and for Direct Non-convex Optimization Method to overcome NP-hardness are vert similar in coincidence. The reasons for this phenomenon should be illustrated in the future.
- This project mainly describes the general low rank matrix completion method, which can be used to solve various types of matrices. However, to achieve better result, some algorithms tailored for different tasks are also worth exploration. For example, in image recovery task and recommendation system, all values in the matrix are non-negative. If this property is used, that will facilitate the completion algorithm.
- Numerous methods require rank information as a prior. Although some rank estimation methods are given in this essay, this problem is always a tough topic in low rank matrix completion area, which need further investigation.
- For many algorithms, the main burden on computational complexity is the calculation of Singular Value Decomposition and pseudo-inverse. So efficient techniques such as partial SVD should be explored and adopted in the algorithms.
- When finding the optimal rank used for Alternating Minimization Method, this project adopts the method described in the essay that try different rank in increasing order and find the optimal. However, we find that the different rank values only influence the initial error but achieve the same final error. The reasons deserve exploration to save LRMC problem from accurate rank estimation constraint.
- This project focuses on low rank 2D matrix completion. However as high dimensional data increases nowadays, low rank tensor completion is also a heated topic, which can be analysed in the future.
- Although this project evaluate the influence of matrix size on the performance of different LRMC algorithms, the matrix is not large enough. Because in rating prediction task, the matrix is easily larger than 1000×1000 dimension. With better equipment in the future, much larger matrix should be tested.

Bibliography

[AHU63] Kenneth Arrow, L. Hurwicz, and H Uzawa. Studies in linear and nonlinear programing. *Econometrica*, 31, 07 1963.

[AMQ+18] Ramlatchan Andy, Yang Mengyun, Liu Quan, Li Min, Wang Jianxin, and Li Yaohang. A survey of matrix completion methods for recommendation systems. *Big Data Mining and Analytics*, 1:308–323, 12 2018.

[ATM06] Argyriou Andreas, Evgeniou Theodoros, and Pontil Massimiliano. Multi-task feature learning. pages 41–48, 01 2006.

[Bar13] Vandereycken Bart. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 23:10.1137/110845768, 04 2013.

[BD09] Thomas Blumensath and Mike Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27:265–274, 11 2009.

[CBSX12] Chen Caihua, He Bing-Sheng, and Yuan Xiaoming. Matrix completion via alternating direction method. *IMA Journal of Numerical Analysis*, 32, 01 2012.

[CC18] Y. Chen and Y. Chi. Harnessing structures in big data via guaranteed low-rank matrix estimation. *Signal Process. Mag.*, 35(4):14–31, Jul. 2018.

[C.R10] C.R.Berger. Double exponential. *IEEE Trans. Signal Process.*, 56(5):1708–1721, 2010.

[CSLT12] Emmanuel Candes, Carlos Sing Long, and Joshua Trzasko. Unbiased risk estimates for singular value thresholding and spectral estimators. *IEEE Transactions on Signal Processing*, 61, 10 2012.

[CT92] Tomasi Carlo and Kanade Takeo. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9:137–54, 11 1992.

[CT09] E.J. Candès and T. Tao. The power of convex relaxation. *Near-optimal matrix completion: IEEE Transactions on Information Theory*, 56:2053–2080, 01 2009.

[Dhr20] Trehan Dhruv. Non-convex optimization: A review. pages 418–423, 05 2020.

[DR16] M. A. Davenport and J. Romberg. An overview of low-rank matrix recovery from incomplete observations. *Topics Signal Process.*, 10(4):608–622, June 2016.

[EB08] Candès Emmanuel and Recht Benjamin. Exact matrix completion via convex optimization. *Communications of the ACM*, 9:717–772, 11 2008.

[FHC12] Nie Feiping, Huang H., and Ding Chenwei. Low-rank matrix recovery via efficient schatten p-norm minimization. *Proceedings of the National Conference on Artificial Intelligence*, 1:655–661, 01 2012.

[JBY+15] Choi Jun, Shim Byonghyo, Ding Yacong, Rao Bhaskar, and Kim Dong-In. Compressed sensing for wireless communications : A few tips and tricks. *IEEE Communications Surveys Tutorials*, PP, 11 2015.

[JFEZ10] Cai Jian-Feng, Candès Emmanuel, and Shen Zuowei. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20:1956–1982, 03 2010.

[JK15] Tanner Jared and Wei Ke. Low rank matrix completion by alternating steepest descent methods. *Applied and Computational Harmonic Analysis*, 40, 08 2015.

[JKB11] Wang Jian, KwonSeokbeop, and Shim Byonghyo. Generalized orthogonal matching pursuit. *IEEE Transactions on Signal Processing*, 60, 11 2011.

[KM12] Mohan Karthik and Fazel Maryam. Iterative reweighted algorithms for matrix rank minimization. *The Journal of Machine Learning Research*, 13:3441–3473, 11 2012.

[KMO10] R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980–2998, 2010.

[KO10] Amin Karbasi and Sewoong Oh. Distributed sensor network localization from local connectivity. *ACM SIGMETRICS Performance Evaluation Review*, 38:61, 06 2010.

[L.07] Eldén L. Matrix methods in data mining and pattern recognition. 01 2007.

[len] lena image download.

[LJB19] Nguyen Luong, Kim Junhan, and Shim Byonghyo. Low-rank matrix completion: A contemporary survey. *IEEE Access*, PP:1–1, 07 2019.

[MG97] Mesbahi Mehran and Papavassilopoulos G.P. On the rank minimization problem over a positive semidefinite linear matrix inequality. *Automatic Control, IEEE Transactions on*, 42:239 – 243, 03 1997.

[MHR10] Fornasier Massimo, Rauhut Holger, and Ward Rachel. Low-rank matrix recovery via iteratively reweighted least squares minimization. *SIAM Journal on Optimization*, 21, 10 2010.

[MHS01] Fazel Maryam, Hindi Haitham, and Boyd S.P. A rank minimization heuristic with application to minimum order system approximation. volume 6, pages 4734 – 4739 vol.6, 02 2001.

[mov] movielens dataset download.

[NEY01] Linial Nathan, London E, and Rabinovich Yuri. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–246, 03 2001.

[NS12] T.T. Ngo and Yousef Saad. Scaled gradients on grassmann manifolds for matrix completion. *Advances in Neural Information Processing Systems*, 2:1412–1420, 01 2012.

[PD04] Chen Peiji and Suter David. Recovering the missing components in a large noisy low-rank matrix: Application to sfm. *IEEE transactions on pattern analysis and machine intelligence*, 26:1051–63, 09 2004.

[PP17] Jain Prateek and Kar Purushottam. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10:142–336, 12 2017.

[PPS12] Jain Prateek, Netrapalli Praneeth, and Sanghavi Sujay. Low-rank matrix completion using alternating minimization. *Proceedings of the Annual ACM Symposium on Theory of Computing*, 12 2012.

[RPCI08] Meka Raghu, Jain Prateek, Caramanis Constantine, and Dhillon Inderjit. Rank minimization via online learning. pages 656–663, 01 2008.

[RPI09] Meka Raghu, Jain Prateek, and Dhillon Inderjit. Guaranteed rank minimization via singular value projection. *NIPS*, 09 2009.

[RS09] Keshavan Raghunandan and Oh Sewoong. A gradient descent algorithm on the grassman manifold for matrix completion. 910, 10 2009.

[SDL09] Ma Shiqian, Goldfarb Donald, and Chen Lifeng. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128, 05 2009.

[SJ98] Sturm and Jos. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11, 11 1998.

[SJ03] Nathan Srebro and Tommi Jaakkola. Weighted low-rank approximations. 2, 08 2003.

[SKJB13] Suhyuk, Kwon, Wang Jian, and Shim Byonghyo. Multipath matching pursuit. *Information Theory, IEEE Transactions on*, 60, 08 2013.

[SRJ04] Nathan Srebro, Jason Rennie, and Tommi Jaakkola. Maximum-margin matrix factorization. volume 17, 11 2004.

[TKCM99] Toh, Kim-Chuan, and Todd Michael. Sdpt3 – a matlab software package for semidefinite programming. *Optimizn Meth. Softwr.*, 11, 01 1999.

[WOE11] Dai Wei, Milenkovic Olgica, and Kerman Ely. Subspace evolution and transfer (set) for low-rank matrix completion. *Signal Processing, IEEE Transactions on*, 59:3120 – 3132, 08 2011.

[XCHW14] Zhou Xiaowei, Yang Can, Zhao Hongyu, and Yu Weichuan. Low-rank modeling and its applications in image analysis. *ACM Computing Surveys*, 47, 01 2014.

[YDJ+13] Hu Yao, Zhang Debing, Ye Jieping, Li Xuelong, and He Xiaofei. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE transactions on pattern analysis and machine intelligence*, 35:2117–30, 09 2013.

[YDRR08] Zhou Yunhong, Wilkinson Dennis, Schreiber Robert, and Pan Rong. Large-scale parallel collaborative filtering for the netflix prize. pages 337–348, 06 2008.

[Yeh09] Koren Yehuda. The bellkor solution to the netflix grand prize. 09 2009.

[YMNS07] Amit Yonatan, Fink Michael, Srebro Nathan, and Ullman Shimon. Uncovering shared structures in multiclass classification. volume 227, pages 17–24, 01 2007.

[YRC09] Koren Yehuda, Bell Robert, and Volinsky Chris. Matrix factorization techniques for recommender systems. *Computer*, 42:30–37, 08 2009.

[Zhi15] Xu Zhiqiang. The minimal measurement number for low-rank matrix recovery. *Applied and Computational Harmonic Analysis*, 05 2015.

[ZL09] Liu Zhang and Vandenberghe Lieven. Interior-point method for nuclear norm approximation with application to system identification. *SIAM J. Matrix Analysis Applications*, 31:1235–1256, 01 2009.

[ZMJZJ14] Wang Zheng, Lai Ming-Jun, Lu Zhaosong, and Ye Jieping. Orthogonal rank-one matrix pursuit for low rank matrix completion. *SIAM Journal on Scientific Computing*, 37, 04 2014.

[ZWY12] Wen Zaiwen, Yin Wotao, and Zhang Yin. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4, 12 2012.

Matrix Completion: Infer Missing Entries

Wang Tingyue

August 28, 2020

Appendix A

Example of operation

An appendix is just like any other chapter, except that it comes after the appendix command in the master file.

One use of an appendix is to include an example of input to the system and the corresponding output.

One way to do this is to include, unformatted, an existing input file. You can do this using `\verbatiminput`. In this appendix we include a copy of the C file hello.c and its output file hello.out. If you use this facility you should make sure that the file which you input does not contain `TAB` characters, since LaTeX treats each `TAB` as a single space; you can use the Unix command `expand` (see manual page) to expand tabs into the appropriate number of spaces.

A.1 Example input and output

A.1.1 Input

(Actually, this isn't input, it's the source code, but it will do as an example)

```
/* Hello world program */

#include <stdio.h>

int main(void)
{
    printf("Hello World!\n") ;
    return 0 ;
}
```

A.1.2 Output

A.1.3 Another way to include code

You can also use the capabilities of the `listings` package to include sections of code, it does some keyword highlighting.

```
/* Hello world program */

#include <stdio.h>

int main(void)
{
    printf("Hello World!\n");
    return 0;
}
```