Ph.D. DISSERTATION

Robust Subspace Learning and Clustering: Sparse and Low-Rank Representations

강인한 저차원 공간의 학습과 분류: 희소 및 저계수 표현

BY

EUNWOO KIM

FEBRUARY 2017

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

COLLEGE OF ENGINEERING

SEOUL NATIONAL UNIVERSITY

Robust Subspace Learning and Clustering:

Sparse and Low-Rank Representations

강인한 저차원 공간의 학습과 분류: 희소 및 저계수 표현

지도교수 오 성 회

이 논문을 공학박사 학위논문으로 제출함

2016 년 12 월

서울대학교 대학원

전기컴퓨터 공학부

김 은 우

김은우의 공학박사 학위논문을 인준함

2016 년 12 월

위 원 장	최 진 영
부 위원장	오 성 회
위 원	조 남 익
위 원	곽 노 준
위 원	이 민 식

Abstract

Learning a subspace structure based on sparse or low-rank representation has gained much attention and has been widely used over the past decade in machine learning, signal processing, computer vision, and robotic literatures to model a wide range of natural phenomena. Sparse representation is a powerful tool for high-dimensional data such as images, where the goal is to represent or compress the cumbersome data using a few representative samples. Low-rank representation is a generalization of the sparse representation in 2D space. Behind the successful outcomes, many efforts have been made for learning sparse or low-rank representation efficiently. However, they are still inefficient for complex data structures and lack robustness under the existence of various noises including outliers and missing data, because many existing algorithms relax the ideal optimization problem to a tractable one without considering computational and memory complexities. Thus, it is important to use a good representation algorithm which is efficiently solvable and robust against unwanted corruptions. In this dissertation, our main goal is to learn algorithms with both robustness and efficiency under noisy environments.

As for sparse representation, most of the optimization problems are relaxed to convex ones based on surrogate measures, such as the l_1-norm, to resolve the computational intractability and high noise sensitivity of the original sparse representation problem based on the l_0-norm. However, if the system at interest, other than the sparsity measure, is inherently nonconvex, then using a convex sparsity measure may not be the best choice for the problems. From this perspective, we propose desirable criteria to be a good nonconvex sparsity measure and suggest a corresponding family of measure. The proposed family of measures allows a simple measure, which enables efficient computation and embraces the

benefits of both l_0- and l_1-norms, and most importantly, its gradient vanishes slowly unlike the l_0-norm, which is suitable from an optimization perspective.

For low-rank representation, we first present an efficient l_1-norm based low-rank matrix approximation algorithm using the proposed alternating rectified gradient methods to solve an l_1-norm minimization problem, since conventional algorithms are very slow to solve the l_1-norm based alternating minimization problem. The proposed methods try to find an optimal direction with a proper constraint which limits the search domain to avoid the difficulty that arises from the ambiguity in representing the two optimization variables. It is extended to an algorithm with an explicit smoothness regularizer and an orthogonality constraint for better efficiency and solve it under the augmented Lagrangian framework.

To give more stable solution with flexible rank estimation in the presence of heavy corruptions, we present a new solution based on the elastic-net regularization of singular values, which allows a faster algorithm than existing rank minimization methods without any heavy operations and is more stable than the state-of-the-art low-rank approximation algorithms due to its strong convexity. As a result, the proposed method leads to a holistic approach which enables both rank minimization and bilinear factorization. Moreover, as an extension to the previous methods performing on an unstructured matrix, we apply recent advances in rank minimization to a structured matrix for robust kernel subspace estimation under noisy scenarios.

Lastly, but not least, we extend a low-rank approximation problem, which assumes a single subspace, to a problem which lies in a union of multiple subspaces, which is closely related to subspace clustering. While many recent studies are based on sparse or low-rank representation, the grouping effect among similar samples has not been often considered with the sparse or low-rank representa-

tion. Thus, we propose a robust group subspace clustering algorithms based on sparse and low-rank representation with explicit subspace grouping. To resolve the fundamental issue on computational complexity of existing subspace clustering algorithms, we suggest a full scalable low-rank subspace clustering approach, which achieves linear complexity in the number of samples.

Extensive experimental results on various applications, including computer vision and robotics, using benchmark and real-world data sets verify that our suggested solutions to the existing issues on sparse and low-rank representations are considerably robust, effective, and practically applicable.

Keywords: Sparse representation, low-rank representation, subspace learning, low-rank matrix factorization, subspace clustering, computer vision

To my family

Without whom none of my success would be possible

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over the past few years, we are facing a deluge of high-dimensional data, such as images, videos, and texts, from recent advances in digital technology. While the high quality data have improved the quality of life, handling or processing such massive data is a daunting and time-consuming task, since the advancement of processing power of a computing device does not follow the geometric growth of the amount of data. The term ”big data” emerges recently from this perspective (see Figure 1.1[1] for more details) and obviously it is difficult to address the huge data by conventional processing tools. Therefore, many researchers are continuously searching for a method to handle such data efficiently without losing critical information in the data. To this end, a number of algorithms using the concept of sparsity and low-rank-ness have been proposed to model the data efficiently in the presence of naturally occurring noises [1, 2, 3, 4].

A fundamental approach using the concept of parsimony is sparse representation [3, 2, 5]. The basic task of the sparse representation is to select informative

[1]Source: Thomson Reuters, http://blog.thomsonreuters.com/index.php/big-data-graphic-of-the-day

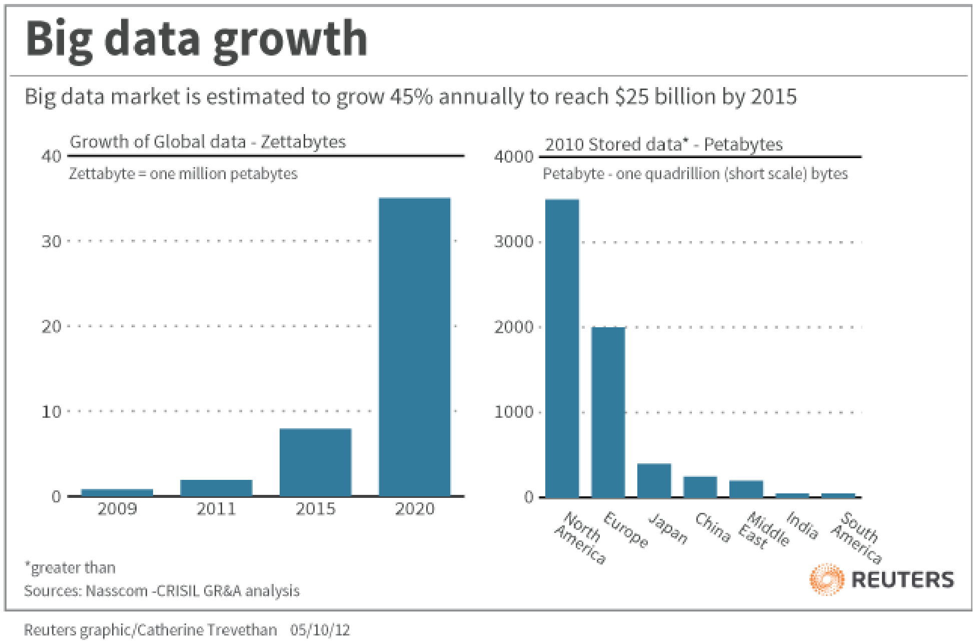


Figure 1.1: Graphical illustration of growth of data and the amount of stored data in the hottest industries in the world.

words in an overcomplete dictionary to fit target data. It is based on the l_0-norm and many algorithms proposed recently use the convex relaxation of the l_0-norm, i.e., l_1-norm, to learn a sparse coefficient vector. The sparse representation can be applied to various problems such as image denoising [3], dictionary learning [6], face recognition [5], and image super-resolution [7], etc.

An extension to the sparse representation to a 2D space is low-rank representation which is also known as low-rank matrix approximation. This approach is motivated by the fact that high-dimensional data can be well represented with a fewer number of basis factors in practice (see Figure 1.2 [8]). For example, in computer vision, most of the structure-from-motion methods are based on a fixed low-rank problem and background subtraction with a static camera can be

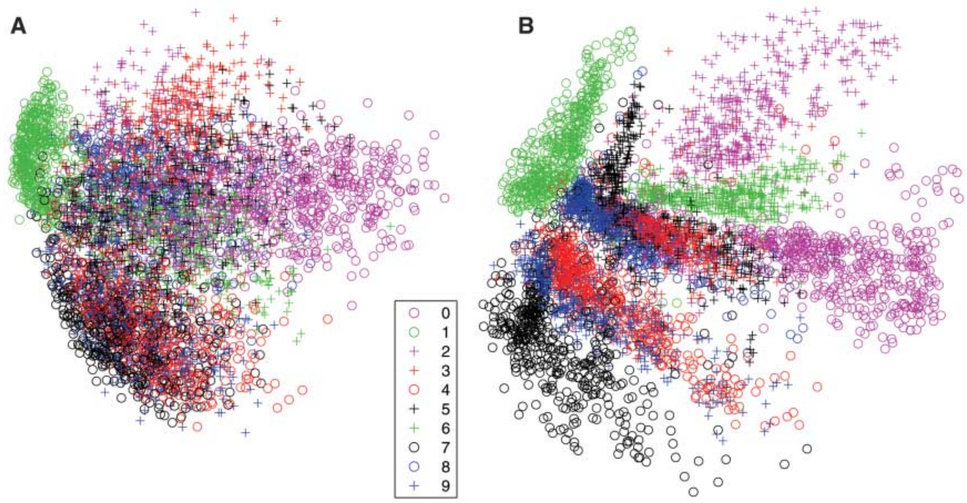
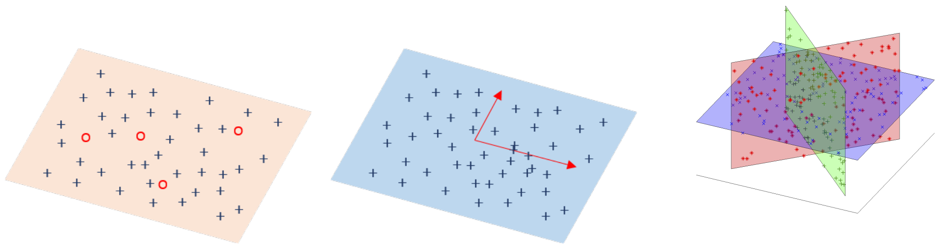


Figure 1.2: Visualization of the MNIST dataset. (A) The two dimensional codes by taking the first two principal components extracted from PCA. (B) The two dimensional codes found by low-dimensional learning using an autoencoder [8].

solved easily by a rank-1 problem with clean data or a rank-2 problem in the presence of corruptions. The most popular algorithm to reduce the dimension of data in high-dimensional space is the principal component analysis (PCA) [9] and its variants for modeling the low-dimensional structures have been proposed for a number of problems, such as data reconstruction [10], image denoising [11], collaborative filtering [1], background modeling [12], structure from motion [13], and photometric stereo [14], to name a few.

As a generalization of the low-rank approximation, which learns basis vectors to construct a single subspace, we can consider data which lie in a union of multiple subspaces. Finding the subspace structures of a complex space is closely related to subspace clustering [15, 16, 4], which identifies subspace membership of each data sample, where unknown multiple subspaces exist, by assuming that data are self-expressive, i.e., a data point can be represented by linear combination



(a) Sparse representation (b) Low-rank representation (c) Subspace clustering [18]

Figure 1.3: Graphical illustration of three subspace representation methods addressed in this thesis.

of other points in the same subspace. Subspace clustering has been successfully applied to a number of clustering problems, such as motion segmentation [15], face clustering [4], and image segmentation [17]. Figure 1.3 illustrates data structures of the three main problems addressed in the thesis.

1.1 Main Challenges

Behind the successful application of sparse and low-rank representation, there are still challenges for the existing algorithms. In this section, we consider two main challenges, robustness and efficiency, for three main tasks, sparse representation [3, 2, 6], low-rank representation [1, 10, 19], and subspace clustering [15, 16, 4], presented in this dissertation, which are summarized as follows:

- **Efficient sparsity measure for inherently nonconvex problems.** For sparse representation, many algorithms use the relaxation of the original sparse representation problem based on the l_0-norm, i.e., the l_1-norm, since it is computationally tractable and easy to guarantee applied algorithms.

Although the relaxed approach successfully applied to many problems with promising results, it is beneficial only when the relaxed problem indeed becomes convex. To tell the truth, there are few sparse representation approaches to consider inherently nonconvex problems. Obviously, there are many nonconvex problems we are faced with, such as matrix factorization [20, 21], rank-constrained optimization [22, 19], and sparse coding jointly optimized with dictionary learning [3, 2, 6]. Even though there are several algorithms performed in a greedy manner to directly solve the l_0-norm [23, 24], they can fail to find a reasonable solution according to the quality of a dictionary. There can be also a computational issue when the size of a dictionary is large.

- **Robustness and computational efficiency for low-rank representation.** Conventional low-rank approximation algorithms based on the l_2-norm is sensitive to outliers and missing entries, because the l_2-norm can sometimes amplify the negative effects of such data. This prevents recognition or machine learning systems from performing well. As an alternative, many studies based on a robust function such as the l_1-norm have been conducted [10, 25] to overcome the weakness of the conventional algorithms by assuming a non-Gaussian noise model. While the algorithms give robust solutions in the presence of outliers, they are too computationally intensive, making them not applicable in practice.

  Recently, robust PCA (RPCA) methods have been emerged to solve the non-Gaussian noise model based on the rank minimization strategy. While the rank minimization methods have been utilized in many problems, they still take heavy computational complexity due to the minimization strategy of a relaxed version of the rank function, and even they are not suitable for fixed-rank problems posed in computer vision literature. In summary, there is no clear winner for the low-rank representation problem satisfying both robustness and computational efficiency under the existence of unwanted corruptions.

- **Robust representation and scalability for subspace clustering.** Recent subspace clustering algorithms [15, 16, 4] consider both noise model and outlier model by switching the loss function in the formulation, but they can only guarantee the correct recovery of a block diagonal structure of subspaces only for clean data. Indeed, it is difficult to show the correctness of the algorithms under noisy scenarios. Furthermore, even though notable results have been reported for existing algorithms, they are still insufficient for achieving high clustering performance because of weak connections among similar samples.

  Another weakness of subspace clustering is heavy computational complexities as in the previous problems. Most of the state-of-the-art algorithms using sparsity or low-rank-ness take at least cubic complexity, which is practically unfavorable. There is additional factor to consider when we obtain an affinity matrix from an optimization: post-processing and spectral clustering steps whose time complexities are also significantly high (in general, over cubic complexity).

1.2 Organization of the Dissertation

Chapter 2 introduces related works in this dissertation. As a simple vector case of parsimonious modeling of data, we discuss sparse representation algorithms which are based on two main family; greedy pursuit and basis pursuit algorithms.

Then, we further discuss the 2D extension of the sparse representation, low-rank representation, and two important problems; fixed-rank representation (or low-rank matrix approximation) and automatic rank minimization (called robust principal component analysis). To consider general scenarios where data come from a union of multiple subspaces, we introduce the subspace clustering task and its two popular methods; sparse subspace clustering and low-rank representation. We also summarize the Gaussian process regression (GPR) which is used to model complex behavior of moving objects or pedestrians, where low-rank structured matrix approximation is considered in GPR for robustness.

In Chapter 3, we present a new sparsity measure, termed slowly vanishing gradient (SVG), for sparse representation in general nonconvex problems. We first suggest that the difficulty of handling the l_0-norm does not only come from the nonconvexity but also from its gradient either being zero (for the most parts) or not being well-defined. Accordingly, we analyze the space of approximate functions for the l_0-norm and the proposed measure, SVG. Locally, it follows the l_1-norm to reduce the chance of numerous local optima without losing the ability of promoting parsimony. Globally, SVG follows the l_0-norm to reduce penalty on large-values, but it still possesses slowly vanishing gradients to help drawing the solution of an optimization algorithm to sparse points. Moreover, we present an efficient proximity operator for the measure. The proposed measure is applied to various applications to demonstrate its adequacy. Experimental results confirm that our proposal performs favorably against those of state-of-the-art algorithms.

Chapter 4 describes several low-rank representation algorithms. We first propose a low-rank matrix approximation method based on the l_1-norm using the proposed alternating rectified gradient approach (l_1-ARG), which finds optimal directions for faster convergence compared to existing algorithms. Then, we introduce an efficient Frobenius-norm regularizer to prevent the overfitting problem which can arise from an alternative minimization algorithm and an orthogonality constraint to reduce the solution space for further speed-up. The new approach, called robust orthogonal matrix factorization (ROMF), is constructed under the augmented Lagrangian framework. It is also extended to handle the rank uncertainty issue by a rank estimation strategy for practical real-world problems. As an extension to the low-rank representation, we present a robust kernel subspace learning method based on recent advances in rank minimization in GPR to model trajectories of pedestrians or moving objects.

In Chapter 5, we develop a robust and stable algorithm with rank estimation for finding subspace structures of grossly corrupted data by proposing elastic-net subspace representation based on elastic-net regularization of singular values of data (FactEN). FactEN is a holistic approach which utilizes both nuclear-norm minimization and bilinear factorization. The strong convexity of the proposed regularizer alleviates the instability problem by shrinking and correcting inaccurate singular values in the presence of unwanted noises. We demonstrate the performance of the proposed methods in terms of the reconstruction error and computational speed using well-known benchmark datasets including non-rigid motion estimation, photometric stereo, and background modeling. Furthermore, in order to address data which lie in a union of multiple subspaces, we extend FactEN to a joint optimization algorithm which updates the data matrix corrupted by noises and subspace representation matrix or affinity matrix based on the noise-reduced data matrix by FactEN. Since we reduce unfavorable noises from the low-rank representation task, we simply adapt the sparse subspace segmentation task in the joint optimization framework.

In Chapter 6 and 7, we discuss algorithms on a subspace clustering task where data lie in a complex space composed of more than two different subspaces. Similar to the previous problems, we first consider robustness of subspace clustering. To this end, we consider grouping capability of the algorithms since the grouping effect among similar samples is very important when constructing an affinity matrix but it has not been often considered with sparse or low-rank representation. Hence, we propose two robust group subspace representation algorithms by extending sparse and low-rank representation with explicit subspace grouping. We show that the proposed methods capture the similarities among data samples collected from the same subspace, theoretically and empirically.

It is worthwhile to note that the previous algorithms with most of the state-of-the-art methods are not applicable for large-scale or streaming data due to their expensive computational cost. As a remedy for the high computational requirement, we propose an end-to-end solution to reduce the complexity of all tasks in subspace clustering, by assuming low-rank-ness of data samples. To the best of our knowledge, this is the first attempt to propose an end-to-end solution over all the tasks in subspace clustering to consider the scalability for large-scale problems with linear time complexity in the number of samples. The above mentioned algorithms are applied to various subspace clustering tasks, including face clustering, motion segmentation, handwritten digits clustering, and action clustering, to demonstrate the superiority of the methods.

Table 1.1 describes the three main problems and their characteristics which will be discussed in detail throughout the dissertation. Table 1.2 summarizes our proposals for every chapter and shows the comparison of them.

Table 1.1: Overview of the main problems discussed in this dissertation. $(\cdot)$ denotes the representative function or algorithm in the literature. "General" means that a wide range of conditions can be applied to the problem.

	Sparse represent.	Low-rank represent.	Subspace clustering
Sparsity	1D	2D	1D or 2D
Rank	No rank	Fixed or unknown	Unknown
No. subspaces	General	Single	Multiple
Data structure	General	General	Structured
Convexity (Methods)	Convex/nonconvex (l_1-norm/l_0-norm)	Convex/nonconvex (RPCA/LRMA)	Convex/nonconvex (SSC,LRR/LRSC)
Challenges	Nonconvexity	Inefficiency, unstable	Scalability
Chapter	Ch.3	Ch.4, 5	Ch.5, 6, 7

Table 1.2: The propose algorithms, represented by loss function f_{loss}, regularization Ω_{reg}, and constraint $\mathcal{C}$. Here, $E \triangleq Y - D$ and $D \triangleq PX$.

	f_{loss}	Ω_{reg}	$\mathcal{C}$
Ch. 3	$\|W \odot E\|_F$	$\|X\|_{\text{SVG}}^{\epsilon}$	$-$
Ch. 4	$\|W \odot E\|_1$	$-$	$-$
	$\|W \odot E\|_1$	$\|X\|_F^2$	$P^T P = I$
	$\|Y - PMP^T\|_1$	$\|M\|_*$	$P^T P = I, M \succeq 0$
Ch. 5	$\|W \odot E\|_1$	$\|D\|_* + \alpha\|D\|_F^2$	$\text{rank}(D) = r$
	$\|W \odot E\|_1$	$\|D\|_* + \alpha\|D\|_F^2 + \beta\|C\|_1$	$D = DC, \text{diag}(C) = 0$
Ch. 6	$\|Y - YZ\|_1$	$\|Z\|_1(\text{or } \|Z\|_*) + \gamma\|Z\|_F^2$	$\text{diag}(Z) = 0$
Ch. 7	$\|Y - YZ\|_F$	$\|Z\|_F^2$	$-$

Chapter 2

Related Work

In this chapter, we first briefly summarize the two main approaches of this dissertation: spare representation and low-rank representation. The low-rank representation usually considers that data lie in a single subspace and it finds a basis matrix whose columns span the subspace. As a general case where data lie in a union of multiple subspaces, we also describe subspace clustering and its popular algorithms. Finally, we also discuss on Gaussian process regression, which is used to model unknown complex functions. With the introduction of the problems and related studies, we describe fundamental and existing practical issues of them, which will be addressed in the subsequent chapters.

2.1 Sparse Representation

Recently, sparse representation of signals has been one of the most successful models in many fields including computer vision and signal processing. Sparse representation has shown to be a powerful tool for high-dimensional data such as images [3, 6], where the goal is to represent or compress cumbersome data using a few representative samples. A simple sparse representation problem (for

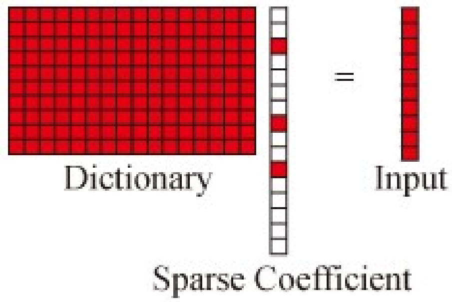


Figure 2.1: Graphical illustration of a simple sparse representation problem.

a noiseless scenario) can be described as follows:

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_0, \quad \text{s.t.} \;\; \boldsymbol{x} = \boldsymbol{D}\boldsymbol{\alpha}, \tag{2.1}$$

where $\|\boldsymbol{\alpha}\|_0 = \#\{i : \alpha_i \neq 0, \ \forall i\}$ is the l_0-norm, $\boldsymbol{x} \in \mathbb{R}^m$ is an observation data, $\boldsymbol{D} \in \mathbb{R}^{m \times p}$ is an overcomplete dictionary ($m \ll p$), and $\boldsymbol{\alpha} \in \mathbb{R}^p$ is the coefficient vector to be estimated. Figure 2.1[1] illustrates the sparse representation problem, where an input vector is represented by sparse linear combination of three selected words in the dictionary. Typical applications of sparse representation include face recognition [5], image restoration [26], and super-resolution [7], to name a few.

Behind the successful outcomes, many efforts have been made for learning the sparse representation efficiently [24, 27, 3, 5, 28, 29, 30, 31, 32, 33], since solving the sparse representation using the l_0-norm has two main drawbacks: (1) the computational intractability of a combinatorial search and (2) its noise sensitivity due to the nature of the l_0 ball. One of the most popular algorithms to estimate sparse signals is the orthogonal matching pursuit (OMP) [24], which finds the best matching projection based on an overcomplete dictionary. However, the greedy pursuit method can find a sub-optimal solution and even can fail to

[1]Source: `http://ranger.uta.edu/~huang/R_Cervigram.htm`

find a reasonable solution. Even worse, there can be a computational issue when the size of the dictionary is large.

There is little doubt that the recent popularity of the sparse representation is attributed to the attempt that the l_0-norm is relaxed to its convex counterpart, i.e., the l_1-norm [34]. In many cases, the use of the l_1-norm turns the problem into convex optimization, which can be efficiently solved with theoretical guarantees. Especially, some analyses showed that the l_1-norm-based problems can exactly recover the best sparse solution under certain conditions [6, 35], making a strong justification for the use of the l_1-norm. Accordingly, the l_1-norm has been extensively utilized in many problems under different forms, and many efficient methods, including the basis pursuit denoising (BPDN) methods such as FISTA [36], have been proposed to solve l_1-norm minimization problems. Even for general problems, for which the exact recovery is not guaranteed, the convex formulation using the l_1-norm may provide an effective and tractable algorithm.

Obviously, the l_1-norm relaxation is beneficial when the relaxed problem or system indeed becomes convex. However, some problems are inherently nonconvex and, for those problems, replacing the sparsity measure to a convex one does not necessarily make the overall problem convex. Some well-known examples of such problems are: matrix factorization [1], rank-constrained subspace learning [22], and recently popularized deep learning [37]. For these problems, using the l_1-norm will not bear as much significance as the previous examples. In fact, for general problems aside from some special (convex) cases mentioned above, the constant slope of the l_1-norm, which is also known as a biased penalty function[2] [28], can over-penalize the values of nonzero elements unlike the l_0-norm and make the solution deviate from the desired solution [28, 29, 32, 33]. This constant slope is

[2]Throughout this paper, we use the term *penalty function* and *measure* interchangeably.

the one that makes the l_1-norm a convex measure, which is not really necessary for the nonconvex settings discussed here. Note that there is a tighter convex approximation to the l_0-norm [38], but it also has a constant gradient along each direction.

As prior works, there have been attempts to use nonconvex smooth (or possibly nonsmooth) approximations of the l_0-norm [27, 39, 28, 29, 40, 30, 32]. We will discuss the theoretical relevance and difference of the proposed measure compared to the nonconvex measures in Section 3.2.2.

2.2 Low-Rank Representation

There are two major approaches to find the low-dimensional subspace structure (low-rank representation) of data: low-rank matrix approximation (LRMA) [1, 10, 41, 42, 21, 19] and robust principal component analysis (RPCA) [43, 5, 41, 35, 44, 45, 12]. In this section, we briefly review the two approaches and consider their limitations.

2.2.1 Low-rank matrix approximation

We briefly review a fixed-rank matrix factorization problem based on the l_1-norm and discuss its related work. The problem arises in a number of problems in computer vision, pattern recognition, and machine learning to handle missing data, such as rigid and non-rigid motion estimation [46, 47], collaborative filtering [1, 41, 42], and background modeling [5, 48, 22], to name a few. A minimization problem based on the l_1-norm can be regarded as a maximum likelihood estimation problem under the Laplacian noise distribution [10, 21].

We first consider a problem for a vector $\mathbf{y} = [y_1 \; y_2 \; \dots \; y_m]^T$ by a multiplication of a vector $\boldsymbol{x} \in \mathbb{R}^m$ and a scalar α, i.e.,

$$\boldsymbol{y} = \alpha \boldsymbol{x} + \boldsymbol{\delta}, \tag{2.2}$$

where $\boldsymbol{\delta}$ is a noise vector whose elements have the independently and identically distributed Laplacian distribution [21]. The probability model for (2.2) can be written as

$$p(\boldsymbol{y}|\boldsymbol{x}) \sim \exp\left(-\frac{\|\boldsymbol{y} - \alpha\boldsymbol{x}\|_1}{s}\right), \tag{2.3}$$

where $\|\cdot\|_1$ denotes the l_1-norm, and $s > 0$ is a scaling constant [10]. Maximizing the log likelihood of the observed data is equivalent to minimizing the following cost function for given $\boldsymbol{x}$:

$$J(\alpha) = \|\boldsymbol{y} - \alpha\boldsymbol{x}\|_1. \tag{2.4}$$

The problem (2.2) can be generalized to the problem of matrix approximation. Let us consider the l_1 approximation of matrix Y such that

$$\min_{P,X} J(P, X) = \|Y - PX\|_1, \tag{2.5}$$

where $Y \in \mathbb{R}^{m\times n}$, $P \in \mathbb{R}^{m\times r}$, and $X \in \mathbb{R}^{r\times n}$ are the observation, projection, and coefficient matrices, respectively. Here, r is a predefined parameter less than $\min(m, n)$ and PX is a low-rank approximation of Y. Typical illustration of the low-rank approximation problem is described in Figure 2.2. In addition, since it is difficult to obtain observations for all entries of the observation matrix in practice, this problem can be considered as the following weighted low-rank matrix approximation problem to consider unknown entries:

$$\min_{P,X} ||W \odot (Y - PX)||_1, \tag{2.6}$$

where W is a weight or mask matrix, whose element w_{ij} is 1 if y_{ij} is known and 0 if y_{ij} is unknown, and $\odot$ is the component-wise multiplication or the Hadamard product.

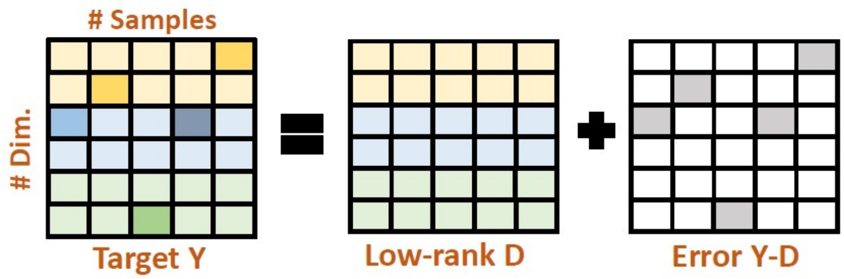


Figure 2.2: Graphical illustration of a typical low-rank representation problem. An observation matrix Y can be decomposed into a clean low-rank matrix D and a noisy matrix $E \triangleq Y - D$. In the problem, D can be factorized by two matrices P and X, i.e., $D = PX$ for fixed-rank representation.

Despite the robustness against outliers, the discussed l_1-norm based methods require a heavy computational load for finding a solution using linear or quadratic programming [10], which requires a large number of iterations to obtain a reasonable solution, making them applicable only for small-scale problems. To overcome the computational complexity issue, methods based on an augmented Lagrangian method (ALM) have been proposed [11, 22] and it solves the problem using an alternating minimization technique, which minimizes the cost function with respect to one target variable while other variables are held fixed. In addition, a nuclear-norm regularized l_1-norm minimization method (Regl_1-ALM) has been proposed to improve convergence by introducing an implicit rank constraint into the cost function via the bilinear form of PX [49, 50]. However, it is difficult for a matrix factorization method to find the global optimal solution because the considered problem is non-convex. Furthermore, when the rank of the data matrix is unknown, the problem becomes more challenging.

2.2.2 Robust principal component analysis

Low-rank matrix approximation finds a low-rank matrix representation of an observation or data matrix, such that the difference between the estimated low-rank matrix and the observation matrix is small. This problem is an attractive topic with a great variety of applications. A well-known method for addressing this issue is robust principal component analysis (RPCA) [43, 5, 41, 35, 44, 45, 12]. RPCA decomposes an observation matrix into a low-rank matrix and a sparse matrix by solving the l_1-norm regularized nuclear-norm minimization problem as follows:

$$\min_{Z,E} ||Z||_* + \lambda ||E||_1, \quad \text{s.t.} \quad Y = Z + E, \tag{2.7}$$

where Z, E, and Y are low-rank, sparse error, and observation matrices, respectively. Here, the nuclear-norm or trace-norm of a matrix is the sum of its singular values, i.e., $||A||_* = \sum_i \sigma_i(A)$, where $\sigma_i(A)$ are singular values of A. RPCA has recently achieved many successful results in machine learning and computer vision, such as background modeling, corruption removal, and collaborative filtering [5, 41, 35, 45]. However, RPCA may not be suitable for solving fixed-rank matrix approximation problems for which the rank of the target matrix is known or can be reliably estimated beforehand. It has been reported that RPCA can sometimes fail to find a (nearly) correct rank when there are many outliers [49, 21]. In addition, since RPCA methods decompose an observation matrix into low-rank and sparse matrices of the same size unlike factorization methods [49, 21], the computational load of RPCA for each iteration can be heavier. Moreover, since RPCA is transductive, it cannot incorporate new data incrementally for online computation [50, 51], making it less scalable.

2.3 Subspace Clustering

Subspace clustering [15] segments data samples into their respective subspaces, which is defined as follows:

Definition 1 (Subspace clustering)**.** *Given a set of samples $\boldsymbol{X} = [\boldsymbol{X}_1, ..., \boldsymbol{X}_k] = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$ drawn from a union of k subspaces $\{\boldsymbol{S}_i\}_{i=1}^k$. Let $\boldsymbol{X}_i$ be a collection of n_i samples drawn from the subspace $\boldsymbol{S}_i$ and $n = \sum_{i=1}^k n_i$. The task of SC is to segment the samples according to the respective subspaces they are drawn from.*

While previously proposed clustering techniques, such as spectral clustering [52], are generally based on a given distance measure, subspace clustering finds cluster memberships of data points using a linear combination of other data points (or a linear combination of basis vectors in a dictionary or observation matrix) with the assumption that data are self-expressive. There are two main tasks to achieve subspace clustering. We first compute an affinity matrix to represent multiple subspaces and then apply clustering algorithms, such as Normalized Cuts [52], to the affinity matrix to identify subspace memberships of data samples. Most of subspace clustering algorithms are focused on finding a good affinity matrix. Two popular algorithms of subspace clustering is sparse subspace clustering (SSC) [53, 16] and low-rank representation (LRR) [54, 4]. Typical applications of subspace clustering include motion segmentation [55], face clustering [54], and digit clustering [56]. Figure 2.3 shows a subspace clustering example.

2.3.1 Sparse subspace clustering

The basic idea of SSC [53, 16] is to find a sparse representation of a sample using a linear combination of other samples in the same cluster by assuming that the

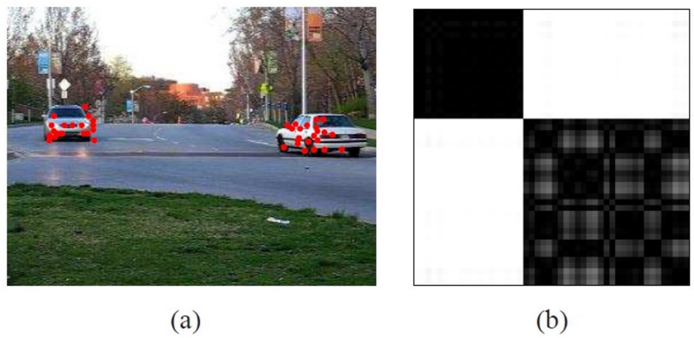


Figure 2.3: Motion segmentation example for subspace clustering [57]. (a) Two motions, each forming one subspace. (b) Affinity matrix obtained by the subspace clustering method in [57]. Clustering results are obtained by performing spectral clustering [52] to the obtained affinity matrix.

observation data can be represented by itself. The basic problem of SSC without noises is formulated as follows:

$$\min_{Z} \|Z\|_1, \quad s.t. \;\; X = XZ, \; \mathrm{diag}(Z) = 0, \tag{2.8}$$

where Z is a subspace representation matrix or a latent matrix to identify clusters in data and $\|Z\|_1$ is the l_1-norm of Z, which is the entry-wise sum of absolute values in Z. Since the subspace representation matrix is unbalanced, an affinity matrix of an undirected graph is built as $Z = (|Z| + |Z^T|)/2$, where $|Z|$ is an element-wise absolute value operator. Finally, by performing spectral clustering, such as Normalized Cuts [52] and NJW [58], we can segment observed samples into k clusters. Although SSC works well in practice, it can seek to find the sparsest representation. Hence, it may divide samples in the same cluster into different clusters. Thus, it lacks the capability of capturing the similarity between samples which are drawn from the same cluster.

2.3.2 Low-rank subspace clustering

LRR [54, 4] is a subspace clustering method which seeks to find the lowest rank subspace representation matrix. By relaxing the rank function to the nuclear norm, which is the sum of singular values of a matrix, the LRR problem is constructed as follows:

$$\min_{Z} \|Z\|_*, \quad s.t. \;\; X = XZ, \tag{2.9}$$

and the problem (2.9) has a closed-form solution [4]. LRR is similar to the well-known low-rank approximation algorithm, robust PCA (RPCA) [35], in that they use a rank minimization approach to find a low-rank solution. Since RPCA does not have a self-expressive system unlike LRR, it cannot perform a clustering task. Therefore, we can see that LRR is a general form which addresses both subspace learning and clustering. Notice that, unlike SSC, LRR is based on a dense representation by enforcing the low-rank property to the representation matrix and has the grouping effect as discussed in [59].

2.3.3 Scalable subspace clustering

While the above mentioned methods have been successfully applied to difficult clustering problems, there are still challenges in terms of scalability and an ability to handle out-of-samples. These methods compute an affinity matrix using all observed samples in a batch mode, which are iterative or computationally intensive approaches.

To address these limitations, three types of methods have been proposed recently: fast [60, 61], distributed [62], and scalable learning [63, 64, 65]. First, two speed-up approaches for solving subspace clustering were proposed [60, 61]. Even though they run faster than existing baseline algorithms, they still have iterative procedures with high computation at each iteration and only consider the affinity learning step. The goal of [62] is to reduce the computation complexity using distributed learning for large-scale problems. It utilizes a divide-factor-combine technique for an LRR problem, which solves LRR for small matrices in a distributed manner and combines resulting small affinity matrices to form an overall affinity matrix. However, its clustering performance depends on the number of partitions and each partition must have an enough number of samples for each cluster to achieve a reasonable performance since LRR assumes that there are enough samples for each cluster [54]. Another type of approaches is a scalable method for handling out-of-sample data, named scalable SSC (SSSC) [63]. It first performs SSC for in-sample data (or a selected small number of samples) and classifies out-of-sample data using the learned subspaces. It assumes that in-sample data are collected from all subspaces to represent out-of-sample data. However, since SSSC assigns the cluster membership using linear classification without spectral clustering, the performance of SSSC can be degraded. Recently, another scalable subspace clustering algorithm, SSC-OMP [64, 65] has been proposed to speed-up SSC. But, it only focuses on reducing complexity when constructing an affinity matrix without considering post-processing and spectral clustering steps, which have heavy computational complexity. Thus, scalability of this approach is still limited in practice.

2.4 Gaussian Process Regression

A Gaussian process (GP) is a collection of random variables which has a joint Gaussian distribution and is specified by its mean function $m(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ [66]. A Gaussian process $f(\boldsymbol{x})$ is expressed as:

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \tag{2.10}$$

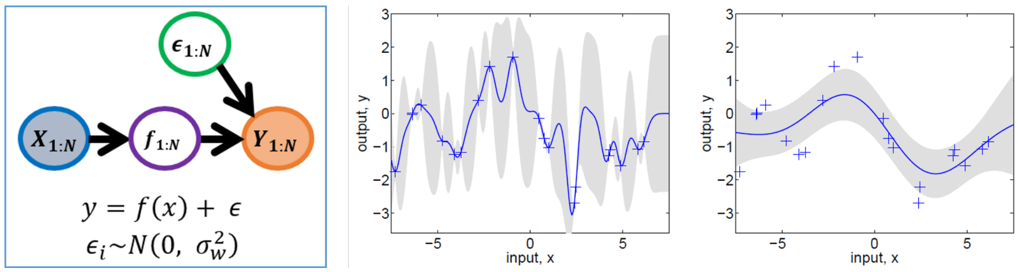


Figure 2.4: Graphical illustration of a Gaussian process. Left: Graphical model for a Gaussian process for regression. Right: Gaussian process regression results for modeling an unknown function.

and its graphical explanation is shown in Figure 2.4. Suppose that $\boldsymbol{x} \in \mathbb{R}^{n_x}$ is an input and $y_i \in \mathbb{R}$ is an output. For a noisy observation set $D = \{(\boldsymbol{x}_i, y_i)|i = 1, ..., n\}$, we can consider the following observation model:

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i, \tag{2.11}$$

where $\epsilon_i \in \mathbb{R}$ is a zero-mean Gaussian noise with variance σ_ϵ^2. Then the covariance of y_i and y_j can be expressed as

$$\mathbf{cov}(y_i, y_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sigma_\epsilon^2 \delta_{ij}, \tag{2.12}$$

where δ_{ij} is the Kronecker delta function which is 1 if $i = j$ and 0 otherwise. $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$ is a covariance function based on some nonlinear mapping function ϕ. The function k is also known as a kernel function.

We can represent (2.12) in a matrix form as follows:

$$\mathbf{cov}(\boldsymbol{y}) = K + \sigma_\epsilon^2 I, \tag{2.13}$$

where $\boldsymbol{y} = [y_1 \ \ldots \ y_n]^T$ and K is a kernel matrix such that $[K]_{ij} = k(x_i, x_j)$.

The conditional distribution of a new output y_* at a new input $\boldsymbol{x}_*$ given D

becomes

$$y_*|D, \boldsymbol{x}_* \;\sim\; \mathcal{N}(\overline{y}_*, \sigma^2_{y_*}), \tag{2.14}$$

where

$$\overline{y}_* = k_*^T(K + \sigma_\epsilon^2 I)^{-1}\boldsymbol{y} = k_*^T \Lambda \boldsymbol{y}, \tag{2.15}$$

where $\Lambda = (K + \sigma_w^2 I)^{-1}$ and the variance of y_* is

$$\sigma^2_{y_*} = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k_*^T(K + \sigma_\epsilon^2 I)^{-1} k_*. \tag{2.16}$$

Here, $k_* \in \mathbb{R}^n$ is a covariance vector between the new data $\boldsymbol{x}_*$ and existing data, such that $[k_*]_i = k(x_*, x_i)$. Note that when it comes to making a prediction given a collected training set, the computational cost of GP can be reduced by pre-computing the inverse of a kernel matrix [67].

Chapter 3

Efficient Nonconvex Sparse Representation

In this chapter, we propose a nonconvex sparsity measure for sparse representation (SR) in general nonconvex problems which complements both l_0- and l_1-norms from practical considerations. The motivation emerges as the following question: What is a good nonconvex sparsity measure if it is not possible to transform a problem to a convex one? As an answer to this question, we first analyze the possible approximations of the l_0-norm. Then, we propose the desirable criteria to be a good nonconvex measure and present a representative family of curves, termed *slowly vanishing gradient* (SVG), that is a solution of a differential equation. We also show that there is a trade-off between the values and the vanishing speed of their gradients. Interestingly, these analyses lead to a simple but effective nonconvex sparsity measure, which was proposed over two decades ago [39], and we shed light on the measure with new analysis and algorithms since it did not receive much attention compared to other popular penalties in the literature. In [39], the measure was simply proposed as an approximation of the l_0-norm without analysis similar to ours. In this study, however, we find that the measure has very important property of having its gradient vanishing slowly. Locally, the measure follows the l_1-norm to reduce the chance of numerous local optima without losing the ability of promoting sparsity. Globally, it follows the l_0-norm to reduce penalty on large-values, but it still possesses slowly vanishing gradients to help drawing the solution of an optimization algorithm to sparse points. Moreover, we present an efficient proximity operator for the measure. The proposed measure is applied to various applications, including low-rank approximation (LRA), sparse coding with dictionary learning (SC), and sparse subspace clustering (SSC) problems, to demonstrate its adequacy and experimental results confirm that the proposed method performs favorably against those of other well-known sparsity measures.

3.1 Analysis of the l_0-norm approximation

3.1.1 Notations

An observation matrix is denoted by $\boldsymbol{X} \in \mathbb{R}^{m\times n}$, where each column corresponds to a data sample in $\mathbb{R}^m$. We denote matrices, vectors, and scalars by bold letters in upper case, bold letters in lower case, and letters in lower case, respectively, unless stated otherwise. Spaces and subspaces are denoted by bold italic letters in upper case. Throughout this chapter, we use $\|\boldsymbol{A}\|_q$ to denote matrix norms of a matrix $\boldsymbol{A}$, with $q = 1$ for the matrix l_1-norm, $\sum_{ij} |a_{ij}|$, and $q = F$ for the Frobenius-norm, $\sqrt{\sum_{ij} |a_{ij}|^2}$. We denote the projection operator by $\mathcal{P}(\cdot)$ and the support set of a matrix $\boldsymbol{A}$ by $\Omega_{\boldsymbol{A}}$. $\text{rank}(\boldsymbol{A})$ denotes the rank of $\boldsymbol{A}$ and $|\cdot|$ denotes the absolute value operation of a scalar. Diagonal elements in a matrix $\boldsymbol{A}$ is denoted by $\text{diag}(\boldsymbol{A})$.

3.1.2 Desirable criteria for a nonconvex measure

In this section, we will mainly discuss about a sparse representation problem whose cost function consists of a data term and a regularizer. As explained earlier, if the problem itself (data term) has a nonconvex structure, then the convexity of the sparsity measure (regularizer) is not absolutely necessary. In this case, the constant slope of the l_1-norm will not necessarily make the problem convex but over-penalize nonzero values in the input, which makes the solution deviate from the desired solution, especially when the problem assumes the presence of noises. Hence, we might be interested in finding a good nonconvex measure for such general nonconvex problems. Prior works support the superiority of nonconvex sparsity-promoting measures [29, 40, 32, 68, 69].

If the nonconvexity of the l_0-norm is not a problem, then the only difficulty in handling it is that its value only changes around zero (or we can imagine that its shape appears as if it gives an extremely local gradient at the origin), which is very bad from the perspective of conventional optimization methods. That is, the derivative of the l_0-norm is zero for nonzero inputs, which has no effect on gradient-based optimization, and is not well-defined otherwise, which can be difficult for discovering a good local optimum. In order to find a measure which has least undesirable effects on nonzero values and can also be handled efficiently in the conventional optimization methods, we might consider smooth approximations of the l_0-norm [29, 30, 33]. However, there can be infinitely many such approximations and we need some criteria for finding a good measure. Below are basic assumptions to be a good candidate:

Assumption 1. *We pose the following criteria on the measure* $\phi(x)$[1] *(defined*

[1]For ease of explanation, we sometimes deal with a scalar function throughout the paper due to the separability of the measure, even though this chapter is about the sparsity-promoting

on $-\infty \leq x < \infty$) we are looking for:

1. *Symmetry: The sign of an input does not matter but the magnitude, hence, we assume $\phi(x) = \phi(-x)$.*

2. *Asymptotic convergence: Assume $\phi(0) = 0$. Then, $\phi(x)$ satisfies $\lim_{x\to\infty} \phi(x) = 1$. This prevents $\phi(x)$ from penalizing large nonzero inputs equally as small ones, and makes it closer to the l_0-norm.*

3. *Monotonicity: In order for ϕ to be a valid measure, we assume $\phi'(x) > 0$ for $x > 0$ where $\phi'(x)$ is the derivative of $\phi(x)$ at x, i.e., ϕ is a monotonically increasing function on $x > 0$.*

4. *Smoothness (Monotonicity of gradient): There can be some choices of ϕ that $\phi'(x)$ goes up and down, but this behavior is unnecessary and will over-complicate $\phi(x)$. Hence, we assume $\phi''(x) < 0$ for $x > 0$, i.e., the gradient decreases monotonically for $x > 0$.*

5. *Finite nonzero gradient at $x = 0$: Let us define the "gradient at $x = 0$" as $\phi'(0^+) = \lim_{x\to 0^+} \phi'(x)$. Then, $\phi'(0^+)$ should be a finite nonzero value to promote sparsity, i.e., $0 < \phi'(0^+) = b < \infty$. In many examples, b will be chosen as $b = 1$ for ease of explanation.*

Remark 1. *We give more details for the last assumption. First, $\phi'(0^+)$ should be nonzero to promote sparsity. This being nonzero makes the Clarke's generalized gradient [70], $\bar{\partial}\phi$, at $x = 0$, has a nonempty interior, which increases the chance of the (local) optimum being a sparse point as for the l_1-norm. Second, $\phi'(0^+)$ should be finite, so that $\bar{\partial}\phi(0)$ is bounded. This can be good for not creating too many local optima at sparse points, because unfavorable local optima can be deviated*

penalty. An extension to a vector case is straightforward.

due to the influence of the data term whose slopes are high enough. If $\phi'(0^+)$ is unbounded, the possibility of local optima can increase for various sparse points, many of which will not be good solutions. The "finite nonzero gradient at $x = 0$" assumption is thus important, in that it makes the problem prefer solutions that are not only sparse, but also have small values for the data term, as for the case of using the l_1-norm.

Aside from the above criteria, we have another criterion on the choice of ϕ. As discussed before, the gradient either being 0 or not being well-defined is what makes the optimization difficult for the l_0-norm. Thus, we aim to find a measure that has an opposite characteristic: $\phi(x)$ whose gradient is as large as possible across the entire interval. Because of the fourth criterion above, this is equivalent to finding $\phi(x)$ that has *slowly vanishing gradient*. If $\phi'(x)$ decreases slowly, then the effect of the sparsity measure can spread across a large region to help drawing the solution to sparse points. This can be viewed as mimicking the constant slope of the l_1-norm under the above criteria. Hence, we might try to find $\phi(x)$ with the most slowly decreasing gradient. However, due to the second criterion, the "total amount" of gradient is finite, i.e., $\int_{0^+}^{\infty} \phi'(x)dx = 1$. This means that we have to divide this finite value for $0 < x < \infty$.

3.1.3 A representative family of measures: SVG

To analyze the situation discussed above more closely, we present two extreme examples among the possible family of measures that satisfy the above criteria. Because of the first criterion, we can assume $\phi(x) = y(|x|)$ for some function y on $\mathbb{R}^+$.

First, let us see an example that is a smooth relaxation of the l_0-norm, but its gradient is still concentrated in a relatively local region. An easy example is

$$y = 1 - e^{-x}, \tag{3.1}$$

which satisfies $y(0) = 0, y(\infty) = 1$, $y'(0^+) = 1$, and all of the above criteria. Its derivative is $y'(x) = e^{-x}$, which means that the gradient vanishes exponentially. Hence, this measure will quickly become negligible except the local region near $x = 0$.

As an opposite example, let us consider a case, in which the gradient vanishes very slowly;

$$y = 1 - \frac{1}{(1 + \frac{x}{a})^a}, \tag{3.2}$$

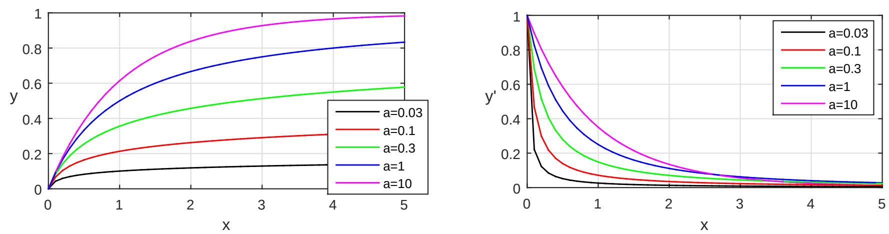
with very small $a > 0$. Its derivative is

$$y'(x) = \frac{1}{(1 + \frac{x}{a})^{1+a}}, \tag{3.3}$$

and this also satisfies $y(0) = 0, y(\infty) = 1$, $y'(0^+) = 1$, and all of the above criteria. Here, since a is very small, $y'(x)$ is close to a reciprocal function $\frac{1}{1+\frac{x}{a}}$. Integrating $\frac{1}{1+\frac{x}{a}}$ for $0 \leq x < \infty$ does not converge, hence, this can be seen as an extreme example with very slowly vanishing gradients. However, $\frac{1}{(1+\frac{x}{a})^{1+a}}$ is very close to 0 for most of x, which is a natural consequence of spreading a finite value ($\int_0^\infty y'(x)dx = 1$) to a broad interval. Indeed, we can verify that $\lim_{a\to 0} \frac{1}{(1+\frac{x}{a})^{1+a}} = 0$ if $x \neq 0$ and the function itself approaches to zero, i.e., $\lim_{a\to 0} 1 - \frac{1}{(1+\frac{x}{a})^a} = 0$. Note that the previous example can be viewed as an opposite extreme in this sense as

$$\lim_{a\to\infty} \frac{1}{(1 + \frac{x}{a})^{1+a}} = e^{-x}. \tag{3.4}$$

Therefore, there is a tradeoff between the spread (vanishing speed) of gradients and their actual values. Some example curves of y and its derivative y' for various values of a are illustrated in Figure 3.1.



(a) (b)

Figure 3.1: Graphical illustration of a family of representative curves (a) y and (b) their derivatives y' for different choices of a.

In addition to the extreme examples, there are infinitely many functions that satisfy our criteria. However, the details of curve shapes do not matter much because local differences between two curves does not bear a significant meaning for general problems. Hence, it suffices to choose a representative family of curves that has a nice interpretation and includes various rates of gradient vanishment, in order to narrow down our choices. In fact, the previous examples are good candidates, since they are solutions to the following differential equation that has an elegant meaning:

$$(1-y)^{1+\frac{1}{a}} = \epsilon y', \quad y(0) = 0, \tag{3.5}$$

where $a > 0$ and $\epsilon > 0$ are parameters. It is worth noting that $(1-y)$ on the left side is the difference between the l_0-norm and y, thus, the decreasing speed of $(1-y)$ is identical to the rate of asymptotic convergence (criterion 2). Therefore, this equation describes the rate of gradient vanishment in terms of the rate of asymptotic convergence. This can be transformed into a Bernoulli equation, and the solution is given as

$$y(x) = 1 - \frac{1}{(1+\frac{x}{a\epsilon})^a}, \tag{3.6}$$

which satisfies $y'(0^+) = \frac{1}{\epsilon}$, $y(0) = 0$, and $y(\infty) = 1$ for $a > 0$. We call the corresponding penalty functions satisfying the equation (3.6) as a family of *slowly vanishing gradient* (SVG) measures. As a special case of the family of SVG measures when $\epsilon = 1$ and $a \to \infty$, the solution leads to (3.1), i.e., $y = 1 - e^{-x}$.

3.2 The Proposed Nonconvex Sparsity Measure

3.2.1 Choosing a simple one among the SVG family

As explained in the previous section, there is a tradeoff between the vanishing speed and the actual value of the gradient. Thus, we can, at best, choose a good compromise between them. Since there is no clear winner between the curves in our SVG family, it is better to choose a simplest one among the reasonable choices. Accordingly, we constrained a to be an integer, and find one that gives the slowest decreasing rate of gradient, which is $a = 1$. As a result, we have $y(x) = 1 - \frac{\epsilon}{x+\epsilon} = \frac{x}{x+\epsilon}$. Based on this function, our proposed sparsity measure[2] is given as follows:

$$\|\boldsymbol{\alpha}\|_{\mathrm{SVG}}^{\epsilon} = \sum_i \frac{|\alpha_i|}{|\alpha_i| + \epsilon}, \tag{3.7}$$

where $\epsilon > 0$ is a weighting factor that determines the slope at $\alpha_i = 0^+$.

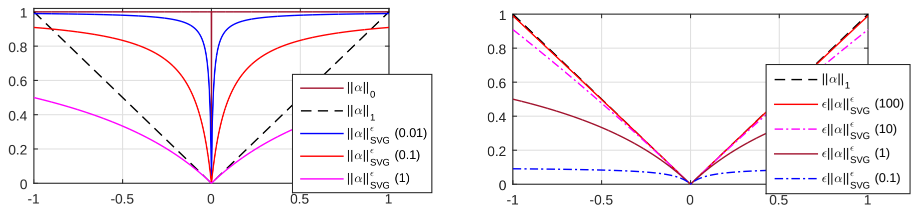
Proposition 1. *SVG approximates the l_0- and l_1-norms:*

1. $\|\boldsymbol{\alpha}\|_{SVG}^{\epsilon} \leq \|\boldsymbol{\alpha}\|_0 \ \forall\epsilon$ *and* $\|\boldsymbol{\alpha}\|_{SVG}^{\epsilon} \to \|\boldsymbol{\alpha}\|_0$ *if* $\epsilon \to 0$.

2. $\epsilon\|\boldsymbol{\alpha}\|_{SVG}^{\epsilon} \leq \|\boldsymbol{\alpha}\|_1 \ \forall\epsilon$ *and* $\epsilon\|\boldsymbol{\alpha}\|_{SVG}^{\epsilon} \to \|\boldsymbol{\alpha}\|_1$ *if* $\epsilon \to \infty$.

Proof. See Appendix C. □

[2] We just denote the measure as SVG in that it is one of our SVG family.



(a) (b)

Figure 3.2: Graphical illustration of SVG of a vector $\boldsymbol{\alpha}$ with respect to various values of ϵ (a) compared to the l_0-norm, and (b) to the l_1-norm. $(\cdot)$ denotes the value of ϵ.

Note that the above properties still hold for the proposed SVG family based on (3.6). Some example curves of SVG are illustrated in Figure 4.6 to visualize these properties.

Another nice property of SVG is that it possesses a simple proximity operator. Recently, there have been remarkable theoretical progresses on convergence analysis for the sparse optimization techniques, and nonconvex versions for the accelerated proximal gradient method (nAPG) [71] and the alternating directional method of multipliers (nADMM) [72] have been proposed to solve sparse optimization problems efficiently in nonconvex settings. Hence, even though SVG is nonconvex, having a simple proximity operator is still a good advantage to incorporate the above methods for efficient nonconvex programming.

The proximity operator for SVG is defined by the following problem:

$$\mathrm{prox}^{\epsilon}_{\mathrm{SVG},\lambda}(\boldsymbol{x}) = \min_{\boldsymbol{u}} \lambda \|\boldsymbol{u}\|^{\epsilon}_{\mathrm{SVG}} + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{u}\|^2. \tag{3.8}$$

Note that this equation is separable, and we can solve it for each element of $\boldsymbol{u}$. Since SVG is a symmetric function for each element, an element of the solution vector $\hat{\boldsymbol{u}}$ will either be of the same sign with the corresponding element of $\boldsymbol{x}$

or be zero. Let us assume that the sign of x_i, the ith element of $\boldsymbol{x}$, is positive without loss of generality. Then, one of the positive solutions of the following cubic equation

$$\begin{aligned}(u_i+\epsilon)^2&\left(\frac{\lambda u_i}{u_i+\epsilon}+\frac{1}{2}(x_i-u_i)^2\right)' \\ &= \lambda\epsilon + (u_i - x_i)(u_i+\epsilon)^2 \triangleq g(u_i) = 0\end{aligned} \tag{3.9}$$

or zero will be the optimal point of u_i. Note that the coefficient of the third-order term of $g(u_i)$ is positive, as well as the value of $g(-\epsilon) = \lambda\epsilon > 0$. This indicates that $g(u_i)$ has at least one root for $u_i < 0$, i.e., there can be at most two roots for $u_i \geq 0$. If there is no root or a double root for $u_i \geq 0$, $g(u_i)$ is nonnegative for $u_i \geq 0$, i.e., the cost function is monotonically increasing for positive u_i, and the optimal point will be 0. If there are two distinct roots, then the solution with a larger value is a local minimum, so either this solution or zero will be the optimal point. In conclusion, the optimal $\hat{u}_i$ is either the largest positive root of (3.9) or zero, and we can compare the costs of these two points to find the final solution. This analysis will relieve the computational complexity when solving the third-order equation.

3.2.2 Relationships with other sparsity measures

There are many nonconvex sparsity-promoting measures (regularizers), such as smoothly clipped absolute deviation (SCAD) [28], minimax concave penalty (MCP) [32], and Capped-l_1 penalty [40], which have been proposed to approximate the l_0-norm. Extensions to low-rank representation for the nonconvex measures have been explored in [69]. A comprehensive study on the nonconvex sparsity measure can be found in [68, 73]. In [28], authors advocate a nonconvex penalty function that has three desired properties: unbiasedness, sparsity, and continuity. More general properties to be a good nonconvex penalty are described in [73] (see Assumption 1). Note that our family of measures satisfies the conditions and so it is covered by the well-developed theory for good nonconvex sparsity penalty functions [73]. Further details on this point are included in Section 3.2.3. Besides, ours further extends the properties by introducing an important new criterion: We suggest the slowly vanishing gradient criterion and derive a corresponding family of measures. The above penalties do not satisfy this condition, since they have large *flat regions* (gradient zero or quickly converging gradient). This may increase the chance of local optima if some local optima of a loss function (data term) are located at the plateau of the penalty functions (regularizers). Our aim is to mitigate this effect.

Unlike the previous functions that give a large flat region, there is another line of penalty functions as an alternative to the original l_0-norm, such as the l_q-norm penalty ($0 < q < 1$) [27], which gives a constantly inclinatory curve analogous to the proposed penalty. However, there is no analysis about the l_q-norm analogous to ours. Even worse, the l_q-norm is known to be difficult to solve because it is not separable and it does not have an efficient proximity operator due to the q-th power, making it less practical. Whereas, ours enjoys a simple proximity operator and handles the raised issues efficiently. Analogous to the l_q-norm penalty, the log-sum penalty (LSP) [29] gives a non-flat curve similar to ours, but it does not give the satisfying performance compared to the proposed penalty as shown in Section 3.3.1. There has been another attempt to use a smooth approximation of the l_0-norm based on an exponential function in [30], but no analysis was provided for justifying such a choice. In fact, our analysis shows that the approximation based on an exponential function also has fast vanishing gradients, which is more prone to local optima, and thus this approximation does not give satisfactory performance as shown in Section 3.3.4.

While preparing this manuscript, we became aware of that our proposal, as a special case of the SVG family, leads to the same type of measure proposed by Geman and Yang [39] (sometimes called the Geman penalty) over two decades ago. However, it is important to note that there are clear differences between their and our studies. First, the specific choice for approximating the l_0-norm is not justified in [39] because its focus is an image reconstruction problem. Second, the optimization approach in [39] is outdated, while we provide efficient algorithms based on a proximity operator derived from a nice property of the penalty.

To the best of our knowledge, our analysis gives a new insight from the optimization perspective for nonconvex sparsity-promoting penalty functions. The proposed penalty provides superior performance compared to the existing nonconvex and convex surrogates of the l_0-norm, because it has (1) a slowly vanishing gradient to reduce the chance of local optima, (2) unbiasedness to reduce the over-penalized issue due to the constant gradient of the l_1-norm. Besides, it is easily solvable by its simple and separable proximity operator. Experimental evidences verify the superiority of the proposed penalty in Section 3.3.

3.2.3 More analysis on SVG

We show that the sparse representation based on the SVG measure (regularizer) satisfies the well-studied theory for nonconvex sparsity-promoting measures [73], whose graphical illustrations are shown in Figure 3.3. In order for the proposed family to apply the theory, we need to show that our family satisfies the following well-analyzed assumptions:

Assumption 2 ([73])**.** *We consider a scalar variable x for simplicity and define a regularizer as $\phi_\lambda : \mathbb{R} \to \mathbb{R}$.*

1. *The function* ϕ_λ *satisfies* $\phi_\lambda(0) = 0$ *and is symmetric around zero (i.e.,* $\phi_\lambda(x) = \phi_\lambda(-x)$ *for all* $x \in \mathbb{R}$*).*

2. *On the nonnegative real line,* ϕ_λ *is nondecreasing.*

3. *For* $x > 0$*, the function* $x \mapsto \frac{\phi_\lambda(x)}{x}$ *is nonincreasing.*

4. *A measure* ϕ_λ *is differentiable for all* $x \neq 0$ *and subdifferentiable at* $x = 0$*, with* $\lim_{x \to 0^+} \phi'_\lambda(x) = \lambda L$*.*

5. *There exist* $\mu > 0$ *such that* $\rho_{\lambda,\mu}(x) \triangleq \phi_\lambda(x) + \frac{\mu}{2}x^2$ *is convex.*

We first show that our representative family of measures satisfying the criteria presented in Assumption 1 meets the above assumptions:

Lemma 1. *The representative family of measures* ϕ_λ *designed by our criteria with the parameters* ϵ *and* a *satisfies the conditions of Assumption 2 with* $L = \frac{1}{\epsilon}$ *and* $\mu = -\frac{(a+1)\lambda}{a\epsilon^2}$*.*

Proof. See Appendix B. □

From the lemma, we directly obtain the following result on the proposed measure as a special case:

Corollary 1. *The SVG measure with the parameter* ϵ *satisfies the conditions of Assumption 2 with* $L = \frac{1}{\epsilon}$ *and* $\mu = \frac{2\lambda}{\epsilon^2}$*.*

By Corollary 1, we confirm that the proposed measure satisfies the Assumption 2 and this makes that the sparse representation based on the proposed measure can directly follows the theory on the error bound under mild conditions [73]. In other words, any stationary points guaranteed by a nonconvex sparse optimization method are close to the small ball around the optimal point.

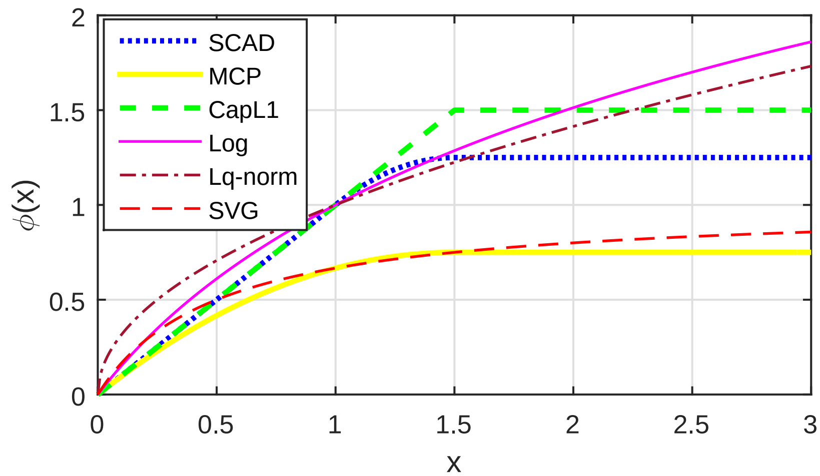


Figure 3.3: Illustrations of curves for nonconvex sparsity measures.

3.2.4 Learning sparse representations via SVG

The proposed measure can be applied to various sparse representation problems that the l_0-norm and l_1-norm are applied. In this section, we focus on three important problems including low-rank approximation (LRA) [1], sparse coding (SC) [3], and sparse subspace clustering (SSC) [53].

SVG for LRA. Sparse representation has been widely used in many applications to filter out outliers in data. One of the most popular applications is the low-rank approximation (LRA) of a matrix under the existence of outliers, and the l_1-norm is usually used to model the sparse outliers [6, 11]. If the rank of a matrix is not specified, then using the nuclear-norm [35] can be a good choice that makes the entire problem convex. However, there are many problems that the rank is explicitly specified, such as structure reconstruction [74] and photometric stereo [35], to name a few. In this case, it becomes a nonconvex problem.

For the LRA problem, we apply SVG for modeling sparse errors, whose problem formulation (LRA-SVG) is constructed as follows:

$$\min_{\boldsymbol{E},\boldsymbol{M}} \|\mathcal{P}_{\Omega_{\boldsymbol{X}}}(\boldsymbol{E})\|_{\mathrm{SVG}}^{\epsilon}, \text{ s.t. } \boldsymbol{E} = \boldsymbol{X} - \boldsymbol{M}, \quad \mathrm{rank}(\boldsymbol{M}) \leq r. \tag{3.10}$$

This problem can be efficiently solved using the nADMM framework [72] as discussed before. The derivation of LRA-SVG is included in Appendix A.

SVG for SC. The proposed measure can be applied to another well-known nonconvex sparse representation problem, sparse coding with dictionary learning [3, 2], which is basically a matrix factorization problem. Unlike LRA problems, SVG is used to enforce the sparsity of the encodings in this case. The problem formulation of SC for an observation vector $\boldsymbol{x}$ based on SVG (SC-SVG) can be given as follows:

$$\min_{\boldsymbol{D},\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_{\mathrm{SVG}}^{\epsilon}, \tag{3.11}$$

where $\boldsymbol{D}$ and $\boldsymbol{\alpha}$ are an overcomplete dictionary consisting of word vectors and a sparse coefficient vector, respectively. This problem is solved in an alternating fashion based on the proximal gradient method.

SVG for SSC. Subspace clustering is a problem to find the cluster memberships of data points based on an assumption that a point can be represented by a linear combination of other points in the same cluster. Note that this problem can be efficiently solved based on convex optimization, nevertheless we apply SVG to this problem, in order to verify the capability of the proposed measure in general problems. We apply SVG to the well-known sparse subspace clustering (SSC) [53], where the corresponding formulation (SSC-SVG) under noisy scenario is given as follows:

$$\min_{\boldsymbol{Z}} \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{X}\boldsymbol{Z}\|_F^2 + \lambda\|\boldsymbol{Z}\|_{\mathrm{SVG}}^{\epsilon}, \quad \text{s.t.} \;\; \mathrm{diag}(\boldsymbol{Z}) = 0. \tag{3.12}$$

This problem can be efficiently solved by nAPG [71]. Especially, we incorporate the nonmonotone update framework [71] to accelerate the convergence of the algorithm.

Note that the initial values of optimization variables for the proposed algorithms are set to zero, based on empirical observations that they are not sensitive to the choice of the initial values.

3.3 Experimental Results

In this section, we report numerical results of the sparse representation algorithms based on SVG. We compare these algorithms with other state-of-the-art algorithms[3]: RPCA-IALM (RPCA-I) [35], ALADM [11], and LRA-L1 (an l_1-norm version of LRA-SVG) for low-rank approximation problems, KSVD [3] and SC [2] for sparse coding problems, and LRR [75], SSC-BP [53], SSC-OMP [76], and SSC-SL0 (SSC based on smoothed l_0-norm [30]) for subspace clustering tasks. We also compare the proposed measure with other well-known nonconvex sparsity measures, SCAD [28], MCP [32], Capped-L1 (CapL1) [40], and LSP [29], in order to demonstrate the superiority of the proposed nonconvex measure for problems described above. For the compared algorithms, we used the codes provided by the authors, unless stated otherwise. For low-rank approximation and sparse coding problems, we compute the reconstruction error as

$$\|\boldsymbol{W} \odot (\boldsymbol{M}^{GT} - \boldsymbol{M})\|_1 / \|\boldsymbol{W}\|_1, \tag{3.13}$$

[3]In order to compare the proposed method with various algorithms, we report experimental results also for convex algorithms based on the l_1-norm.

where $\boldsymbol{M}^{GT}$ and $\boldsymbol{M}$ are the ground-truth and reconstructed matrices, respectively, $\boldsymbol{W}$ is a weight matrix concerning missing entries, and $\odot$ is the Hadamard product operator. For subspace clustering, we compute the accuracy by the Hungarian method [77]. We set the parameter ϵ of SVG to 0.05 for entire experiments, since it was not sensitive to various problems in our empirical experiences. More analyses on parameters are included in Section 3.3.5. All experiments were performed using MATLAB environment on a desktop computer with 24GB RAM and a 3.4GHz quad-core CPU.

3.3.1 Evaluation for nonconvex sparsity measures

We first evaluate the proposed penalty, SVG, on synthetically made examples to compare with other renowned nonconvex sparsity-promoting penalties. We used the codes of other compared penalties provided by the work in [33], which solves the nonconvex optimization problems efficiently with a convergence guarantee. Following the experiments in [33], we performed the sparse approximations based on the penalties, whose problem formulation is to find a sparse coefficient vector $\boldsymbol{\alpha}$:

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{\alpha}\|_2^2 + \phi(\boldsymbol{\alpha}), \tag{3.14}$$

where $\boldsymbol{x} \in \mathbb{R}^m$ is a target vector, $\boldsymbol{D} \in \mathbb{R}^{m \times p}$ is a data matrix, and $\phi(\boldsymbol{\alpha})$ is a penalty function. For all experiments in this subsection, we set $m = p = 500$. We made a scenario by varying sparsity ($0 \sim 90\%$) of a ground-truth coefficient vector $\boldsymbol{\alpha}^{GT}$, where lower sparsity means denser representation, and made an observation $\boldsymbol{x}^{GT}$ from the multiplication of $\boldsymbol{D}$ and $\boldsymbol{\alpha}^{GT}$, which are obtained by the Gaussian distribution from $\mathcal{N}(0, 1)$. Based on $\boldsymbol{x}^{GT}$, we made $\boldsymbol{x}$ by adding Gaussian noises from $\mathcal{N}(0, 10^{-2})$. For each setting in the scenario, we performed k independent runs, where k is set to 30. The average reconstruction error is computed as

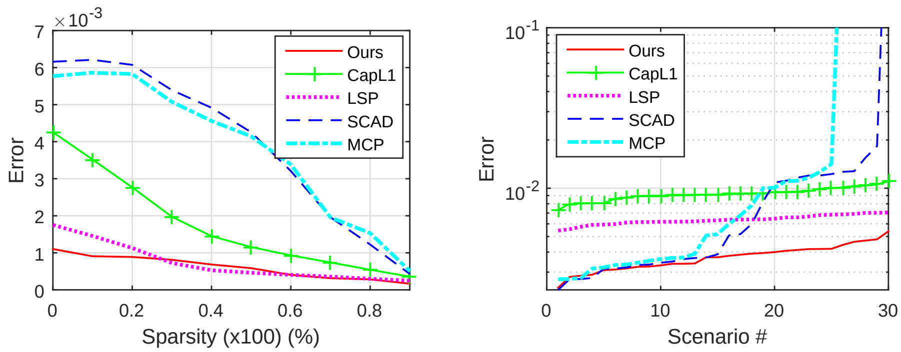
$$\frac{1}{k}\sum_{i=1}^{k}\|\boldsymbol{x}_i^{GT}-\boldsymbol{D}_i\boldsymbol{\alpha}_i\|_2, \tag{3.15}$$

where $\boldsymbol{x}_i^{GT}$ is the ground-truth vector for the i-th scenario.

Results of the compared measures are shown in Figure 3.4. As shown in Figure 3.4(a), the proposed measure performs better than the other nonconvex measures on average. LSP, which represents a similar non-flat curve, gives the similar performance to ours when the sparsity ratio is larger than 30%. SCAD and MCP show the similar but worst performances in this problem. Figure 3.4(b) shows the l_2 errors between the true coefficient vector and obtained vectors based on different measures under the sparsity ratio of 90%. The proposed measure finds all the sparse coefficients with the lowest errors, whereas LSP and CapL1 give larger errors than ours for all cases. SCAD and MCP perform competitively compared to the proposed measure for some scenarios, but they sometimes fail to find the exact coefficient vectors. The average computation times (sec) of the measures for the reconstruction problem are as follows: 0.15 for CapL1, 0.28 for SCAD, 0.26 for MCP, 0.23 for LSP, and 0.3 for SVG, respectively. In the problem, most of the methods take similar execution times.

3.3.2 Low-rank approximation of matrices

We report the results for low-rank approximation problems using both synthetic and real-world problems. To generate synthetic examples, we made a matrix whose size is 500×500 and set the rank of the matrix to 10. In the matrix, we added Gaussian noises with $\mathcal{N}(0, 10^{-5})$ and outliers with magnitude of 10 for randomly chosen elements. The outlier ratio is varied from 0% to 60% to verify the robustness of the proposed method. Here, we compare with three nonconvex



(a) (b)

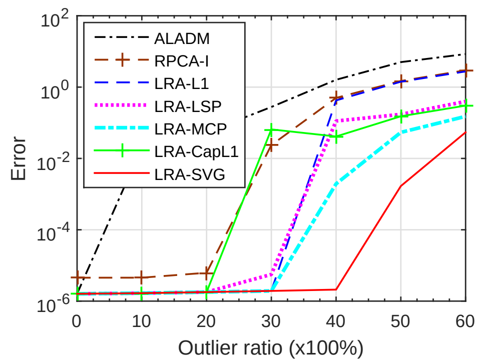
Figure 3.4: Average performances on synthetic examples for nonconvex sparsity measures. (a) Reconstruction errors w.r.t. the sparsity. (b) Errors in ascending order for different scenarios.

penalties in the same framework to ours, LRA-CapL1, LRA-MCP, and LRA-LSP, based on CapL1 [40], MCP [32], and LSP [29], respectively. The experimental results of the synthetic problems for 50 independent trials are described in Figure 3.5(a). From the figure, we can see that the proposed method withstands much higher outlier ratios than the other methods, which confirms its excellent robustness, whereas other methods fail to find a good solution roughly over 30%. The three nonconvex penalty based algorithms mentioned above perform better than the other methods based on the convex penalty, i.e., the l_1-norm, on average, but they could not endure as many outliers as the proposed penalty. The average computation times (sec) of the algorithms are as follows: 0.62 for ALADM, 11.74 for RPCA-I, 1.76 for LRA-L1, 50.24 for LRA-LSP, 13.77 for LRA-MCP, 13.8 for LRA-CapL1, and 3.16 for LRA-SVG, respectively.

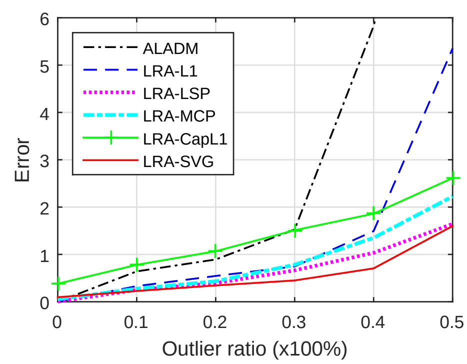
We have performed real-world experiments on two problems; nonrigid motion estimation [13] and photometric stereo [74]. For nonrigid motion estimation, we used the Shark sequence [13]. The rank of the problem is set to $r = 6$. In order to consider missing environments, we replaced 10% randomly selected entries in the Shark dataset as missing. For photometric stereo, we used Static Face dataset [74] which has 42% missing entries. We set the rank to $r = 4$ for this problem. For these problems, we did not evaluate RPCA-IALM because they are rank-constrained matrix completion problems. Figure 3.5(b) and 3.5(c) show the average reconstruction errors of the algorithms for 50 independent runs under various outlier ratios ($0 \sim 50\%$). From the figure, we can confirm that the proposed method outperforms the other methods for both problems. Especially, the proposed method is highly robust against outliers and missing data for the Static Face dataset. While LRA-LSP gives competitive results to the proposed method for the Shark sequence, it performs poorer than ours for the Static Face dataset. The l_1-norm based approaches, LRA-L1 and ALADM, perform worse than other nonconvex measure based algorithms on average for both datasets.

3.3.3 Sparse coding

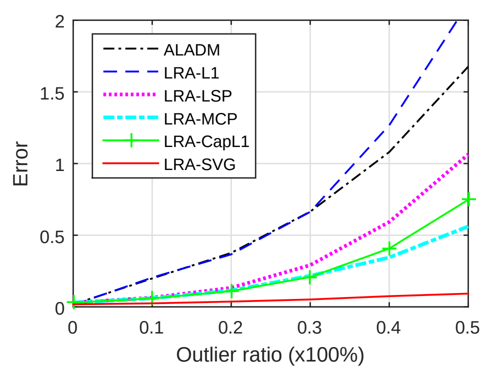
We have conducted experiments for a sparse coding problem (3.11) based on well-known example images in the literature: Barbara, Lena, Boat, and Peppers. Following the practice of [3], we extracted n 64-dimensional word vectors based on 8×8 local patches for each image, where n is the number of training data which was set to $n = 15,000$. Based on these word vectors, we learned both a dictionary and a sparse code for each sample. In the problem, we compare with two well-known sparse coding methods with dictionary learning: SC [2] and KSVD [3]. For all tested images, the size of dictionary $\boldsymbol{D}$ was set to 250, i.e., $\boldsymbol{D} \in \mathbb{R}^{64\times 250}$. In each dataset, we added Gaussian noises from $\mathcal{N}(0, 0.3)$. The average reconstruction errors of the tested algorithms are shown in Table 3.1.



(a) Synthetic dataset



(b) Shark sequence



(c) Static Face dataset

Figure 3.5: Average performances on low-rank approximation problems in the presence of outliers and missing data.

Table 3.1: Average reconstruction errors ($\times 10^2$) for sparse coding.

Methods	Barbara	Lena	Boat	Peppers	Average
KSVD [3]	2.23	1.90	2.04	2.05	2.06
SC [2]	2.11	2.02	2.15	2.12	2.1
SVG (Ours)	**1.15**	**0.7**	**0.97**	**1.09**	**0.98**

In the table, our algorithm gives excellent results for all cases. KSVD, which uses OMP, performs slightly better than SC based on the l_1-norm, but it is unsatisfactory compared to ours.

3.3.4 Subspace clustering

Face clustering. We have evaluated the proposed measure on the Extended Yale B database [78] for subspace clustering. The dataset used for this experiment consists of 38 subjects, each of which has 64 frontal face images under illumination changes. We collected the first c subjects, where $c \in \{2, 5, 8, 10\}$, and performed subspace clustering on the image of these subjects. In this problem, we compare with state-of-the-art subspace clustering algorithms assuming sparsity [53, 76] and low-rank-ness [75]. For each problem, we used PCA to project images in $9c$-dimensional subspaces to make an overcomplete dictionary. Table 3.2 shows the clustering accuracy for different numbers of subjects. The proposed method, SSC-SVG, shows a superior clustering performance compared to the existing algorithms based on the convex or nonconvex regularizers. SSC-OMP performs better than SSC-BP, SSC-SL0, and LRR on average, but it gives lower accuracy than ours for most cases. Especially, its performance collapses considerably when the number of clusters is larger than 5. SSC-SL0 shows the worst performance among the tested algorithms.

Table 3.2: Performance comparison on clustering accuracy (%) on the Extended Yale B dataset for face clustering.

No. clusters (c)	2	5	8	10	Average
LRR [75]	96.9	89.1	87.5	80.3	88.5
SSC-BP [53]	94.5	93.1	88.9	70.5	86.8
SSC-OMP [76]	98.4	**97.8**	81.1	82.9	90.5
SSC-SL0 [30]	98.4	75.6	66.2	53.4	73.4
SSC-SVG (Ours)	**99.2**	96.3	**95.7**	**90.3**	**95.4**

Motion segmentation. The goal of motion segmentation task is to segment trajectories of rigidly moving objects based on tracked points along the frames. Since collected trajectories from a rigid motion lie in a low-dimensional subspace, we can solve the motion segmentation as a subspace clustering problem [53]. Hence, we applied SSC-SVG to the well-known benchmark dataset, Hopkins 155 [55], which consists of 155 video sequences with two or three motion clusters. Four quantitative measures were used for clustering performance: mean, standard deviation (Std.), minimum, and median, following the work in [53]. The average performance of the algorithms are shown in Table 3.3. As shown in the table, our proposal outperforms existing algorithms approximating the l_0-norm and the dense representation method, LRR. SSC-BP and LRR give the similar performance, but they are unsatisfactory compared to ours. Two algorithms approximating the l_0-norm, SSC-OMP and SSC-SL0, show the disappointing results in this problem. Some graphical results on the dataset for four selected methods are illustrated in Figure 3.6.

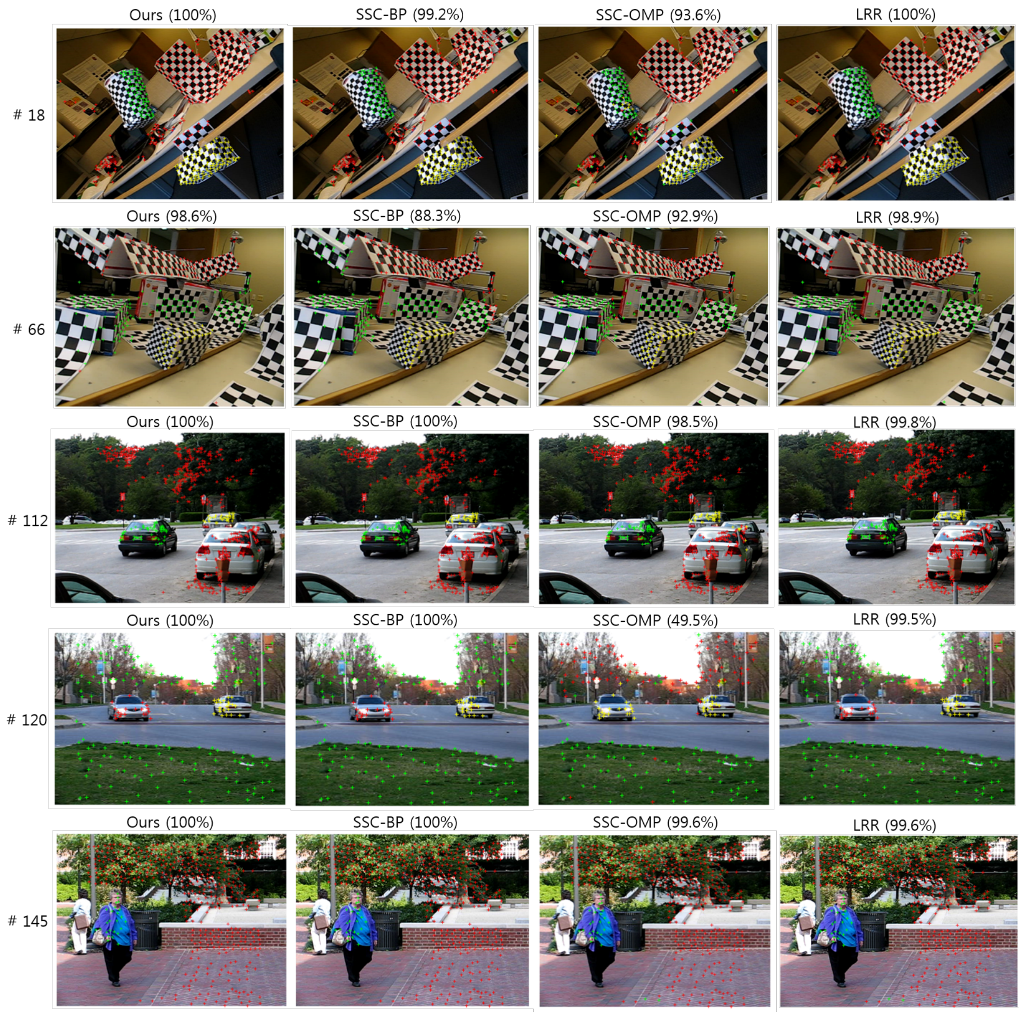


Figure 3.6: Motion segmentation results (snapshots) of five randomly chosen video sequences from the Hopkins 155 dataset by four methods: the proposed method, SSC-BP [53], SSC-OMP [76], and LRR [75]. Tracked points are marked by a symbol '+'. Different colors in the mark correspond to independent motion clusters. $(\cdot)$ denotes the segmentation accuracy. Best viewed in color (x2).

Table 3.3: Performance comparison with respect to clustering accuracy on the Hopkins 155 dataset for motion segmentation.

Algorithms	LRR	SSC-BP	SSC-OMP	SSC-SL0	SSC-SVG (Ours)
Mean	96.53	96.47	87.16	77.93	**97.31**
Std.	8.04	9.12	14.04	16.82	**7.25**
Median	99.72	**100**	93.10	80.82	**100**
Minimum	**58.19**	52.81	46.82	39.44	58.14

3.3.5 Parameter Analysis

The proposed measure has two parameters: the measure parameter ϵ and the balancing parameter λ. Following our analysis on the slowly vanishing gradient of the measure as shown in Figure 3.2, we can set the measure parameter ϵ to a small value (usually, it is recommended to have in the range of $[10^{-2}, 1]$). Nonetheless, we evaluate the impact of the parameter ϵ on the low-rank approximation problems using the Shark and Face data sets. Figure 3.7 gives the reconstruction error with variations of ϵ for the data sets. From the figure, we can observe that the proposed measure performs similarly with the choice of any value in the enough range of the parameter for each scenario, which confirms that our measure does not sensitive to the choice of the parameter value.

Now, we further report specific values of the parameters for all conducted experiments as shown in Table 3.4. Note that there is no specified λ in the formulation of the low-rank approximation problem, thus we do not report the value of the parameter for the problem. Since we have seen that the parameters are not sensitive to the choice of the values for data sets in each experimental subsection, we fix the two parameters for each subsection. Especially, we set the measure

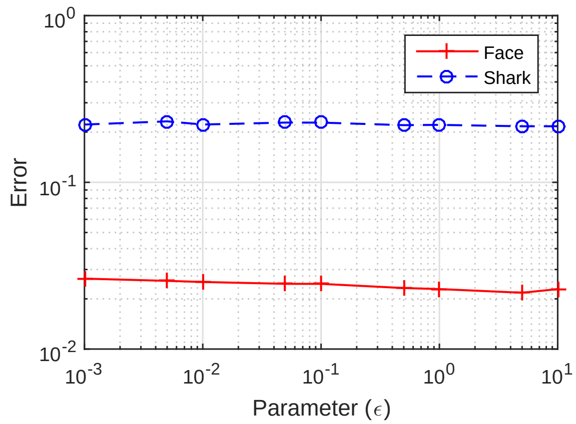


Figure 3.7: Reconstruction error with respect to values of the parameter ϵ for two data sets.

Table 3.4: Parameter values of (ϵ, λ) used in this work.

Parameter	ϵ	λ
Evaluation-Synthetic (Section 3.3.1)	0.05	0.05
Low-rank approximation (Section. 3.3.2)		–
Sparse coding (Section 3.3.3)		0.6
Subspace clustering (Section 3.3.4)		15

parameter ϵ to 0.05 throughout the experiments due to the empirical observations that it consistently gives satisfying performance with a fixed value for all tested problems. Since λ is a balancing parameter between the sparse regularizer and data term, it is natural to have different values according to independent problems.

3.4 Summary

In this chapter, we have analyzed desirable criteria to be a good nonconvex sparsity measure and presented a corresponding family of measures that are a solution of a differential equation, named slowly vanishing gradients (SVG). Among the SVG measures, we selected a practical one as a proposed measure, which complements both l_0- and l_1-norms from practical considerations. The penalty is a good alternative to the l_0-norm that possesses slowly vanishing gradient, which can be good for gradient-based optimization, and a simple proximity operator, which can be efficiently utilized in nonconvex optimizations. The proposed measure has been tested on various applications to demonstrate its effectiveness and empirical results have confirmed the superiority of the proposal.

Chapter 4

Robust Fixed Low-Rank Representations

This chapter describes several robust low-rank matrix approximation algorithms for an unstructured matrix and a structure matrix based on the robust l_1-norm. The motivation of the algorithms is derived from the fact that conventional low-rank approximation algorithms are neither robust to outliers nor efficient when handling real-world applications. We first propose a gradient descent based algorithm for an l_1 minimization problem, where the alternating rectified gradient method is suggested to solve the algorithm quickly. For better performance than the gradient-based algorithm which only consider the error measure, we introduce an efficient regularizer and an orthogonality constraint and the overall framework is solved using alternating minimization under the augmented Lagrangian framework. Since they assume a user-defined fixed-rank problem, we extend to handle rank uncertainty issue by proposing a rank estimation strategy for practical real-world problems. We also study a case where an observation matrix is structured, in which a robust kernel subspace learning algorithm based on the recently attracted rank minimization is devised to model trajectories of moving objects under noisy environments. The performance of the algorithms are demonstrated from several experiments on well-known real-world data sets.

4.1 The Alternating Rectified Gradient Method for l_1 Minimization

[1]In this section, we propose two alternating rectified gradient algorithms that solve the l_1-based factorization problems at significantly less running time and memory for large-scale problems. Even though the proposed methods are based on an alternating minimization method, they give fast convergence rates owing to the novel method of finding the update direction by a rectified representation based on matrix orthogonalization. These methods are derived from the observation that there are numerous projections and coefficient matrices that give the same multiplication result while the convergence speed depends largely on how these matrices are chosen. The methods proposed in this section are more efficient and robust than other l_1-norm based factorization and RPCA methods in solving various problems in Section 4.1.3.

4.1.1 l_1-ARG$_A$ as an approximation method

Gradient-based update

We first describe the problem of low-rank matrix approximation in the l_1-norm by an alternating gradient descent framework. The cost function for the low-rank

[1]This section is based on the paper appeared in *IEEE Transactions on Neural Networks and Learning Systems*: "Efficient l1-Norm-Based Low-Rank Matrix Approximations for Large-Scale Problems Using Alternating Rectified Gradient Method" [21].

matrix approximation is

$$\min_{P,X} \quad J(P,X) = \|Y - PX\|_1, \tag{4.1}$$

where $Y \in \mathbb{R}^{m\times n}$, $P \in \mathbb{R}^{m\times r}$, and $X \in \mathbb{R}^{r\times n}$ are the observation, projection, and coefficient matrices, respectively. Here, r is a predefined parameter and less than $\min(m,n)$. Since $|x|$ is not differentiable, we approximate $|x|$ by $\lim_{\epsilon\to 0}\sqrt{x^2+\epsilon^2}$. Then we approximate the derivative of $|x|$ using the derivative of $\lim_{\epsilon\to 0}\sqrt{x^2+\epsilon^2}$ as follows:

$$\frac{d|x|}{dx} \approx \lim_{\varepsilon\to 0}\frac{\partial\sqrt{x^2+\varepsilon^2}}{\partial x} = \lim_{\varepsilon\to 0}\frac{x}{\sqrt{x^2+\varepsilon^2}} = \operatorname{sgn}(x), \tag{4.2}$$

where $\operatorname{sgn}(x)$ is the signum function of x and the approximation is exact except at $x=0$. In this way, we can differentiate (4.1) with respect to (w.r.t.) X and find that its derivative is

$$\nabla_X J(P,X) = -P^T \operatorname{sgn}(Y - PX). \tag{4.3}$$

Here, $\operatorname{sgn}(Y)$ for matrix Y represents a matrix whose (i,j)-th element is $\operatorname{sgn}(y_{ij})$.

Now, we consider the problem of finding an optimal step size $\alpha > 0$ to update X by the steepest gradient descent method.

$$\begin{aligned}\min_{\alpha} J(\alpha|P,X,\nabla_X J) &= \|Y - P(X - \alpha\nabla_X J(P,X))\|_1 \\ &= \|Y' - \alpha PP^T \operatorname{sgn}(Y')\|_1 \\ &= \|Y' - \alpha A\|_1,\end{aligned} \tag{4.4}$$

where $Y' = Y - PX$ and $A = PP^T \operatorname{sgn}(Y')$. We apply the weighted median algorithm to the ratio y'_{ij}/a_{ij} with weight $|a_{ij}|$ to get the step size α that minimizes the cost function (4.4). Note that in this algorithm, we apply the weighted median algorithm to update either P or X at a time, to reduce the total computation time and this is different from [10], where the algorithm is applied columnwise.

Finally, Y' and X are updated as

$$\begin{aligned} Y' &\leftarrow Y' - \alpha P P^T \operatorname{sgn}(Y'), \\ X &\leftarrow X + \alpha P^T \operatorname{sgn}(Y'). \end{aligned} \tag{4.5}$$

For P, we can also differentiate (4.1) w.r.t. P in the same manner as

$$\nabla_P J(P, X) = -\operatorname{sgn}(Y - PX)X^T. \tag{4.6}$$

The projection and coefficient matrices P and X are updated alternatingly until convergence is achieved.

However, a serious issue arises in this updating procedure, because there are numerous pairs of P and X that give the same multiplication result of PX. To see this, let us reexamine the minimization problem (4.1). If $P' = PH^{-1}$ and $X' = HX$ for some nonsingular matrix $H \in \mathbb{R}^{r \times r}$, then

$$\min_{P', X'} \quad J(X', P') = \|Y - P'X'\|_1 = \|Y - PX\|_1. \tag{4.7}$$

Accordingly, the step-size problem for X' can be written as

$$\min_{\beta} J(\beta | P', X', \nabla_{X'} J) = \|Y' - \beta P' P'^T \operatorname{sgn}(Y')\|_1, \tag{4.8}$$

where β is a step size. When H is orthogonal, (4.4) and (4.8) are the same because of the relation $P'P'^T = PH^{-1}H^{-T}P^T = PH^THP^T = PP^T$. If it is not the case, then the update direction of (4.8) changes depending on H, i.e.,

$$PP^T \operatorname{sgn}(Y') \neq P'P'^T \operatorname{sgn}(Y'). \tag{4.9}$$

This means that the update direction depends on the choice of P and X. Therefore, it is important to find P and X that will give a good update direction for fast convergence.

Finding an optimal direction for alternating updates

In the previous subsection, we have shown that the update direction depends on the representation of P and X, which can influence the convergence rate. This happens because P and X are the intermediate variables of the following basic problem:

$$\begin{aligned} \min_{G} \quad & \|Y - G\|_1 \\ s.t. \quad & G \in \mathbb{R}_r^{m\times n}, \end{aligned} \tag{4.10}$$

where $\mathbb{R}_r^{m\times n}$ is a set of $m \times n$ matrices with rank r. However, this problem is difficult to solve directly because $\mathbb{R}_r^{m\times n}$ is not convex. This is why it is common to use alternating updates based on intermediate variables like P and X for low-rank matrix approximation. In summary, it is difficult to solve the problem (4.10), while the less difficult problem (4.1) can still lead to a slow convergence because of the ambiguity of the update direction.

Then, how do we compromise? To answer this question, notice that the gradient w.r.t. X can also be expressed as the solution to the following problem:

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X) = \|Y - P(X + \Delta X')\|_1 \\ s.t. \quad & \|\Delta X'\|_F^2 = \breve{\epsilon}^2, \end{aligned} \tag{4.11}$$

where $\Delta X'$ is the variation of X that we are seeking and $\breve{\epsilon} \ll 1$. This problem is to minimize the directional derivative of the cost function w.r.t. $\Delta X'$ and the optimal $\Delta X'$ is the same as $\nabla_X J$ up to scale if $\breve{\epsilon} \to 0$. To avoid the ambiguity in representing P and X, and to convert the problem as if it were to be solved for $G \in R_r^{m\times n}$ in the basic problem, we modify the constraint as

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X) = \|Y - P(X + \Delta X')\|_1 \\ & \qquad \triangleq \|Y' - \Delta G'\|_1 \\ s.t. \quad & \|\Delta G'\|_F^2 \triangleq \|P\Delta X'\|_F^2 = \epsilon^2. \end{aligned} \tag{4.12}$$

In this modified problem, we search the update direction for X, but the new constraint limits the search domain with respect to $\Delta G' = P\Delta X'$, the update of G, instead of $\Delta X'$. In this manner, we can preserve the convexity of the search domain while avoiding the difficulty that arises from the ambiguity in representing P and X.

By introducing a Lagrange multiplier to (4.12), the resulting Lagrangian is

$$\|Y' - P\Delta X'\|_1 + \frac{\lambda}{2}(\mathrm{tr}(\Delta X'^T P^T P \Delta X') - \epsilon^2), \tag{4.13}$$

where tr is the trace operator ($\|A\|_F^2 = \mathrm{tr}(A^T A)$). Differentiating (4.13) w.r.t. $\Delta X'$ and equating it to zero, we obtain

$$-P^T \operatorname{sgn}(Y' - P\Delta X') + \lambda P^T P \Delta X' = 0,$$

which gives

$$\Delta X' = \frac{1}{\lambda} P^+ \operatorname{sgn}(Y' - P\Delta X'), \tag{4.14}$$

where $P^+ = (P^T P)^{-1} P^T$ is the left pseudo-inverse of P. By applying (4.14) to $\|P\Delta X'\|_F^2 = \epsilon^2$, we get

$$\frac{1}{\lambda} = \frac{\epsilon}{\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F}, \tag{4.15}$$

and finally

$$\Delta X' = \frac{P^+ \operatorname{sgn}(Y' - P\Delta X')}{\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F} \cdot \epsilon. \tag{4.16}$$

For an infinitesimal ϵ, the update direction becomes

$$\begin{aligned} \lim_{\epsilon \to 0} \Delta X' &\propto \lim_{\epsilon \to 0} P^+ \operatorname{sgn}(Y' - P\Delta X') \\ &= P^+ \operatorname{sgn}(Y') \triangleq \Delta X. \end{aligned} \tag{4.17}$$

Note $\lim_{\epsilon \to 0} \operatorname{sgn}(Y' - P\Delta X') = \operatorname{sgn}(Y')$ in (4.14) because $\lim_{\epsilon \to 0} \Delta X' = 0$ in (4.16) and we regard sgn as a limit of a smooth function as defined in (4.2). Here, we ignore $\|PP^+ \operatorname{sgn}(Y' - P\Delta X')\|_F$ in (4.16) because we are interested only in the direction, which is denoted as ΔX, and the step size for the update will be found next. Note that the update direction of the low-rank approximation is given as

$$\Delta G \triangleq P\Delta X = PP^+ \operatorname{sgn}(Y'), \tag{4.18}$$

and this does not change depending on the representation of P and X, i.e., there is no ambiguity in ΔG unlike $PP^T \operatorname{sgn}(Y')$ in (4.9). With the new update direction ΔG, we revise the step-size problem (4.4) as the following:

$$\min_{\alpha} \|Y' - \alpha\Delta G\|_1 = \|Y' - \alpha PP^+ \operatorname{sgn}(Y')\|_1, \tag{4.19}$$

where α is determined by the weighted median technique. For updating P, we can obtain ΔP in the same manner under the constraint ($\|\Delta P' X\|_F^2 = \epsilon^2$) as

$$\Delta P = \operatorname{sgn}(Y')X^+, \tag{4.20}$$

and find the optimal step size as in (4.19).

There is an observation to be made on this updating rule. This new update direction is analogous to the Gauss-Newton update direction in the least-squares problem. The Gauss-Newton direction of $\|F(x)\|_F^2$ is given as $-\nabla_x F(x)^+ F(x)$. If we regard $F(x)$ as a result of $\frac{\partial \|F(x)\|_F^2}{\partial F(x)}$ ignoring its scale, then it is similar to the expression $\Delta X = P^+ \operatorname{sgn}(Y')$. Hence, we may consider this update direction as an extension of the Gauss-Newton method to l_1-norm problems and expect it to be better than the normal gradient direction.

Note that this procedure is equivalent to changing the representation of P and X so that the fixed matrix, either P or X, is orthonormal. This means that the step size problem (4.8) of the normal gradient method becomes the same as (4.19) when P and X are chosen so that P is orthogonal. We can easily find such an orthogonal matrix using the QR decomposition. We call this as the rectified representation. Hence, it is better to use ordinary gradient descent in conjunction with this representation change, which is faster than calculating a pseudo-inverse.

Summary of the proposed algorithm

First, we update P while X is fixed in (4.1). To make X orthonormal, we apply QR decomposition to X^T:

$$\begin{aligned} X^T &= X'^T R, \\ PX &= PR^T X' = P'X', \end{aligned} \tag{4.21}$$

where orthogonal matrix X'^T and upper triangular matrix R are obtained from QR decomposition, and $P' = PR^T$. Then, we can compute ΔP by using X' and find the optimal step size using the weighted median algorithm.

Once the update of P is finished, we update X with P fixed. Again, we apply QR decomposition to P to change the representation. The update rule is similar to that of the P update. Then, we continue to update P and X alternatingly; the overall procedure is described in Algorithm 1. We call the method as l_1-norm-based alternating rectified gradient method based on approximation, l_1-ARG$_A$, because it find the gradient by approximated manner. In the algorithm, P and X are rectified by the QR decomposition at line 8 and 14, respectively.

To deal with numerical errors, we modify the signum function as:

$$\mathrm{sgn}'(x) = \begin{cases} 1 & x \geq \gamma, \\ 0 & -\gamma < x < \gamma, \\ -1 & x \leq -\gamma, \end{cases} \tag{4.22}$$

where γ is a threshold with a small positive value. Using this modified function, we can find a better solution despite the difficulties that numerical errors might create.

Algorithm 1 l_1-norm-based matrix approximation using the approximated alternating rectified gradient method (l_1-ARG$_A$)

Input: $Y \in \mathbb{R}^{m\times n}$, the subspace dimension r
Output: $P \in \mathbb{R}^{m\times r}, X \in \mathbb{R}^{r\times n}$
Initialize P to a zero matrix and X randomly
$Y' \leftarrow Y$
while residual Y' does not converge **do**
    ## P update (Fix X, update P)
    while residual Y' does not converge **do**
        $X'^T R \leftarrow X^T$, $P' \leftarrow PR^T$
        $\Delta P \leftarrow \mathrm{sgn}'(Y')X'^T$
        $(Y', P') \leftarrow \mathrm{Update}(Y', P', X', \Delta P)$
    end while
    ## X update (Fix P, update X)
    while residual Y' does not converge **do**
        $PR \leftarrow P'$, $X \leftarrow RX'$
        $\Delta X \leftarrow P^T\,\mathrm{sgn}'(Y')$
        $(Y'^T, X'^T) \leftarrow \mathrm{Update}(Y'^T, X^T, P^T, \Delta X^T)$
    end while
end while

In Algorithm 1, the update of either P or X is repeated until convergence, and then the roles of the matrices are switched. Even though the algorithm can work by just alternating the updates of P and X one by one, the present approach gave us better performance in some of the experiments, such as the nonrigid motion estimation in Section 4.1.3. This is not exactly an "alternating" update, but we

Algorithm 2 Function: Update (Y, U, V, Z)

1: Input: Y, U, V, Z: matrices

2: Output: T, R: matrices

3: ## Line-search (by weighted median)

4: $\alpha \leftarrow \arg\min_{\alpha} \|Y - \alpha ZV\|_1$

5: $T \leftarrow Y - \alpha ZV$

6: $R \leftarrow U + \alpha Z$

still call it alternating rectified gradient method. The projection and coefficient matrices are updated by line-search technique using the weighted median method in Algorithm 2.

As mentioned earlier, the step size α is determined by using the weighted median algorithm. For the weighted median algorithm, we may use a divide and conquer algorithm such as quick-select [79, 80], which can find the solution in linear time on average. However, in practice, it is faster to use existing sorting functions when the number of elements is not large. Moreover, since we are applying the weighted median algorithm to find the step size, which does not need to be accurate, it is better to calculate the weighted median of randomly selected samples, when the number of samples is large. To see how the weighted median depends on the number of samples, we consider the problem of finding an approximate weighted median from a set consisting of an infinite number of elements. To simplify the problem, we assume that elements have the same weights. Then the cumulative probability $F(q; 2d+1)$ that the sample median of $2d+1$ samples is less than the $(100 \times q)\%$ quantile of original elements is equal to the cumulative probability that the success is no more than d for a binomial distribution $B(2d+1, 1-q)$. Since the cumulative distribution function of a binomial distribution can be represented in terms of the regularized incomplete beta function, the result is given as

$$F(q; 2d+1) = P(Z \leq d) = I_q(d+1, d+1), \tag{4.23}$$

where Z is the binomial random variable and I_q is the regularized incomplete beta function. This expression can be calculated numerically, and we have found that

$$F(1/2 + 0.005; 10^5 + 1) - F(1/2 - 0.005; 10^5 + 1) \approx 0.998.$$

This means that if we use 10^5 samples, then the sample median resides within the $\pm 0.5\%$ range of the true median with probability 0.998. Even if this result applies for the case of equally weighted samples, the result is also meaningful for the weighted median if the weights are moderately distributed. This is a valid assumption because ΔG, which is an orthogonal projection of $\operatorname{sgn}'(Y')$, is bounded by $\| \operatorname{sgn}'(Y') \|_F$. In experiments, we randomly selected 10^5 samples if the number of elements is greater than 10^5, and then applied an existing sorting function to find the weighted median. There is a small chance that the weighted median technique may not reduce the cost function due to random sampling, but the problem can be resolved by a slight tweak in the algorithm, such as repeating the random sampling until it reduces the cost function.

The downside of the proposed algorithm is the difficulty of guaranteeing whether $P^{(t)}$ and $X^{(t)}$ will converge to a local minimum, due primarily to the assumption that the derivative of $|x|$ is $\operatorname{sgn}(x)$, which is in fact not differentiable at 0. Hence, there is a possibility that the algorithm may find an update direction that does not decrease the cost function when many of the elements of Y' are zero, even though it is not a local minimum. In that case, the step size will be zero and the algorithm will be terminated. Nonetheless, if this happens, it will be near a local minimum since many of the residual elements are zero. Besides, there is usually some Gaussian noise in Y for practical problems, which prevent many of the residual elements from being zero at the same time. Therefore, the proposed algorithm will work well in practical problems and we verify the convergence using real world problems in Section 4.1.3.

Weighted method of l_1-ARG$_A$ with missing data

In real applications, there are not only outliers but also missing data, which can have a negative effect on vision and recognition systems. We solve the problem of low-rank matrix approximation using the l_1-norm in the presence of missing data which is also known as a matrix completion (MC) problem by extending the result from the previous subsection.

The problem can be formulated as

$$\min_{P,X} \quad J(P, X|W) = \|W \odot (Y - PX)\|_1, \tag{4.24}$$

where $\odot$ is the component-wise multiplication or Hadamard product. Here, $W \in \mathbb{R}^{m\times n}$ is a weight matrix, whose element w_{ij} is 1 if y_{ij} is known, and is 0 if y_{ij} is unknown. Similar to the problem (4.12), we can formulate the weighted low-rank matrix factorization in the l_1-norm under the constraint $\|P\Delta X'\|_F^2 = \epsilon^2$ as

$$\begin{aligned} \min_{\Delta X'} \quad & J(\Delta X'|P, X, W) = \|(W \odot (Y' - P\Delta X'))\|_1, \\ s.t. \quad & \|P\Delta X'\|_F^2 = \epsilon^2. \end{aligned} \tag{4.25}$$

Similarly as in Section 4.1.1, the solution to this problem can be represented in vector form as

$$\begin{aligned} \mathrm{vec}(\Delta X) &= (I \otimes P^{+})\overline{W}\mathrm{vec}(\mathrm{sgn}(W \odot Y')) \\ &= (I \otimes P^{+})\mathrm{vec}(W \odot \mathrm{sgn}(W \odot Y')), \end{aligned} \tag{4.26}$$

where $\otimes$ is the Kronecker product, $\overline{W} = \mathrm{diag}(\overline{\boldsymbol{w}}) \in \mathbb{R}^{mn\times mn}$, $\overline{\boldsymbol{w}} = (\boldsymbol{w}_1^T, \boldsymbol{w}_2^T, ..., \boldsymbol{w}_n^T)^T \in \mathbb{R}^{mn\times 1}$, $\boldsymbol{w}_i$ is the i-th column vector of W, and I denotes an $n \times n$ identity matrix.

Because the elements of W are either 0 or 1, (4.26) can be rewritten as

$$\begin{aligned} \mathrm{vec}(\Delta X) &= (I \otimes P^{+})\mathrm{vec}(\mathrm{sgn}(W \odot Y')) \\ &= \mathrm{vec}(P^{+}\,\mathrm{sgn}(W \odot Y')), \end{aligned} \tag{4.27}$$

and this gives

$$\Delta X = P^{+}\,\mathrm{sgn}(W \odot Y'). \tag{4.28}$$

Similar to (4.19), the cost function to find the step size α becomes

$$\begin{aligned} \min_{\alpha} J(\alpha|P, X, W, \Delta X) &= \min_{\alpha} \|W \odot (Y' - \alpha P \Delta X)\|_1 \\ &= \min_{\alpha} \|W \odot Y' - \alpha W \odot (P P^{+}\,\mathrm{sgn}(W \odot Y'))\|_1. \end{aligned} \tag{4.29}$$

Compared to (4.19), the only difference is the presence of W in the cost function.

When we vary P for a fixed X, we can obtain ΔP and the cost function to find the optimal step size similarly.

$$\Delta P = \mathrm{sgn}(W \odot Y')X^{+}, \tag{4.30}$$

$$\begin{aligned} \min_{\alpha'} J(\alpha'|P, X, W, \Delta P) &= \min_{\alpha'} \|W \odot (Y' - \alpha' \Delta P X)\|_1 \\ &= \min_{\alpha'} \|W \odot Y' - \alpha' W \odot (\mathrm{sgn}(W \odot Y')X^{+}X)\|_1. \end{aligned} \tag{4.31}$$

The step sizes in (4.29) and (4.31) can also be solved by the weighted median algorithm.

4.1.2 l_1-ARG$_D$ as a dual method

l_1-ARG$_D$ in the presence of outliers

In this section, we propose a second novel method to find a proper descending direction without the gradient approximation of ΔX. Since it is difficult to guarantee that l_1-ARG$_A$ converges to a local minimum, we propose the second novel method with a convergence guarantee. We refer to the algorithm as a l_1-norm-based alternating rectified gradient method using the dual problem, l_1-ARG$_D$. As mentioned earlier, the problem to find the gradient of X for a fixed P in low-rank matrix approximation is formulated as

$$\begin{aligned} \min_{P,X} \quad & \|Y' - P\Delta X\|_1 \\ s.t. \quad & ||P\Delta X||_F^2 = \epsilon^2. \end{aligned} \tag{4.32}$$

We reformulate (4.32) to an unconstrained problem as

$$\min_{\Delta X} \quad f_\eta(X, \Delta X) \triangleq ||Y' - P\Delta X||_1 + \frac{1}{2\eta}||P\Delta X||_F^2, \tag{4.33}$$

where $\eta > 0$ is a weight parameter. Here, we assume that P is orthonormalized using the QR decomposition, i.e., $||P\Delta X||_F^2 = ||\Delta X||_F^2$.

We can obtain the Lagrangian of (4.33) by substituting $||Y' - P\Delta X||_1$ to Z as

$$\begin{aligned} \mathcal{L}(\Delta X, \Lambda, M) = \mathbf{1}^T Z \mathbf{1} + \frac{1}{2\eta}||\Delta X||_F^2 \\ + \operatorname{tr}(\Lambda^T(Y' - P\Delta X - Z)) + \operatorname{tr}(M^T(-Y + P\Delta X - Z)), \end{aligned} \tag{4.34}$$

where $\mathbf{1} \in \mathbb{R}^m$ and $\Lambda, M \leq 0$ are Lagrange multipliers. By taking a derivative of (4.34) and solving for Z and ΔX at a stationary point, we can obtain $\mathbf{1}\mathbf{1}^T - \Lambda - M = 0$ and $\Delta X = \eta P^T(\Lambda - M) = \eta P^T \widetilde{V}$, respectively, where $\widetilde{V} \triangleq \Lambda - M$ and $-1 \leq \widetilde{v}_{ij} \leq 1$ for all elements of $\widetilde{V}$. Therefore, (4.34) can be reformulated as

$$\begin{aligned} & \frac{1}{2\eta}||\Delta X||_F^2 + \operatorname{tr}((\widetilde{V})^T(Y' - P\Delta X)), \\ & s.t. \quad -1 \leq \widetilde{v}_{ij} \leq 1. \end{aligned} \tag{4.35}$$

Hence, the dual problem of (4.33) is constructed by using the corresponding primal solution $\Delta X = \eta P^T \widetilde{V}$ and $\eta \widetilde{V} = V$ as

$$\begin{aligned} \max_{V} \quad & g_\eta(V) \triangleq \frac{1}{\eta}\operatorname{tr}(V^T Y') - \frac{1}{2\eta}||P^T V||_F^2, \\ s.t. \quad & -\eta \leq v_{ij} \leq \eta. \end{aligned} \tag{4.36}$$

We use the proximal gradient technique [81] to solve this problem. We convert the sign of (4.36) and reformulate it as an unconstrained problem

$$\min_{V} \quad -\frac{1}{\eta}\operatorname{tr}(V^T Y') + \frac{1}{2\eta}||P^T V||_F^2 + I_\eta(V), \tag{4.37}$$

where $I_\eta(V)$ is the indicator function for each element of matrix V

$$I_\eta(v_{ij}) = \begin{cases} 0 & -\eta \leq v_{ij} \leq \eta, \\ \infty & \text{else.} \end{cases} \tag{4.38}$$

Denoting U as the V in the previous step, the proximal approximation [81] of (4.37) is given as

$$\begin{aligned} &\frac{1}{\eta}\operatorname{tr}((V-U)^T(-Y' + PP^T U)) + \frac{L}{2\eta}||V-U||_F^2 \\ &+ \frac{1}{2\eta}||P^T U||_F^2 - \frac{1}{\eta}\operatorname{tr}(U^T Y') + I_\eta(V), \end{aligned} \tag{4.39}$$

where L is the Lipschitz constant of (4.37) and is 1 in this case because P is orthogonal.

The above equation can be simplified as

$$\frac{1}{2\eta}||V - U - Y' + PP^T U||_F^2 + I_\eta(V) + \text{constant}, \tag{4.40}$$

and this gives the following result

$$V = \begin{cases} \eta & V' > \eta, \\ V' & -\eta < V' < \eta, \\ -\eta & V' < -\eta, \end{cases} \tag{4.41}$$

where

$$V' = Y' + U - PP^T U. \tag{4.42}$$

Since this iterative process itself can take a non-ignorable amount of time, we perform the iteration just enough to find a good descending direction, rather than calculating the exact optimal solution. We update the solution V and corresponding primal solution $\Delta X = P^T V$ until the ratio between the difference of the previous and current primal cost values and the difference of the previous primal and current dual cost values is no less than a positive scalar $0 < \beta \leq 1$ as

$$\frac{f_\eta(X, \Delta X_k) - f_\eta(X, \Delta X_{k+1})}{f_\eta(X, \Delta X_k) - g_\eta(V_{k+1})} \geq \beta. \tag{4.43}$$

Let $\Delta X^* = \arg\min_{\Delta X} f_\eta(X, \Delta X)$, then we obtain the following relation

$$\begin{aligned} & f_\eta(X, 0) - f_\eta(X, \Delta X) \geq \beta(f_\eta(X, 0) - g_\eta(V)) \\ & \geq \beta(f_\eta(X, 0) - f_\eta(X, \Delta X^*)). \end{aligned} \tag{4.44}$$

Note that during the proximal optimization, $g_\eta(V_{k+1})$ is always not larger than $f_\eta(X, \Delta X_{k+1})$. After finding a solution that satisfies (4.43), we apply the weighted median method as an exact line-search[2] to find the optimal step size of the gradient. The overall procedure is described in Algorithm 3. In the algorithm, η is decreased during the iteration and is bounded by $0 < \eta_{\min} \leq \eta \leq \eta_{\max} < \infty$ where $\eta_{\min}$ and $\eta_{\max}$ are predefined constants. P and X are rectified by the QR decomposition at line 7 and 11 in the algorithm, respectively. We find the gradient of P or X by Algorithm 4.

The main difference between the two proposed methods is that we can formally guarantee that l_1-ARG$_D$ converges to a subspace-wise local minimum (see Section 4.1.2), whereas a local minimum is not guaranteed for l_1-ARG$_A$ due to the approximation of the l_1 cost function. Although both algorithms may reach similar cost values, they can find different solutions as shown in Section 4.1.3.

[2]Here, we assume that an exact line-search is performed in order to simplify the proof in the below.

Algorithm 3 l_1-norm-based matrix approximation using the exact alternating rectified gradient method (l_1-ARG$_D$)

Input: $Y \in \mathbb{R}^{m\times n}$, low-rank r, $\beta = 10^{-4}$, $\eta_{\min} = 10^{-6}$
Output: $P \in \mathbb{R}^{m\times r}, X \in \mathbb{R}^{r\times n}$
Initialize P to a zero matrix and X randomly, $\eta = \infty$
$Y' \leftarrow Y$
while residual Y' does not converge **do**
    \# P update (Fix X, update P)
    $X'^T R \leftarrow X^T$, $P' \leftarrow PR^T$
    $\Delta P^T \leftarrow \text{findGradient}(X'^T, P^T, Y'^T, V^T, \eta, \eta_{\min}, \beta)$
    $(Y', P') \leftarrow \text{Update}(Y', P', X, \Delta P)$
    \# X update (Fix P, update X)
    $PR \leftarrow P'$, $X \leftarrow RX'$
    $\Delta X \leftarrow \text{findGradient}(P, X, Y', V, \eta, \beta)$
    $(Y'^T, X^T) \leftarrow \text{Update}(Y'^T, X^T, P^T, \Delta X^T)$
end while

Algorithm 4 Function: findGradient $(K, L, Y, V, \eta, \eta_{\min}, \beta)$

Input: K, L, Y, and V: matrices; η, $\eta_{\min}$, β: scalars

Output: ΔS: a matrix

Description:

$\eta \leftarrow \max(\min(\eta, ||Y||_1/mn), \eta_{\min})$, $k = 1$, $V_0 = 0$

$f_\eta(K, \Delta K_0) = f_\eta(K, \Delta K_1) = ||Y||_1$, $g_\eta(V_1) = 0$

while $\frac{f_\eta(K,\Delta K_{k-1}) - f_\eta(K,\Delta K_k)}{f_\eta(K,\Delta K_{k-1}) - g_\eta(V_k)} < \beta$ **do**

  $\eta \leftarrow \max(\frac{\eta}{2}, \eta_{\min})$

  $V_k \leftarrow Y + V_{k-1} - V_{k-1}L^T L$ and by (4.41)

  $\Delta K_k \leftarrow V_k L^T$

  $f_\eta(K, \Delta K_{k+1}) \leftarrow ||Y - \Delta K_k L||_1 + \frac{1}{2\eta}||\Delta K_k||_F^2$

  $g_\eta(V_{k+1}) \leftarrow \mathrm{tr}(Y^T V_k) - \frac{1}{2\eta}||\Delta K_k||_F^2$

  $k \leftarrow k + 1$

end while

$\Delta S \leftarrow \Delta K_{k-1}$

Proof of convergence

Regardless of the initial point, the proposed method, l_1-ARG$_D$, which is a descent algorithm, converges to a *subspace-wise* local minimum according to the Zangwill's global convergence theorem [82, 83]. Subspace-wise local minimum is defined as follows:

Definition 2 (Subspace-wise local minimum)**.** *Let the cost function of l_1-ARG$_D$ be $J(P, X) \triangleq ||Y - PX||_1$. If there is no ΔX or ΔP such that $||Y - P(X + \Delta X)||_1 < ||Y - PX||_1$ or $||Y - (P + \Delta P)X||_1 < ||Y - PX||_1$, then (P, X) is a subspace-wise local minimum.*

A local minimum is a subspace-wise local minimum. If a cost function is smooth, a subspace-wise local minimum is also a local minimum [83]. However, the cost function (4.1) is not smooth, and consequently, a subspace-wise local minimum may not be a local minimum. Nonetheless, it is worth finding a subspace-wise local minimum because a subspace-wise local minimum is a necessary condition to be a local minimum. It also minimizes the cost function as well as the other state-of-the-art methods in the experiments of Section 4.1.3.

Let us denote $A : (\mathcal{P}, \mathcal{X}) \to (\mathcal{P}, \mathcal{X})$ as a point-to-set mapping [82, 83] that describes the behavior of l_1-ARG$_D$, where $\mathcal{P}$, and $\mathcal{X}$ are the domains of P and X, respectively. According to the Zangwill's theorem, a descent algorithm is globally convergent under the following three conditions (converges to a subspace-wise local minimum irrespective of the initial point).

1. All (P_k, X_k) should be contained in a compact set.
2. For cost function $J(P, X) = ||Y - PX||_1$,
   (a) if (P, X) is not in the solution set consisting of subspace-wise local

minimums, $J(P', X') < J(P, X)$ for all $(P', X') \in A(P, X)$.

(b) if (P, X) is in the solution set, $J(P', X') \leq J(P, X)$ for all $(P', X') \in A(P, X)$.

3. Mapping A is closed at points that are not subspace-wise local minimum.

Theorem 1. *l_1-ARG_D converges to a subspace-wise local minimum irrespective of the initial point under the three conditions.*

Proof. See Appendix D □

The local convergence rate is hard to find, but we show empirically that l_1-ARG$_D$ gives fast convergence in Section 4.1.3. Table 4.1 shows the comparison between the proposed methods with and without applying rectification (QR decomposition) for three reconstruction problems with 5% outliers over 10 independent runs. As shown in the table, the methods using rectification take much shorter execution time and need less number of iterations, and give lower reconstruction error.

Weighted method of l_1-ARG$_D$ with missing data

The proposed method, l_1-ARG$_D$, can be applied to real application problems in the presence of missing data. We solve the problem of low-rank matrix approximation using the l_1-norm by extending the proposed method as a weighted low-rank approximation problem.

The problem can be formulated as

$$||W \odot (Y' - P\Delta X)||_1 + \frac{1}{2\eta}||P\Delta X||_F^2, \tag{4.45}$$

where η is a small positive constant. We assume that P is orthonormalized by the QR decomposition.

Table 4.1: Performance of the proposed methods with/without applying rectification

	Algorithm	l_1-ARG$_A$	l_1-ARG$_A$(no QR)	l_1-ARG$_D$	l_1-ARG$_D$(no QR)
	Error	0.867	0.870	0.868	0.870
m=n=500, r=40	Time (sec)	1.1 ± 0.0	6.5 ± 0.7	0.7 ± 0.2	157.8 ± 25.1
	Iterations	6 ± 0.0	36.7 ± 3.7	19.2 ± 4.5	365.1 ± 45.1
	Error	0.869	0.871	0.869	0.871
m=n=1000, r=80	Time (sec)	2.8 ± 0.1	16.6 ± 0.7	1.8 ± 0.1	970.3 ± 60.0
	Iterations	6.1 ± 0.3	36.4 ± 3.9	15.2 ± 0.9	358.1 ± 16.4
	Error	0.869	0.872	0.869	0.871
m=n=2000, r=160	Time (sec)	10.0 ± 0.1	60.2 ± 1.1	7.6 ± 0.5	5526.7 ± 240.5
	Iterations	6 ± 0.0	35.6 ± 1.5	15.3 ± 1.6	365.2 ± 13.7

The dual problem of (4.45) is constructed in the similar fashion as in the previous section

$$\begin{aligned} \max_{V} \quad & \frac{1}{\eta}\operatorname{tr}((W \odot V)^T Y') - \frac{1}{2\eta}||P^T(W \odot V)||_F^2, \\ s.t. \quad & -\eta \le V_{ij} \le \eta, \end{aligned} \tag{4.46}$$

and this gives the following unconstrained minimization problem as a proximal mapping operator

$$\min_{V} \quad \frac{1}{2\eta}||P^T(W \odot V)||_F^2 - \frac{1}{\eta}\operatorname{tr}((W \odot V)^T Y') + I_\eta(V), \tag{4.47}$$

where $I_\eta(V)$ is an indicator function. Now, we consider the following approximation of (4.47):

$$\begin{aligned} &\frac{1}{\eta}\operatorname{tr}((V-U)^T[-W \odot Y' + W \odot (PP^T(W \odot U))]) \\ &\qquad + \frac{L}{2\eta}||V-U||_F^2 + I_\eta(V) + \text{constant}, \end{aligned} \tag{4.48}$$

where L is the Lipshitz constant ($L = 1$). Then this can be reformulated as

$$\frac{1}{2\eta}||V - U - W \odot Y' + W \odot (PP^T(W \odot U)||_F^2 + I_\eta(V) + \text{constant}. \tag{4.49}$$

Therefore, we obtain the result in the same form as (4.41) with $V' = U + Y' + W \odot (PP^T U)$.

4.1.3 Experimental results

We evaluated the performance of the proposed methods (l_1-ARG$_A$ and l_1-ARG$_D$) by experimenting with various data. We compared the proposed algorithms to other methods (IALM and EALM [43], ALADM [11], l_1-AQP [10], Regl_1-ALM [49]) in terms of the reconstruction error and execution time. The initial projection and coefficient matrices were set to zero and Gaussian random numbers, respectively, for the proposed methods and l_1-AQP. All the elements of the weight matrix for Regl_1-ALM was set to 1 for non-weighted factorization problems. In addition, the weighted median method used in the proposed methods was implemented as described in Section 4.1.1. We set $\rho = 10^{-5}$ in the stopping condition and γ as the same as ρ for all of the proposed methods. The trace-norm regularizer of Regl_1-ALM was set to 20, which gave the best performance in the experiments, if not stated otherwise.

We also performed experiments with missing data using the weighted version of the proposed methods (Wl_1-ARG$_A$ and Wl_1-ARG$_D$) in Section 4.1.1 and Section 4.1.3, and the performances were compared to those of other methods that can handle missing data (ALADM-MC which is a weighted version of ALADM [11], Regl_1-ALM [49]). We did not evaluate the methods l_1-AQP [10] for large-scale data because of its heavy computational complexity and memory requirement. We set the parameters of ALADM and ALADM-MC as described in [11], and all of the parameters of the proposed methods were the same as those of non-weighted versions. To show the usefulness of the proposed algorithm, we also applied the proposed methods to the non-rigid structure from motion problem [49]. All experiments were conducted using MATLAB on a computer with 8GB RAM and a 3.4GHz quad-core CPU.

Synthetic data with outliers

Firstly, we applied the proposed methods to synthetic examples with outliers. We generated an $(m \times r)$ matrix B and an $(r \times n)$ matrix C whose elements were uniformly distributed in the range $[-1, 1]$. We also generated an $(m \times n)$ noise matrix N whose elements had the Gaussian distribution with zero mean and variance of 0.01. Based on $Y_0 = BC + N$, we constructed the observation matrix Y by replacing 25 percent of the elements from the 25 percent randomly selected samples in Y_0 by outliers that were uniformly distributed in the range $[-10, 10]$. We generated five sets from small-size to large-scale examples ($500 \times 500 \sim 10,000 \times 10,000$). We set the rank of each example matrix to $\min(m, n) \times 0.08$. We compared the proposed methods to IALM, EALM, ALADM, Regl_1-ALM, and l_1-AQP in terms of the reconstruction error and execution time. We used the global parameter for IALM and EALM as in [43].

In the experiment, the average reconstruction error $E_1(r)$ was calculated as

$$E_1(r) = \frac{1}{n}||Y^{org} - Y^{low-rank}||_1, \tag{4.50}$$

where n is the number of samples, Y^{org} is the ground truth, $Y^{low-rank}$ is the matrix approximated by an algorithm.

The average reconstruction errors and execution times are shown in Table 4.2. We did not evaluate the methods l_1-AQP, EALM, and Regl_1-ALM for large-scale data because of their heavy computational load. Unlike the fixed-rank approximation methods that give the matrix whose rank is approximately 8% of the original matrix dimension, IALM and EALM give the matrix whose rank is approximately 55% of the original matrix dimension on average in this section. In the table, l_1-ARG$_D$ gives the best result in terms of the reconstruction error and execution time. Although ALADM takes a shorter execution time compared to the proposed methods, it gives poor reconstruction performance. The proposed methods are superior to the other methods especially for large-scale problems because it uses the weighted median algorithm to handle large-scale problems efficiently. The computational complexities of the proposed methods, ALADM, and l_1-AQP are $O(rmn)$ for each iteration. However, l_1-AQP have to perform a whole convex optimization in each iteration, which is very inefficient in terms of processing time.

The computational complexity is $O(\min(m, n) \max(m, n)^2)$ for IALM and EALM,

Table 4.2: Average performance of the tested algorithms with respect to the reconstruction error and processing time for 25 percent outliers

	m=n=500, r=40		m=n=1,000, r=80		m=n=2,000, r=160		m=n=5,000, r=400		m=n=10,000, r=800	
Algorithm	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time
l_1-ARG$_A$	5.202	2.719	15.294	9.761	28.595	30.437	83.911	224.023	156.335	1657.59
l_1-ARG$_D$	2.583	0.693	6.453	2.002	11.791	8.872	35.678	73.718	71.658	387.328
IALM	3.814	2.973	9.371	25.921	21.378	209.978	172.153	3054.72	-	-
EALM	3.375	83.237	7.340	694.947	14.455	5337.55	-	-	-	-
ALADM	6.141	0.159	19.937	0.848	33.435	4.045	179.167	38.449	387.052	447.013
$Regl_1$-ALM	2.757	31.443	5.142	165.049	14.154	926.017	636.373	8760.74	-	-
l_1-AQP	21.984	4222.22	-	-	-	-	-	-	-	-

Table 4.3: Reconstruction error with respect to various r for a 1,000 $\times$ 1,000 matrix with rank 80

Algorithm	r=70	r=75	r=80	r=85	r=90
l_1-ARG$_A$	202.74	141.64	14.88	15.08	19.68
l_1-ARG$_D$	188.28	126.03	6.16	23.19	45.03
ALADM	199.76	144.13	17.16	30.61	46.63
$Regl_1$-ALM	193.06	129.19	5.01	12.39	21.39

and $O(r \max(m,n)^2)$ for Regl_1-ALM, for each iteration. IALM, EALM, and Regl_1-ALM perform SVD in each iteration, and hence, need much computation time for a large-scale matrix. Figure 4.1 shows the cost function of the proposed methods at each iteration for three examples (500 $\times$ 500, 1000 $\times$ 1000, 2000 $\times$ 2000). As shown in the figure, the cost function of l_1-ARG$_D$ decreases much faster than that of l_1-ARG$_A$, and both methods converge to nearly the same value. Figure 4.2 shows the reconstruction error with respect to the execution time for an example (1,000 $\times$ 1,000). In the figure, the proposed method l_1-ARG$_D$ outperforms other methods. Table 4.3 shows the reconstruction error with respect to various r for a 1,000 $\times$ 1,000 matrix with rank 80. As shown in the table, l_1-ARG$_D$ gives the best results when r is lower than or equal to the exact rank, whereas l_1-ARG$_A$ shows good results when r is larger than the exact rank. It can be explained as follows. Since l_1-ARG$_D$ tries to find a solution that minimizes the cost function for a given r, the performance can be a little bit poorer when r is not correct. l_1-ARG$_A$ finds an approximate solution to the original problem, hence, its result may be worse than l_1-ARG$_D$. But l_1-ARG$_A$ is less sensitive to the rank uncertainty.

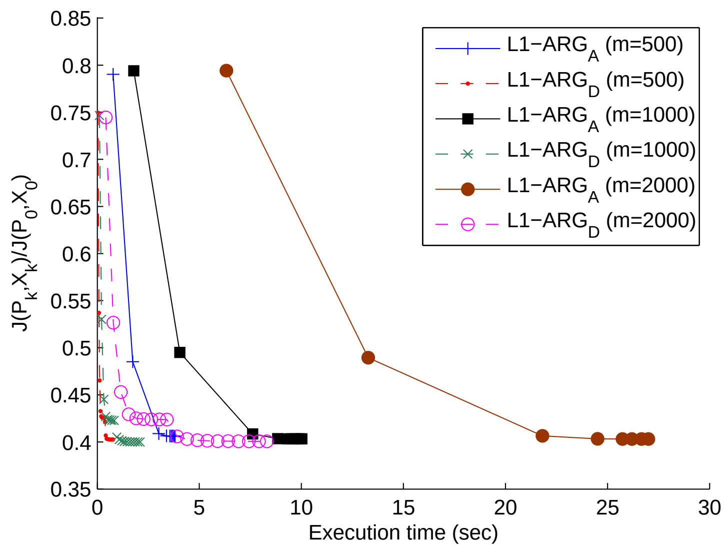


Figure 4.1: Normalized cost function of the proposed algorithms for three examples (500×500, 1000×1000, 2000×2000).

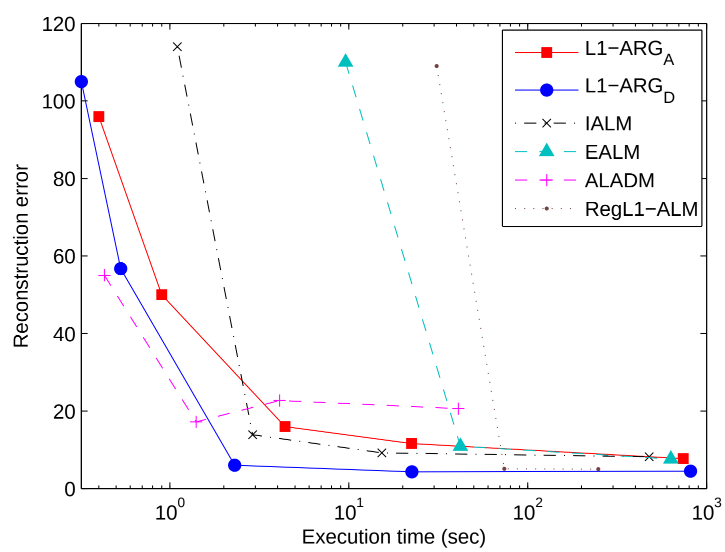


Figure 4.2: Reconstruction error as a function of the execution time (m = 1,000, n = 1,000, r = 80).

Face reconstruction

We applied various methods to face reconstruction problems and compared their performances. In the experiments, we used 830 images having five different illuminations for 166 people from the Multi-PIE face database [84], which were resized to 100×120 pixels. The intensity of each pixel was normalized to have a value in the range of [0, 1]. Each 2-D image was converted to a 12,000-dimensional vector. We only considered an occlusion case for the experiments of the images and measured the average reconstruction error for occluded images. To generate occlusions, 50 percent of the images were randomly selected, and each of selected images was occluded by a randomly located rectangle, whose size varied in the range of 20×20 pixels to 60×60 pixels, with each pixel of the rectangle having a value randomly selected from [0, 1]. We could not apply l_1-AQP and EALM to these problems because they required too much computation time (more than an hour).

Figure 4.3 shows some examples of face images with occlusions and their reconstructed faces with 100 projection vectors. In the figure, we can see that the occlusion blocks have almost disappeared for most of the cases. IALM and EALM tend to produce blurry images, and ALADM gives the poorest results among the methods. Table 4.4 shows the average reconstruction errors E_1 for the face images. In the table, we can see that our methods show competitive performance in both of the reconstruction error and processing time compared to the other methods. IALM and EALM give a little bit smaller errors than our methods, because the ranks of their reconstructed matrices are higher (around 200) than the others (100). Except ALADM, which gives the poorest reconstruction error, all the compared methods are about 4 to 350 times slower than our methods.

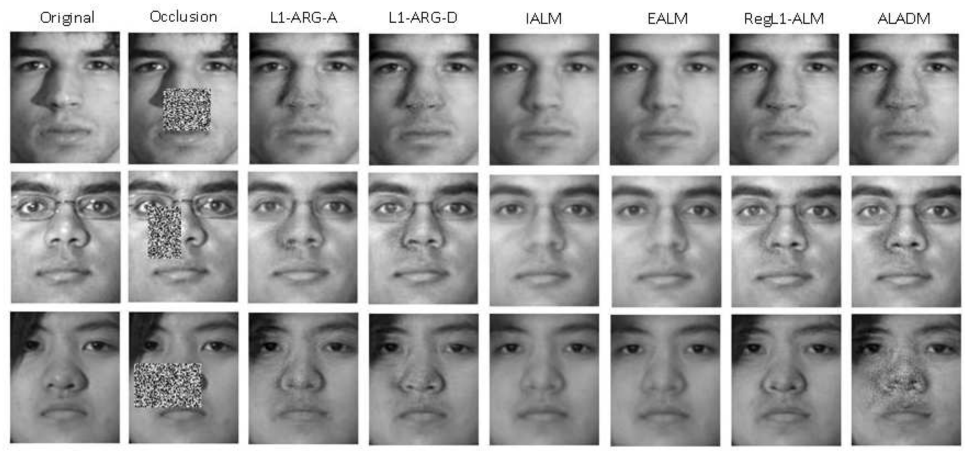


Figure 4.3: Face images with occlusions and their reconstructed faces.

Experiments with missing data

We performed experiments with simple examples in the presence of missing data using the proposed methods Wl_1-ARG$_A$ and Wl_1-ARG$_D$ compared with the other state-of-the-art methods, ALADM-MC [11] and Regl_1-ALM [49], which can handle missing data. We generated five examples as in the previous synthetic problem. Here, we did not perform the experiment for a matrix of $10,000 \times 10,000$ because of memory limitation. To construct the weight matrix, we randomly selected 20 percent of the elements of matrix W for each example and set them to zero (missing), while the other elements were set to one.

Table 4.5 shows the average result for the five examples with outlier and missing data. In the table, Wl_1-ARG$_D$ gives the best performance and needs much shorter execution time than the other methods except ALADM-MC. Although ALADM-MC gives the shortest execution time, its performance is much worse than the proposed methods. Because of the execution time and the performance, Regl_1-ALM is impractical to use for large-scale data.

Table 4.4: Average performance for face data with occlusions

	m=12,000, n=830, r=100	
Algorithm	Error (E_1)	Time (sec)
l_1-ARG$_A$	276.957	71.164
l_1-ARG$_D$	279.442	29.760
IALM	261.895	275.426
EALM	257.392	10543.432
$Regl_1$-ALM	287.749	478.168
ALADM	314.298	9.902

We also performed a face image reconstruction experiment using the proposed methods and the other methods in the presence of occlusions and missing data. Occlusion blocks were generated as described before in 50 percent randomly selected images. To generate missing blocks, 50 percent of images were randomly selected again, and a randomly located square block, whose side length varied from 30 to 60 pixels, was considered as missing in each image. The values of the block elements were set to zero. The number of projection vectors was set to 100. The average reconstruction error E_1 and execution time for various methods are shown in Table 4.6. In the table, Wl_1-ARG$_D$ shows good performance in both of the reconstruction error and execution time compared to the other methods. Although Regl_1-ALM gives the comparable reconstruction error to the proposed methods, its computation time is longer than the proposed methods. Figure 4.4 shows the reconstructed face images in the presence of occlusions and missing data. We do not see much difference between the reconstructed images of the proposed methods and Regl_1-ALM in this figure.

Table 4.5: Average performance for 20 percent outliers and missing data. Rank r is set to $\lceil 0.08 \times \min(\mathrm{m,n}) \rceil$.

	m=n=500		m=n=1,000		m=n=2,000		m=n=5,000		m=n=8,000	
Algorithm	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time (sec)
$\mathrm{W}l_1$-ARG$_A$	3.966	4.188	8.619	11.565	19.506	43.357	51.087	308.300	76.690	968.353
$\mathrm{W}l_1$-ARG$_D$	2.384	0.877	4.967	3.116	10.104	13.272	26.197	95.394	40.127	302.015
ALADM-MC	3.214	0.251	10.122	1.184	25.891	5.748	67.453	48.059	92.389	325.493
$Regl_1$-ALM	2.575	9.402	45.074	53.152	69.273	264.823	89.772	2528.845	191.142	8111.565

Table 4.6: Average performance for face data with occlusions and missing blocks

	m=12,000, n=830, r=100	
Algorithm	Error (E_1)	Time (sec)
Wl_1-ARG$_A$	305.893	262.976
Wl_1-ARG$_D$	319.462	82.671
ALADM-MC	387.628	11.872
$Regl_1$-ALM	327.556	538.014

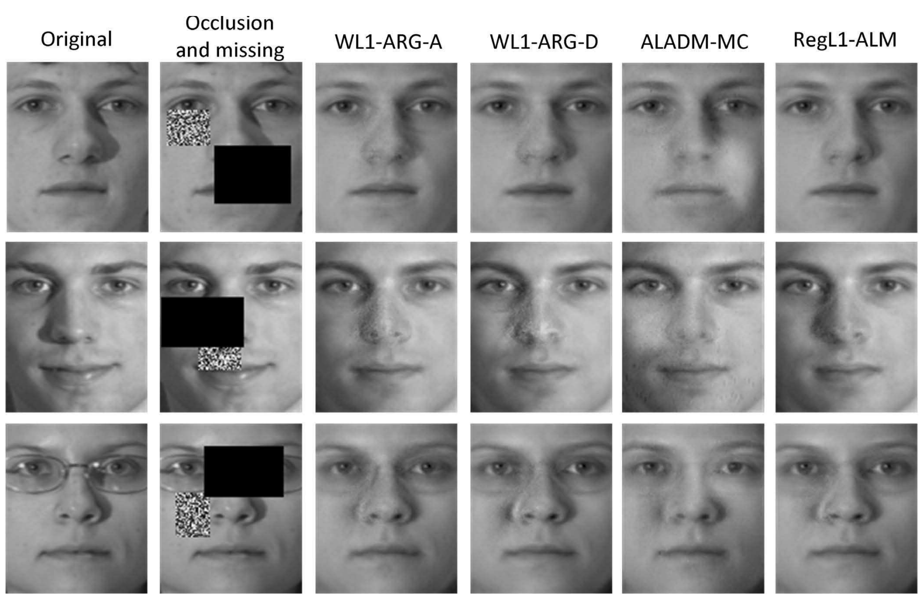


Figure 4.4: Face images with occlusions and missing blocks, and their reconstructed faces.

Non-rigid motion estimation

Non-rigid motion estimation [13] with outliers and missing data from image sequences can be considered as a factorization problem. In this problem, l_1-norm-based factorization can be applied to restore 2D tracks contaminated by outliers and missing data. In this experiment, we used the well-known giraffe sequence[3] consisting of 166 tracked points and 120 frames. The data size is 240×166 and 30.24% of entries are missing. In this section, we also present another algorithm, Wl_1-ARG$_{A+D}$, which is Wl_1-ARG$_D$ using the result of Wl_1-ARG$_A$ as an initial value. The goal of using Wl_1-ARG$_{A+D}$ is to verify the superiority of Wl_1-ARG$_D$ compared to Wl_1-ARG$_A$ by showing that Wl_1-ARG$_D$ can improve the quality of the solution beyond what is possible by Wl_1-ARG$_A$.

To demonstrate the robustness of the proposed method, we replaced 10 percent of the points in a frame by outliers in the range of [-1,000, 2,000], whereas the data points are in the range of [127, 523]. In another experiment, we constructed the data by replacing 20 percent of the points in a frame by outliers. The number of shape bases was set to 2, which gave a matrix of rank $6 = 2 \times 3$ (for x, y, and z coordinates). We compared the proposed weighted version to ALADM-MC and Regl_1-ALM. We set the stopping condition ρ to 10^{-6} and β in (4.43) to 10^{-1}. The result of reconstruction error[4] for the observation data can be seen in Table 4.7. As shown in the table, Wl_1-ARG$_A$ gives better performance than Wl_1-ARG$_D$ but poor than Wl_1-ARG$_{A+D}$ in this problem. We suspect that Wl_1-ARG$_D$ is more sensitive to the initial value and can be trapped in a local minimum for a complex problem. Thus, Wl_1-ARG$_A$ can sometimes find a better solution than Wl_1-ARG$_D$. But when we apply Wl_1-ARG$_D$ with a good initial value, such

[3] Available at `http://www.robots.ox.ac.uk/~abm/`

[4] Reconstruction error is calculated as stated at `http://www.robots.ox.ac.uk/~abm/`

Table 4.7: Reconstruction results for giraffe sequence in the presence of additional outliers

	10% outliers		20% outliers	
Algorithm	Error	Time (sec)	Error	Time (sec)
Wl_1-ARG$_A$	2.910	3.623	3.006	1.589
Wl_1-ARG$_D$	3.224	1.217	3.950	0.895
Wl_1-ARG$_{A+D}$	2.847	4.051	2.979	1.754
$Regl_1$-ALM	3.792	0.810	3.939	0.820
ALADM-MC	9.835	0.017	21.908	0.013

as a solution found by Wl_1-ARG$_A$, we can improve the quality of the solution further. It suggests that the combination Wl_1-ARG$_{A+D}$ can be a good approach for many complex problems. Although ALADM-MC takes shorter execution time than the other methods, it gives poor reconstruction results. Regl_1-ALM gives the competitive results compared to Wl_1-ARG$_A$ w.r.t. to the error and execution time in this experiment.

We also performed the non-rigid motion estimation problem using the shark sequence [13] which consists of 91 tracked points for each non-rigid shark shape in 240 frames. In this data, we examine how robust the proposed methods are for various missing ratios in the presence of outliers. We replaced 10 percent of the points in each frame by outliers in the range of $[-1000, 1000]$, whereas the data points were located in the range of $[-105, 105]$. We set from 10 percent to 70 percent of tracked points as missing in each frame. The number of shape basis for each coordinate was set to two, thus it can be formulated as a rank-6 approximation problem.

Table 4.8: Average error and time (sec) for the Shark sequence.

	missing 10%		missing 30%		missing 50%		missing 70%	
Algorithm	Error	Time	Error	Tim	Error	Time	Error	Time
Wl_1-ARG$_A$	0.069	0.562	0.106	0.819	0.460	0.660	1.767	1.590
Wl_1-ARG$_D$	0.266	0.078	0.366	0.217	0.929	0.233	3.101	0.895
Wl_1-ARG$_{A+D}$	0.063	0.615	0.087	0.895	0.443	0.744	1.676	1.889
$Regl_1$-ALM	0.032	0.805	0.039	0.815	2.739	0.872	24.806	0.364
ALADM-MC	0.402	0.025	0.942	0.023	7.449	0.206	10.015	0.029

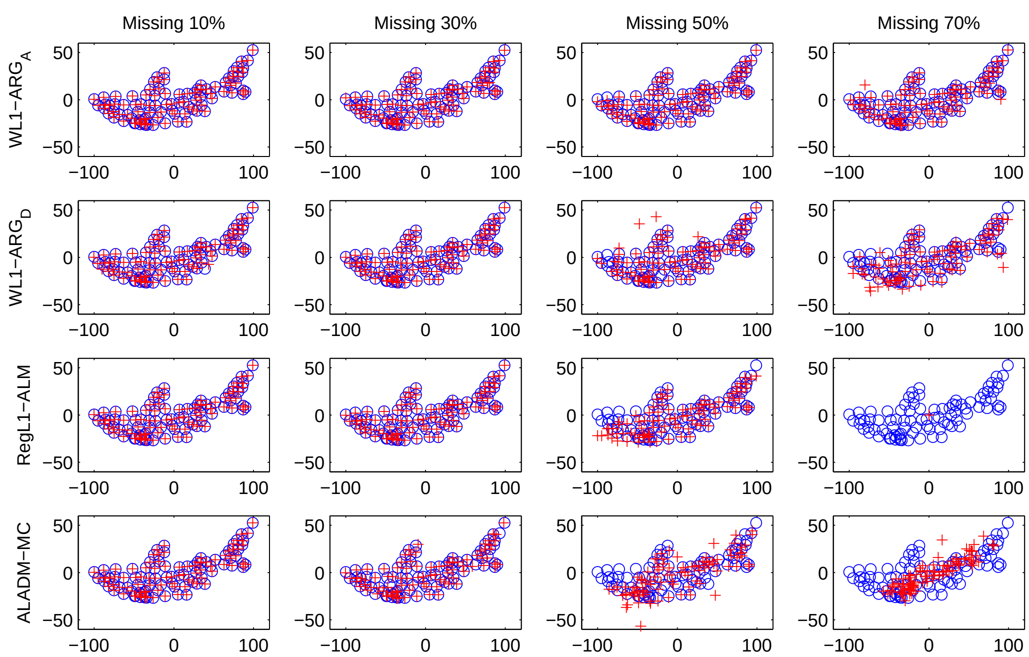


Figure 4.5: Non-rigid shape estimation from the Shark image sequences.

The average performance for the various methods are shown in Table 4.8. Similar to Table 4.7, Wl_1-ARG$_A$ gives better reconstruction results than Wl_1-ARG$_D$ for this problem but performs worse than Wl_1-ARG$_{A+D}$ due to the approximated nature of Wl_1-ARG$_A$. Although Regl_1-ALM gives excellent reconstruction error when 10% and 30% of data were missing, but its performance gets worse as the missing data increases. The reconstruction results for a few selected frames are shown in Figure 4.5.

4.2 Smooth Regularized Fixed-Rank Representation

[5]Since the previous algorithms are based on pure l_1-norm error term without any regularization term, they may be vulnerable to an overfitting issue. Moreover, conventional gradient based methods do not give satisfying results compared to recent advanced in low-rank optimization using augmented Lagrangian framework. From the motivation, we present a new robust orthogonal matrix approximation method using fixed-rank factorization based on the l_1-norm for low-rank subspace learning problems in the presence of various corruptions. We introduce an efficient Frobenius-norm regularizer to prevent the overfitting problem which can arise from an alternative minimization algorithm and orthogonality constraint to reduce the solution space for faster convergence. The proposed regularized optimization problem is constructed under the augmented Lagrangian framework and solved using an alternating direction approach. We also present a rank estimation strategy for the proposed method without increasing the computational complexity to overcome the disadvantage of fixed-rank factorization and the parameterization issue when the exact rank of a problem is unknown.

[5]This section is based on the paper appeared in *Neurocomputing*: "Robust Orthogonal Matrix Factorization for Efficient Subspace Learning" [85].

4.2.1 Robust orthogonal matrix factorization (ROMF)

Problem formulation

In this section, we consider the weighted low-rank matrix approximation problem based on the l_1-norm to consider missing entries simultaneously as follows:

$$\min_{P,X} ||W \odot (Y - PX)||_1, \tag{4.51}$$

where $\|\cdot\|_1$ denotes the entry-wise l_1-norm, i.e., $||S||_1 = \sum_{i,j} |S_{ij}|$ for a matrix S, which is different from the induced l_1-norm. But, when there are no missing entries, we can also solve the problem by setting the values of all elements of W to one. Generally, (4.51) is a nonconvex and nonsmooth problem which is difficult to solve. To solve the problem in practice, a common strategy is to use an alternating minimization approach which solves for one variable while other variables are fixed [10]. In addition, it is reasonable to enforce an orthogonality constraint to the basis matrix, i.e., enforcing P to be a column orthogonal matrix, for the robustness and faster convergence by shrinking the solution space of P. Notice that there can be many pairs of P and X which generate the same multiplication result of PX, i.e.,

$$P'X' = (PH)(H^{-1}X) = PX, \tag{4.52}$$

for some nonsingular matrix $H \in \mathbb{R}^{r\times r}$. Hence, the orthogonality constraint finds P and X, such that $H^T H = I_r$, and this leads to a smaller solution space to work with. We also consider a regularization term for P and X to prevent overfitting.[6]

Note that, without these regularization terms, the problem (4.51) becomes pure l_1 minimization problem and it can be solved by algorithms, such as [11, 21].

[6]Note that regularization constrains a learning algorithm to select a simpler hypothesis h from a hypothesis set $\mathcal{H}$ in order to control overfitting [86].

We can omit the regularization term for P because we enforce an orthogonality constraint over P, which has the smoothness effect as well. From the above analysis, we reformulate the low-rank matrix approximation problems as follows:

$$\min_{P,X} ||W \odot (Y - PX)||_1 + \frac{\lambda}{2}||X||_F^2, \quad \text{s.t.} \quad P^T P = I_r, \tag{4.53}$$

where λ is a weighting parameter and I_r is an $r \times r$ identity matrix.

If the nuclear norm $\|X\|_*$ is used instead of $\|X\|_F$ in (4.53), the problem becomes Regl_1-ALM proposed in [49], which finds a solution by factorization in conjunction with the nuclear-norm minimization to improve convergence. However, it requires a longer computation time than the proposed method since it keeps trying to find a solution with a smaller nuclear-norm under the fixed-rank constraint by performing two singular value decomposition operations at each iteration. There is another approach using a l_1-norm regularized nuclear-norm minimization problem [50] by applying the weight factor λ to the other term. Note that both methods can find a suboptimal solution since the optimization based on the nuclear-norm may find a solution with a rank lower than the desired rank of the problem (see Section 4.2.3 for examples). If (4.53) has another regularization term for P, namely $\|P\|_F$, instead of the orthogonality constraint, it becomes a nuclear-norm regularized optimization problem due to the alternative form of the nuclear-norm [87, 22], $\|S\|_* = \min_{S=PX} \frac{1}{2}(\|P\|_F^2 + \|X\|_F^2)$, when the rank of S is smaller than $\min(m, n)$.

Due to the difficulty of solving the problem (4.53) directly, we introduce an auxiliary variable D and solve the following problem instead.

$$\begin{aligned} \min_{P,X,D} & ||W \odot (Y - D)||_1 + \frac{\lambda}{2}||X||_F^2 \\ \text{s.t.} & \quad P^T P = I_r, \ D = PX. \end{aligned} \tag{4.54}$$

To solve (4.54), we utilize the augmented Lagrangian framework which converts the constrained optimization problem into the following unconstrained optimization problem:

$$\begin{aligned}\mathcal{L}(P,X,D,\Lambda,\beta) = ||W \odot (Y-D)||_1 + \frac{\lambda}{2}||X||_F^2 \\ + \operatorname{tr}(\Lambda^T(D-PX)) + \frac{\beta}{2}||D-PX||_F^2,\end{aligned} \tag{4.55}$$

such that $P^TP = I_r$, where $\Lambda \in \mathbb{R}^{m\times n}$ is a Lagrange multiplier and $\beta > 0$ is a small penalty parameter. We apply the alternating minimization approach iteratively to minimize the augmented Lagrangian as follows:

$$\begin{cases} P = \arg\min\limits_{P}\ \mathcal{L}(P,X,D,\Lambda,\beta) \quad \text{s.t.}\ \ P^TP = I_r \\ X = \arg\min\limits_{X}\ \mathcal{L}(P,X,D,\Lambda,\beta) \\ D = \arg\min\limits_{D}\ \mathcal{L}(P,X,D,\Lambda,\beta) \\ \Lambda = \Lambda + \beta(D-PX). \end{cases} \tag{4.56}$$

Algorithm

To solve for P, we fix the other variables and solve the following optimization problem:

$$\begin{aligned} P =& \arg\min_{P}\ \mathcal{L}(P,X,D,\Lambda,\beta) \\ =& \arg\min_{P}\ \operatorname{tr}(\Lambda^T(D-PX)) + \frac{\beta}{2}||D-PX||_F^2 \\ =& \arg\min_{P}\ \frac{\beta}{2}||D-PX+\beta^{-1}\Lambda||_F^2,\ \text{s.t.}\ \ P^TP = I_r. \end{aligned} \tag{4.57}$$

This optimization problem is the well-known orthogonal Procrustes problem [88]. The orthogonal Procrustes problem finds an orthogonal matrix Ω which minimizes $||A - B\Omega||_F$. A solution to the problem can be found by singular value decomposition (SVD) over B^TA, i.e., if $U\Sigma V^T = \text{SVD}(B^TA)$, then $\Omega = UV^T$ [88]. Therefore, we can solve for orthogonal matrix P using SVD over $(D+\beta^{-1}\Lambda)X^T$.

Hence, if

$$U\Sigma V^T = \mathrm{SVD}((D + \beta^{-1}\Lambda)X^T), \tag{4.58}$$

then the solution to (4.57) becomes $P = UV^T$. Note that SVD is used for an $m \times r$ matrix in (4.58), whereas RPCA performs a single SVD operation on an $m \times n$ matrix and Regl_1-ALM [49] performs two SVD operations on $m \times r$ and $r \times n$ matrices at each iteration. The computational complexity is $O(m^2 r)$ for the proposed method, $O(\min(m,n)\max(m,n)^2)$ for RPCA, and $O(m^2 r + n^2 r)$ for Regl_1-ALM at each iteration. Hence, RPCA and Regl_1-ALM require more computational efforts than the proposed method. The computational complexity of pure l_1 minimization methods such as ALADM [11] and l_1-ARG [21], which do not have a regularization term, is $O(mnr)$ from least squares operations performed at each iteration. When $m > r$, pure l_1 minimization methods are faster than methods using regularization. However, methods using regularization usually perform better than pure l_1 minimization methods in terms of the reconstruction error as demonstrated in Section 4.2.3.

For X, we solve the following optimization problem:

$$\begin{aligned} X &= \arg\min_X \; \mathcal{L}(P, X, D, \Lambda, \beta) \\ &= \arg\min_X \; \frac{\lambda}{2}||X||_F^2 + \mathrm{tr}(\Lambda^T(D - PX)) + \frac{\beta}{2}||D - PX||_F^2. \end{aligned} \tag{4.59}$$

The problem (4.59) is a least-square problem and, thanks to the orthogonality property of P, we obtain the following simple solution:

$$X = \frac{1}{\lambda + \beta} P^T(\Lambda + \beta D). \tag{4.60}$$

For fixed P, X, and Λ, we have the following optimization problem for finding

D:

$$\begin{aligned} D &= \arg\min_{D} ||W \odot (Y - D)||_1 + \frac{\lambda}{2}||X||_F^2 \\ &\quad + \mathrm{tr}(\Lambda^T(D - PX)) + \frac{\beta}{2}||D - PX||_F^2 \\ &= \arg\min_{D} ||W \odot (Y - D)||_1 + \frac{\beta}{2}||D - PX + \beta^{-1}\Lambda||_F^2, \end{aligned} \tag{4.61}$$

and the solution can be computed using the shrinkage (soft-thresholding) operator $\mathcal{S}(\cdot,\cdot)$ [43, 35, 49]:

$$\begin{cases} W \odot D \leftarrow W \odot \left(Y - \mathcal{S}\left(Y - PX + \dfrac{\Lambda}{\beta}, \dfrac{1}{\beta}\right)\right) \\ \overline{W} \odot D \leftarrow \overline{W} \odot \left(PX - \dfrac{\Lambda}{\beta}\right), \end{cases} \tag{4.62}$$

where $\mathcal{S}(x,\tau) = \mathrm{sgn}(x)\max(|x| - \tau, 0)$ for a variable x and a threshold τ and $\overline{W} \in \mathbb{R}^{m\times n}$ is a complementary matrix of W whose element $\overline{w}_{ij}$ is 0 if y_{ij} is known, and is 1 if y_{ij} is unknown.

Finally, we update the Lagrange multiplier Λ as follows:

$$\Lambda = \Lambda + \beta(D - PX). \tag{4.63}$$

Based on the previous analysis, we can derive a robust orthogonal matrix factorization (ROMF) algorithm and it is summarized in Algorithm 5. Note that we can slightly change the algorithm by inserting an inner loop similar to RPCA methods [43], such that we solve for P, X, and D iteratively until they converge in the inner loop, to find a solution elaborately. In the algorithm, we have assumed a normalized observation matrix. Hence, the output matrices P and X can be obtained by rescaling them using the scaling factor. We have found empirically that the algorithm is not sensitive to the choice of initial values. For all results shown in Section 4.2.3, the initial values are all set to zero matrices.

For a real-world application whose elements have nonnegative values, we en-

Algorithm 5 Robust orthogonal matrix factorization (ROMF)

Input: $Y \in \mathbb{R}^{m\times n}$, r, ρ, $\beta = \frac{\beta_0}{\|Y\|_\infty}$, $\beta_{\max} = 10^{10}$, and $\lambda = 10^{-3}$
Output: $P \in \mathbb{R}^{m\times r}$, $X \in \mathbb{R}^{r\times n}$
Initialization: P, X, D, Λ are all zeros
Normalization: $Y \leftarrow Y/||Y||_\infty$
while not converged **do**
  Update P using (4.58)
  Update X using (4.60)
  Update D using (4.62)
  Update the Lagrange multiplier Λ using (4.63)
  $\beta = \min(\rho\beta, \beta_{\max})$
  Check the convergence condition (4.65)
end while
Re-scale P and X

force a lower bound for matrix D at each iteration as follows[7]:

$$\begin{cases} D_{ij} = 0 & \text{if } \ D_{ij} \leq 0, \\ D_{ij} = D_{ij} & \text{if } \ D_{ij} > 0. \end{cases} \tag{4.64}$$

Based on this technique, we obtain better performance empirically when approximating a nonnegative matrix.

In our algorithm, we set the stopping criterion as follows:

$$\frac{||D^{(t)} - P^{(t)}X^{(t)}||_1}{||Y||_1} < \theta, \tag{4.65}$$

[7]Note that the proposed method is not exactly the same as the nonnegative matrix factorization (NMF) methods since NMF enforces the nonnegative constraint for P and X instead of D. But, we borrow the concept from NMF, such that the proposed method can be applied to find a nonnegative low-rank representation.

where t is the number of iterations and θ is a small positive number. Since it is enough for the algorithm to achieve a nearly stationary point when the difference between the terminating cost of adjacent iterations becomes small, we set the stopping condition as $\theta = 10^{-5}$. Here, we compute the whole elements of D including elements corresponding to the unknown entries.

To the best of our knowledge, there is no solid convergence proof for the nonconvex problem (4.53). Shen et al. [11] showed that a nonconvex problem based on a bilinear multiplication under the l_1-norm can achieve a local optimality using the KKT optimality conditions. But, it is difficult to show the convergence of the proposed algorithm due to its nonconvex cost function and the orthogonality constraint. Although it is difficult to guarantee the convergence to a local minimum, an empirical evidence suggests that the proposed algorithm has a strong convergence behavior. Figure 4.6 shows cost values of the proposed method at each iteration for three examples (500×500, 1000×1000, and 2000×2000) described in Section 4.2.3. We have scaled cost values as $(||W \odot (Y - PX)||_1 + \frac{\lambda}{2}||X||_F^2)/||W \odot Y||_1$ in order to display three cases under the same scale. As shown in the figure, the cost value of proposed method (ROMF) decreases fast and converges to a stationary point in a small number of iterations.

4.2.2 Rank estimation for ROMF (ROMF-RE)

Although low-rank matrix approximation based on the fixed-rank factorization is suitable for problems with known ranks, such as structure from motion problems, there are problems for which the target rank is not available. A good rank estimation is essential for low-rank matrix factorization for problems whose rank is unknown. But, there are few methods considering this issue. Cabral et al. [22] suggested a rank continuation strategy, but it is time consuming task because it

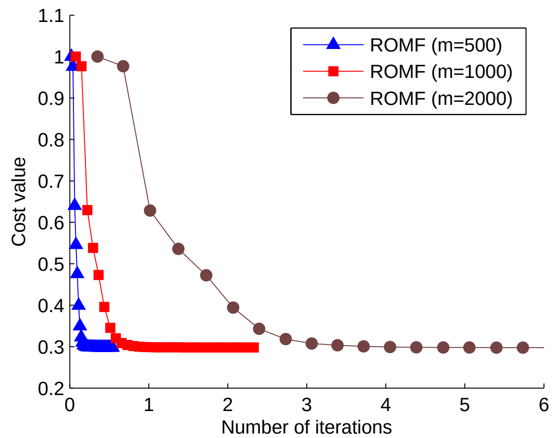


Figure 4.6: Scaled cost values at each iteration of the proposed algorithm for three examples (500×500, 1000×1000, 2000×2000).

performs an additional SVD operation at each iteration, which results in much higher complexity than its fixed-rank optimization algorithm. In this section, we describe a rank estimation extension of the proposed method to handle problems with unknown ranks from the algorithm described in Section 4.2.3 without additional increase in its computational complexity.

Suppose that r_0 is an initial rank which is relatively large compared with the exact rank $r^\star$ ($r_0 > r^\star$) . From the initial rank r_0, we solve the orthogonal Procrustes problem using singular value decomposition (SVD) in the proposed algorithm and check the singular values of diagonal matrix Σ. Note that we do not need additional methods or time consuming computations to estimate the rank information in the algorithm. We can detect the largest drop between adjacent singular values from SVD and this gives a rank estimate such that the largest difference is larger than a minimum threshold θ_{min} as follows:

$$\mathrm{diff(SVs)} = |\sigma_i - \sigma_j| \geq \theta_{min}, \tag{4.66}$$

where i and j are the adjacent indexes satisfying the largest drop. It should be clear that if the algorithm converges to a well-conditioned low-rank solution, then SVD will eventually give a correct answer provided that a proper thresholding value is used [11]. The overall procedure of the rank estimation based robust orthogonal matrix factorization algorithm (ROMF-RE) is described as Algorithm 6.

Algorithm 6 ROMF-RE

Input: $Y \in \mathbb{R}^{m \times n}$, ρ, $\beta = \frac{\beta_0}{\|Y\|_\infty}$, $\beta_{\max}$, and $\lambda = 10^{-3}$
Output: $P \in \mathbb{R}^{m \times r^*}$, $X \in \mathbb{R}^{r^* \times n}$ with output rank r^*
Initialization: P, X, D, Λ are all zeros; initial rank r_0
Normalization: $Y \leftarrow Y/||Y||_\infty$
while not converged **do**
  Update P, X, D using (4.58), (4.60), (4.62), respectively
  if # of iterations $> \theta_c$ **then**
    Find the most reduced point between singular values satisfying diff(SVs) $\geq \theta_{min}$
    Reduced rank: r'
    Update $P = P_{1:r'}$ and $X = X_{1:r'}$
  end if
  Update the Lagrange multiplier Λ using (4.63)
  $\beta = \min(\rho\beta, \beta_{\max})$
  Check the convergence condition (4.65)
end while
Re-scale P and X with final rank r^*

We used the threshold θ_{min} as 10% of the next singular value, i.e., $\frac{1}{10}\sigma_j$, satisfying the largest drop in our experiments. While the RPCA methods find the rank using the soft-thresholding [43], the proposed rank estimation technique finds the rank using a simple thresholding based on singular values obtained from SVD. Since we estimate the rank after a small number of iterations θ_c, the rank estimation step does not increase the total running time of the algorithm significantly. We verify that this simple technique is sufficient to obtain exact solutions in Section 4.2.3 and compare our approach to RPCA.

4.2.3 Experimental results

We evaluated the performance of the proposed method, ROMF, by experimenting with various real-world problems: giraffe [74] and shark [13] sequences for non-rigid motion estimation, the MovieLens dataset [41] for collaborative filtering (CF), and Hall [35], PETS2009 [89] , and Wallflower [90] datasets for background modeling. We compared the proposed algorithm to the state-of-the-art l_1-norm based low-rank matrix approximation methods, ALADM[8] [11], Regl_1-ALM[9] [49], Unifying [22], l_1-ARG$_A$, and l_1-ARG$_D$ [21]. All algorithms listed above can handle missing data and give better performance for practical applications than rank estimation based methods [43, 35], in terms of the reconstruction error and execution time [11, 49]. We also compared ROMF-RE to the rank estimation methods, i.e., IALM, EALM[10] [43, 35], and ROSL [12] for synthetic and background modeling examples in the presence of outliers. We also compared with l_1-ALP [10] for non-rigid motion estimation problems, APG[11] [41] for CF tasks, and nonnegative matrix factorization (NMF)[12] [91] for background modeling problems.

We set the parameters of the proposed method as $\rho = 1.3$ and $\beta_0 = 0.5$ for

[8] `http://lmafit.blogs.rice.edu/`

[9] `https://sites.google.com/site/yinqiangzheng/`

[10] `http://perception.csl.illinois.edu/matrix-rank/sample_code.html/`

[11] `http://perception.csl.illinois.edu/matrix-rank/sample_code.html/`

[12] `http://www.csie.ntu.edu.tw/~cjlin/nmf/`

synthetic and background modeling problems and $\rho = 1.1$ and $\beta_0 = 2 \times 10^{-2}$ for non-rigid motion estimation and CF problems. The trace-norm regularizer λ of Regl_1-ALM [49] was set to 10 with $\rho = 1.05$, which gave the best performance on average in the experiments, unless noted otherwise. The maximum number of inner loops of Regl_1-ALM was set to 100 as stated in [49]. We set the parameters of Unifying [22] to have the best performance according to problems. We set the parameters of ALADM, l_1-ARG$_A$, and l_1-ARG$_D$ as described in [11] and [21], respectively, and initial values for l_1-ALP are chosen randomly.

Synthetic data

First, we applied the proposed method to synthetic examples with outliers and missing data, which is a matrix completion problem. We generated an $m \times r$ matrix B and an $r \times n$ matrix C whose elements are random samples from the Gaussian distribution with zero mean and unit variance. We also generated an $m \times n$ noise matrix N using the Gaussian distribution with zero mean and variance of 0.01. Letting $Y_0 = BC+N$, we constructed an observation matrix Y by replacing 20% of randomly selected entries of 20% of randomly selected columns in Y_0 by outliers, which were uniformly distributed in the range of $[-40, 40]$. We also randomly selected 20% of elements of Y as missing. We generated five test sets: $1,000 \times 1,000$, $2,000 \times 2,000$, $5,000 \times 5,000$, $8,000 \times 8,000$, and $10,000 \times 10,000$. We set the rank of each test data matrix as $r = \lceil \min(m, n) \times 0.08 \rceil$. For Reg$l_1$-ALM, we set $\rho = 1.2$ for synthetic problems. In the experiment, the average reconstruction error E_{Syn} is calculated as

$$E_{Syn} = \frac{1}{n} ||M^{gt} - M^{lr}||_1, \tag{4.67}$$

where n is the number of samples, $M^{gt} = BC$ is the ground truth, and M^{lr} is the low-rank matrix approximated by the applied algorithm.

Table 4.9: Average performance for synthetic problems in the presence of outliers and missing data.

	m=n=1,000		m=n=2,000		m=n=5,000		m=n=8,000		m=n=10,000	
Algorithm	E_{Syn}	Time	E_{Syn}	Time	E_{Syn}	Time	E_{Syn}	Time	E_{Syn}	Time
ROMF	4.71	2.619	9.54	12.28	23.98	100.78	37.88	313.78	47.54	560.05
Unifying	4.75	6.363	9.50	30.31	23.77	256.20	37.98	815.89	47.54	1403.56
l_1-ARG$_A$	9.70	11.28	19.01	44.56	47.41	294.76	75.56	919.68	92.76	1592.3
l_1-ARG$_D$	5.34	3.296	10.10	12.50	25.75	106.44	41.25	290.83	50.46	548.46
Regl_1-ALM	7.53	52.04	14.66	261.79	42.75	2300.5	107.95	7869.4	193.83	13753.9
ALADM	8.80	1.417	16.10	7.03	44.62	54.48	66.74	174.05	82.16	303.69

The average reconstruction errors and execution times (in seconds) are shown in Table 4.9. We could not evaluate l_1-ALP for this experiment because of its heavy execution time. In the table, the proposed method, ROMF, gives the best performance in terms of reconstruction errors and execution times. Although ALADM requires a shorter execution time compared to the proposed method, it performs very poorly in terms of the reconstruction error. The proposed method is superior to other methods, especially for large-scale problems. l_1-ARG$_D$ shows slightly lower performance than the proposed method with respect to both the reconstruction error and execution time. Unifying gives similar reconstruction results to the proposed method, but it takes more computation time than that of the proposed method. In the experiment, Regl_1-ALM takes about 34 times longer on average than the proposed method and it gives poor performance for the case with size $10,000 \times 10,000$, hence, it is not suitable for a large-scale problem.

In order to validate the performance of the proposed method under different settings of parameters β[13] and ρ, we performed an experiment for a 1,000×1,000 synthetic matrix with some outliers similar to the previous experiment. We com-

[13]Note that we used values of β directly from the range of $[10^{-2}, 0.7]$, without dividing it by $\|Y\|_\infty$, for fair comparison in this experiment.

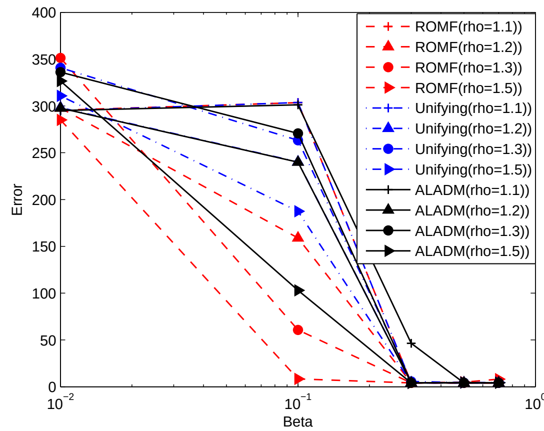


Figure 4.7: Reconstruction results according to variations of two parameters (ρ and β) for three methods (ROMF, ALADM [11], and Unifying [22]).

pared with two other methods, ALADM [11] and Unifying [22], which are based on the ALM framework and showed good performance in the previous examples. Figure 4.7 shows the reconstruction results with respect to various values of β and ρ. All methods find a good solution when β is between 0.3 and 0.5. Overall, the proposed method shows better results than the compared methods on average at different values of β and ρ. Especially, it finds the best solution even when β is lower than 0.3. It shows that the proposed method is less sensitive to changes in parameters than other methods.

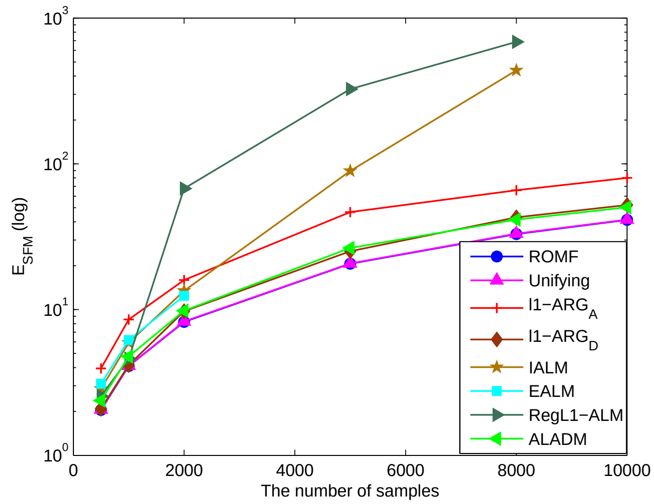
We also applied the proposed method to synthetic examples in the presence of outliers without missing data to compare with the rank minimization methods, IALM and EALM [43, 35], including five fixed-rank approximation methods listed above. We generated Y_0 as before and constructed an observation matrix Y by replacing 20% of randomly selected entries of 20% of randomly selected columns in Y_0 by outliers, which were uniformly distributed in the range of $[-20, 20]$. We generated six test sets with sizes same as the previous example and set the rank of each data matrix as before. All entries are known and all entries of the weight matrix W are one. We set the global parameter for IALM and EALM as described in [35].

Figure 4.8 shows average reconstruction errors and execution times (in seconds) of different algorithms. Similar to the case with outliers and missing entries, the proposed method outperforms the other methods with respect to the reconstruction error in all cases. We could not evaluate the IALM and EALM for large scale experiments since they require much longer computation times. Although Regl_1-ALM shows the similar performance compared with the proposed method, it takes a longer computation time to get a good solution and shows poor performance for large-scale problems. Similar to the previous examples as shown in Table 4.9, Unifying finds the best solution along with ROMF but requires a longer computation time than that of ROMF. The computing time of l_1-ARG$_D$ and ALADM are faster than ROMF, but they give poorer performance than ROMF.

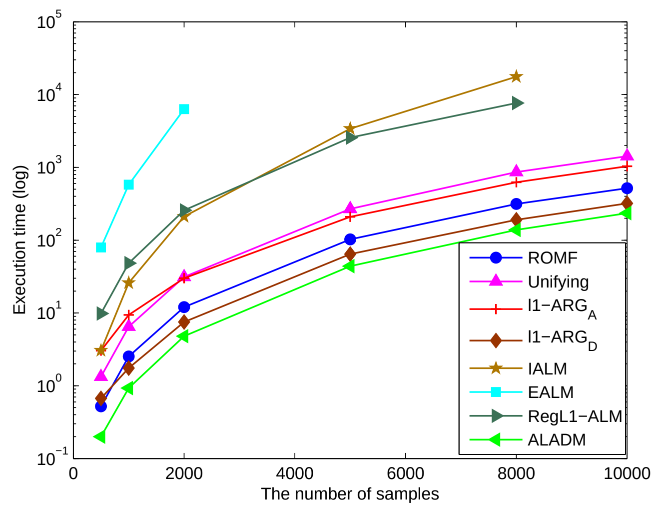
Non-rigid motion estimation

Non-rigid motion estimation [46, 13, 47] in the presence of missing data from image sequences can be considered as a low-rank approximation problem using fixed-rank matrix factorization. In this problem, the proposed robust matrix factorization method based on the l_1-norm can be applied to restore 2D tracks contaminated by outliers and missing data. We conducted two experiments using the well-known benchmark datasets: giraffe [74] and shark [13] sequences. The giraffe sequence[14] consists of 166 tracked points in 120 frames. The data size

[14] `http://www.robots.ox.ac.uk/~abm/`



(a)



(b)

Figure 4.8: Average performances for synthetic problems in the presence of corruptions. (a) Average reconstruction errors with random outliers for various data sizes. (b) Average execution times for various data sizes.

is 240×166 and 30.24% of entries are missing. To demonstrate the robustness and efficiency of the proposed method, we replaced 5% of the randomly selected points in a frame by outliers in the range of $[-50, 50]$, whereas the data points are in the range of $[127, 523]$. In other experiments, we constructed the data by replacing 10% and 15% of points in a frame by outliers, respectively. The number of shape bases was set to two, which gave a matrix of rank $6 = 2 \times 3$ (for x, y, and z coordinates). For non-rigid motion estimation problems, we computed the mean absolute error (MAE) over the observed entries as

$$E_{SFM} = \frac{||W \odot (M^{gt} - M^{lr})||_1}{\sum_{i,j} W_{ij}}. \tag{4.68}$$

The result for the giraffe sequence in the presence of various outlier levels (0% $\sim$ 15%) is shown in Table 4.10. The table also includes the case when no outliers are added. As shown in the table, ROMF gives the best performance regardless of the outlier ratio with fast running times. Although ALADM shows a similar reconstruction error to the proposed method when there is no outlier, the difference between them gets larger when the outlier ratio increases. Regl_1-ALM gives competitive performance compared to ROMF when there are many outliers, but it requires a longer computation time. l_1-ARG$_A$ and l_1-ARG$_D$ shows a higher reconstruction error than the proposed method. l_1-ALP requires the longest execution time and returns a poor reconstruction result when the outlier ratio increases.

We also performed the non-rigid motion estimation problem using the shark sequence [13] which consists of 91 tracked points for each non-rigid shark shape in 240 frames. In this data, we examine how robust the proposed method is for various missing ratios in the presence of outliers. We replaced 5% of the points in each frame by outliers in the range of $[-1000, 1000]$, whereas the data points were located in the range of $[-105, 105]$. Likewise, we replaced 10% and 15% of

Table 4.10: Reconstruction results for the giraffe sequence in the presence of additional outliers.

	no outliers		5% outliers		10% outliers		15% outliers	
Algorithm	E_{SFM}	Time	E_{SFM}	Time	E_{SFM}	Time	E_{SFM}	Time
ROMF	0.294	0.092	0.397	0.098	0.596	0.104	1.442	0.101
Unifying	0.302	0.088	0.463	0.089	1.116	0.098	2.001	0.097
l_1-ARG$_A$	0.638	3.05	0.697	2.239	0.780	1.450	1.345	1.449
l_1-ARG$_D$	0.491	0.603	0.531	0.611	1.461	0.671	3.214	0.691
Regl_1-ALM	0.606	21.78	0.653	19.301	0.673	18.517	0.808	18.517
ALADM	0.387	0.064	1.379	0.060	3.199	0.061	7.702	0.061

the points in each frame by outliers. We set 10% of tracked points as missing in each frame. The number of shape basis for each coordinate was set to two, thus it can be formulated as a rank-6 approximation problem.

Average reconstruction errors at various outlier ratios by different methods are shown in Figure 4.9. As shown in the figure, the proposed method gives good performance compared to other algorithms, except l_1-ARG$_A$. In this case, gradient based methods using the l_1-norm find good solutions. Although l_1-ARG$_A$ gives excellent performance than the proposed method in the presence of outliers, its performance is worse than the proposed method when there are fewer outliers. Unifying finds a suboptimal solution compared to the proposed method on average in this problem. Although Regl_1-ALM and ALADM give good reconstruction results when the number of missing data points is small, its performance gets worse as the missing data ratio increases. It is interesting to notice that Regl_1-ALM sometimes finds a solution whose rank is five for this rank-6 problem. This is due to the fact that Regl_1-ALM minimizes the nuclear-norm of PX, making the method unsuitable for fixed-rank approximation problems. The execution times of the methods are 0.039 sec for the proposed method, 0.026 sec for Unifying,

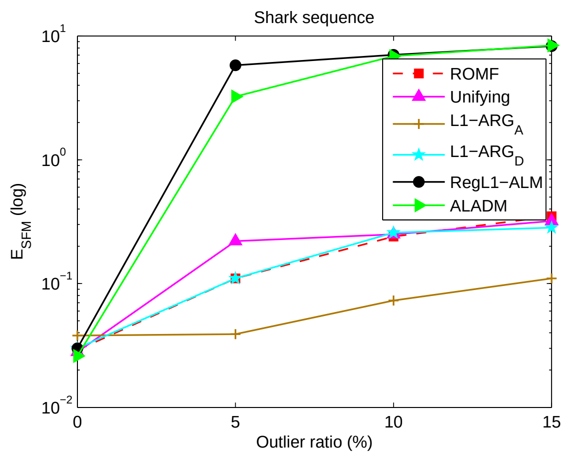


Figure 4.9: Average reconstruction errors at various missing ratios for the shark sequence by different algorithms.

0.528 sec for l_1-ARG$_A$, 0.073 sec for l_1-ARG$_D$, 1.866 sec for Regl_1-ALM, and 0.074 sec for ALADM, respectively, for the case with 20% missing data. For another experiment, we replaced 10% of the points in each frame by outliers and set from 0% to 60% of tracked points as missing in each frame. The reconstruction results for the 5-th frame are shown in Figure 4.10. From the figure, we can observe excellent reconstruction results by the proposed method against missing data and outliers compared to the other approaches.

Collaborative filtering

We conducted two collaborative filtering (CF) problems. Low-rank matrix factorization is a common tool for CF problems and has shown successful results [1, 41]. We used two popular recommendation system datasets, MovieLens100K and MovieLens1M[15]. MovieLens100K consists of 100,000 observation ratings from

[15] `http://www.grouplens.org/node/73`

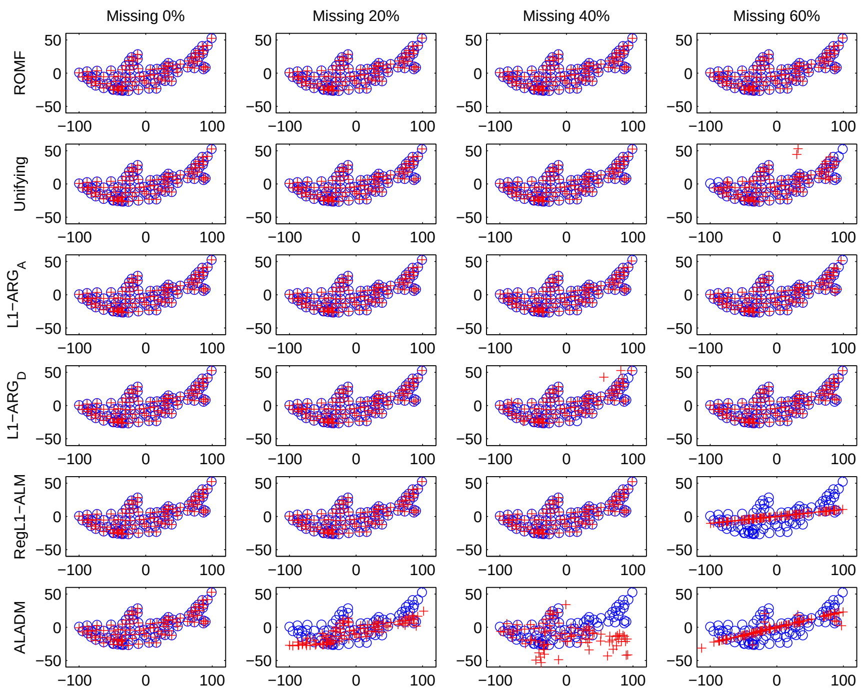


Figure 4.10: Some reconstruction results from the Shark sequence by different methods. Each row shows the result of each method. From top to bottom: the proposed method, Unifying [22], l_1-ARG$_A$ [21], l_1-ARG$_D$ [21], Regl_1-ALM [49], and ALADM [11]. Each column represents the result according to the different missing ratio.

943 users for 1,682 movies, hence, the data size is $943 \times 1,682$ and has 6.3% sparsity. Ratings are integer-valued ranging from one to five, and no ratings are missing. MovieLens1M consists of one million ratings from 6,040 users for 3,952 movies which leads to an observation matrix of size $6,040 \times 3,952$ with 4.2% sparsity. We did not experiment the largest dataset, MovieLens10M, whose size is $71,567 \times 10,674$, due to the memory limitation of the PC used in the experi-

Table 4.11: Reconstruction results for two CF problems.

	MovieLens100K		MovieLens1M	
Algorithm	E_{CF}	Time (sec)	E_{CF}	Time (sec)
ROMF	0.1702	6.848	0.1587	96.47
l_1-ARG$_A$	0.1797	45.035	0.1637	674.74
l_1-ARG$_D$	0.1709	20.468	0.1596	264.05
Regl_1-ALM	0.1738	261.44	0.1591	3952.49
ALADM	0.1861	1.507	0.1843	20.90
APG	0.1921	4.375	0.1997	98.049

ment.

Given the observation data, we split the data into training and test datasets by randomly selecting 90% as a training set and remaining 10% as a test set. In this experiment, we used the normalized mean absolute error (NMAE):

$$E_{CF} = \frac{E_{SFM}}{d_{\max} - d_{\min}}, \tag{4.69}$$

where $d_{\max}$ and $d_{\min}$ are the upper and lower bound of ratings to measure the performance. We set the number of inner loops of Regl_1-ALM to 10 because of the time limitation.

Table 4.11 shows the estimation results of the proposed method compared to other methods: l_1-ARG$_A$ [21], l_1-ARG$_D$ [21], Regl_1-ALM [49], ALADM [11], and APG [41]. We set the rank r to three for MovieLens100K and five for MovieLens1M. In the table, the proposed method, ROMF, gives the best estimation results with shorter execution times for both datasets. Although ALADM is about four times faster than ROMF, it shows worse estimation results than ROMF in all experiments while Regl_1-ALM takes a very long time to obtain a solution. l_1-ARG$_D$ gives the similar reconstruction results compared to the proposed method, but it takes a longer computation time than the proposed method. APG shows the worst results among the methods tried in this experiment. This result is similar to the results reported in [41], which reports 0.193 for MovieLens100K and 0.194 for MovieLens1M, using randomly chosen subsamples.

Background modeling

Modeling background from a video sequence is an important step to separate foreground objects from background and applied to many applications, including video surveillance, traffic monitoring, and abnormal behavior detection [92]. A background modeling task can be considered as a low-rank matrix approximation problem [41, 35]. We used three benchmark video datasets: Hall[16] [35], PETS2009[17] [89], and Wallflower[18] [90] datasets. The Wallflower dataset is used to compare different methods quantitatively since it provides the ground-truth data as well.

The Hall dataset is a sequence of 200 frames taken in a hall of a business building. The frame size is 176 × 144 and the whole data size is 25,344 × 200. We converted color images into gray-scale images and performed the proposed method compared with other fixed-rank matrix approximation methods: l_1-ARG$_D$ [21], NMF [91], Regl_1-ALM [49], and ALADM [11]. The rank r of the fixed-rank approximation methods was set to 3. Figure 4.11 shows the background modeling results of the methods for two selected frames. From the figure, the proposed method successfully decomposes into background and foreground images, while some of other methods (NMF, Regl_1-ALM, and ALADM) shows afterimages in the estimated background image (see the second column of Figure 4.11). l_1-ARG$_D$ and Unifying shows good separation results which are comparable to ROMF.

[16] http://perception.i2r.a-star.edu.sg/bk_model/bk_index.html/

[17] http://http://www.cvg.rdg.ac.uk/PETS2009/a.html/

[18] http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm/

Figure 4.11: Background modeling results of the proposed method, l_1-ARG$_D$, Unifying, NMF, Regl_1-ALM, and ALADM (Hall dataset). Each algorithm decomposes the original image into background and foreground images.

We also compared the proposed method with the rank estimation methods (IALM, EALM [43], and ROSL [12]). In this experiment, the proposed method used the rank estimation technique, ROMF-RE, described in Section 4.2.2. We set the initial rank r_0 to three times of r in this problem. We set the parameter of IALM, EALM, and ROSL as described in [35]. Figure 4.12 shows the background modeling results of the rank estimation methods. All methods separated foreground from background in all cases. However, ROSL sometimes finds a sub-

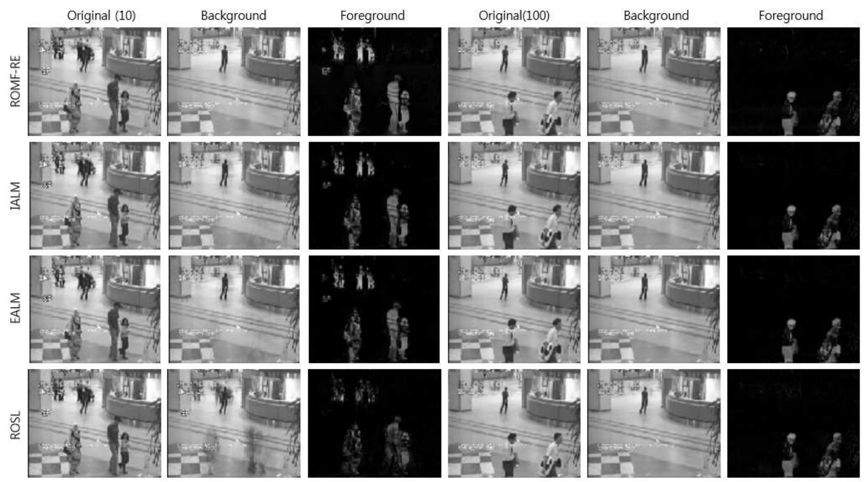


Figure 4.12: Background modeling results of ROMF, IALM, EALM, and ROSL (Hall dataset). Each algorithm decomposes the original image into background and foreground images.

optimal solution as shown in the first frame in Figure 4.12. In common with rank minimization methods, the proposed method using rank estimation technique has successfully found good solutions.

The PETS2009 dataset is a sequence of 221 frames taken in a school. Unlike the previous experiment, we used the color image of the PETS2009 dataset as it is. The frame size is 576×768 and the stacked data size is $442{,}368 \times 221$ for each channel. In this case, the proposed method with a rank estimation is compared to two selected rank estimation methods (IALM and ROSL). Figure 4.13 shows the separation results. As shown in the figure, IALM fails to separate background and foreground correctly while the proposed method separates background and foreground exactly. ROSL seems to find a background image very well, but it fail to find a foreground image as shown in the figure.

Figure 4.13: Background modeling results of ROMF, IALM, and ROSL (PETS2009 dataset). Each algorithm decomposes the original image into background and foreground images.

For the Wallflower dataset, we used the Bootstrapping sequence which consists of several minutes of an overhead view of a cafeteria [90]. The sequence has no separate data for background modeling [90] and more complex than other sequences in the Wallflower dataset. We selected first 300 frames as an observation. The 300th frame comes with a foreground ground-truth image and this frame is used to compute the background modeling performance of each algorithm in terms of precision and recall. Figure 4.14 shows an example of the 300th frame image with its corresponding ground-truth mask. The frame size is 160×120 and the whole dataset is $19{,}200\times300$. We converted images into gray-scale images and added a mean-zero unit variance Gaussian noise to 25% pixels which are selected randomly. The rank of factorization methods was set to 2. For quantitative comparison, pixel-wise thresholding and mathematical morphology (closing) were performed for foreground images extracted from each method. The final foreground mask after post-processing was used to compute the precision and

Figure 4.14: An image from the Bootstrapping sequence and its ground truth mask.

recall as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \tag{4.70}$$

where TP is the number of correctly estimated foreground pixels, FP is the number of background pixels that are wrongly estimated as foreground, and FN is the number of foreground pixels that are wrongly estimated as background. Figure 4.15 shows precision-recall curves of different methods, including two proposed methods, for the Bootstrapping sequence. The proposed method, ROMF, outperforms other methods especially when the precision is lower than 0.85. Although ROMF with rank estimation, ROMF-RE, shows a moderate improvement when precision is low, it shows good performance on average. The l_1-norm based approaches (l_1-ARG$_D$, ALADM, and Regl_1-ALM) show poor performance.

The required computation times of all methods for three datasets are shown in Table 4.12. For the PETS2009 dataset, we compared execution times using a single channel. For Hall and Bootstrapping datasets, we compared execution times for gray-scale images. The proposed method shows the second fastest computation time on average except NMF, which is an l_2-norm base approach. In addition, ROMF-RE requires a longer computation time than ROMF since it needs addi-

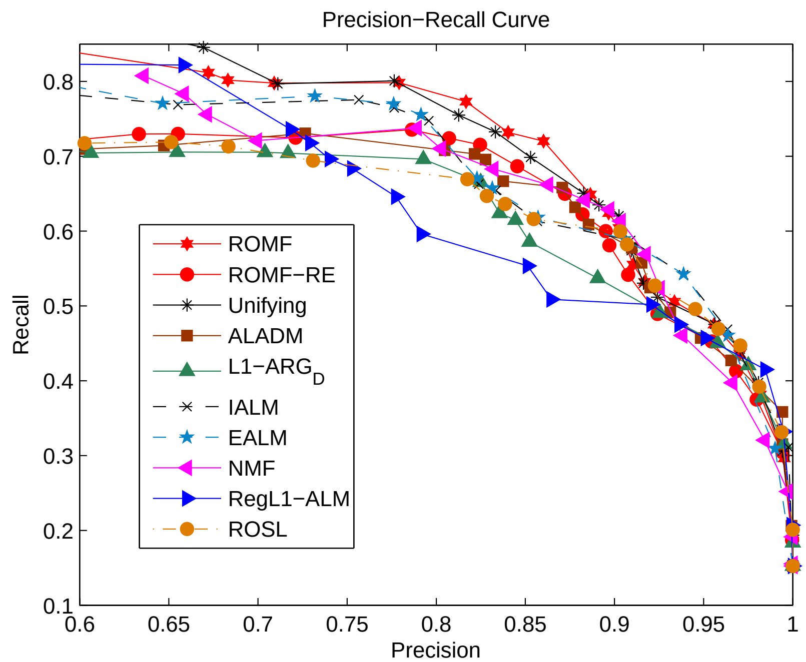


Figure 4.15: Precision-recall curves of different methods for the Bootstrapping dataset.

tional operations, but the difference is relatively small. Although ALADM gives the fastest computation time of all l_1-norm base methods, it sometimes fails to provide good approximations compared to other methods.

4.3 Structured Low-Rank Representation

[19]The previous algorithms in this chapter generally solve an unstructured matrix with a column- or row-wise low-rank assumption. However, what if an observa-

[19]This section is based on the paper appeared in *IEEE International Conference on Robotics and Automation*: "Structured Low-Rank Matrix Approximation in Gaussian Process Regression for Autonomous Robot Navigation" [93].

Table 4.12: Comparison of execution times (sec) of all methods for background modeling.

Algorithm	Hall (25,344×200)	PETS2009 (442,368×221)	Bootstrapping (19,200×300)
ROMF	7.019	133.02	5.536
ROMF-RE	9.250	230.80	7.433
l_1-ARG$_D$	6.442	147.78	25.354
Unifying	38.487	622.66	24.669
NMF	1.891	81.45	1.039
IALM	18.791	298.92	27.342
EALM	468.960	21354.31	1950.97
ROSL	10.316	193.10	5.954
Regl_1-ALM	161.720	3348.83	208.28
ALADM	4.780	86.76	3.219

tion matrix or a problem at interest is structured situation unlike the previous cases? In this section, we address a general matrix approximation problem where an observation is structured condition or a kernel matrix. We first discuss a kernel subspace learning problem as a basic problem. Then, we propose a novel factorization-based robust structured kernel subspace learning with low-rank assumption. We apply the proposed learning algorithm to Gaussian process regression (GPR) which is a important method based on a kernel matrix. The proposed method based GPR, named *FactGP*, is applied to various regression and motion prediction problems in simulation to demonstrate its robustness against outliers.

4.3.1 Kernel subspace learning

To reduce the computational cost of inverting the kernel matrix Λ in (2.15), a number of approximation methods have been proposed, including Incomplete Cholesky Factorization (ICF) [94] and the Nyström method [95]. In this section, we consider low-rank kernel matrix approximation to invoke robustness in the presence of noises or outliers, which is also known as kernel principal component analysis [96]. It has been attracted much attention for a wide range of problems in order to efficiently process a large quantity of data and to discover a hidden low-dimensional structure based on the Euclidean distance (l_2-norm).

The main idea behind the kernel-based approximation method is that, by using a kernel function, the original linear operations of principal component analysis (PCA) are performed in a high-dimensional Hilbert space [96]. Performing linear PCA in a high-dimensional space has an effect of performing nonlinear PCA in the original input space [96]. Hence, we can apply low-rank kernel matrix approximation to reduce the computation load of Λ in (2.15) to speed up the kernel machine.

Suppose that a nonlinear function $\Phi : \mathbb{R}^{n_x} \to \mathbb{X}$ is a mapping from the input space $\mathbb{R}^{n_x}$ with dimension n_x to a high-dimensional feature space $\mathbb{X}$. Then, for centered data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the covariance matrix in $\mathbb{X}$ is

$$C = \frac{1}{n} \sum_{i=1}^{n} \Phi(\boldsymbol{x}_i) \Phi(\boldsymbol{x}_i)^T$$

and the eigenvector $\boldsymbol{v}$ with nonzero eigenvalue of C can be represented as $\boldsymbol{v} = \sum_{i=1}^{n} \beta_i \Phi(\boldsymbol{x}_i)$. The coefficients $\boldsymbol{\beta} = [\beta_1 \cdots \beta_n]^T$ can be found by solving the following eigenvalue problem [96]:

$$K\boldsymbol{\beta} = n\lambda\boldsymbol{\beta}, \tag{4.71}$$

where K is a kernel matrix such that $[K]_{ij} = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j) \rangle$. It follows that principle components in $\mathbb{X}$ can be extracted using top r largest eigenvectors, $\boldsymbol{v}_k$ for $k = 1, \ldots, r$, over the entire eigenvectors of K based on their corresponding eigenvalues which are computed using coefficients found from (4.71) with a proper normalization. Hence, we can effectively represent a kernel matrix using a subset of eigenvectors with r largest eigenvalues.

Given the eigenvalue decomposition of $K = R\Sigma R^T$, where $\Sigma = \text{diag}(\lambda_1, \cdots, \lambda_n)$ is a diagonal matrix of eigenvalues of K, such that $\lambda_1 \geq \cdots \geq \lambda_n$, we can approximate the inverse of K as follows:

$$K^{-1} = (R\Sigma R^T)^{-1} = R\Sigma^{-1}R^T \approx \widetilde{R}\widetilde{R}^T, \tag{4.72}$$

where $\widetilde{R} = R_r\Sigma_r^{-\frac{1}{2}}$. Here, $R_r \in \mathbb{R}^{n\times r}$ collects the first r vectors from R and $\Sigma_r = \text{diag}(\lambda_1, \cdots, \lambda_r) \in \mathbb{R}^{r\times r}$ is a diagonal matrix of r largest eigenvalues. Hence, we can reformulate (2.15) by treating Λ as K in (4.72) as

$$\overline{\boldsymbol{y}}_* = k_*^T\Lambda\boldsymbol{y} \approx k_*^T\widetilde{R}\widetilde{R}^T\boldsymbol{y} = \widetilde{k}_*^T\widetilde{\boldsymbol{y}}, \tag{4.73}$$

where $\widetilde{k}_*^T = k_*^T\widetilde{R}$ is a kernel vector which is projected into the orthogonal feature space $\widetilde{R}$ and $\widetilde{\boldsymbol{y}} = \widetilde{R}^T\boldsymbol{y}$ is a projected output vector into $\widetilde{R}$. This means that the low-dimensional approximation of a kernel matrix can be applied to Gaussian process regression problems by using $\widetilde{k}_*$ and $\widetilde{\boldsymbol{y}}$ which are projected on $\widetilde{R}$, and the inverse of a kernel matrix becomes an identity matrix which represents the independent relationship between basis vectors. Hence, (4.73) can be another representation of $\overline{y}_*$ in the dimensionally reduced orthogonal feature space $\widetilde{R}$. Figure 4.16 shows the concept of the proposed method using low-rank kernel matrix approximation.

In addition, Λ can be approximated by a conventional low-rank approximation method which transforms data into a low-dimensional subspace which maximizes the variance of the given data based on the Euclidean distance (l_2-norm). However, the method is sensitive to outliers because the l_2-norm can sometimes amplify the negative effects of such data. Therefore, l_2-norm based low-rank approximation methods may find projections which are far from the desired solution due to the corruptions. As an alternative, various methods based on the l_1-norm have been proposed recently and it is known that l_1-norm based methods find

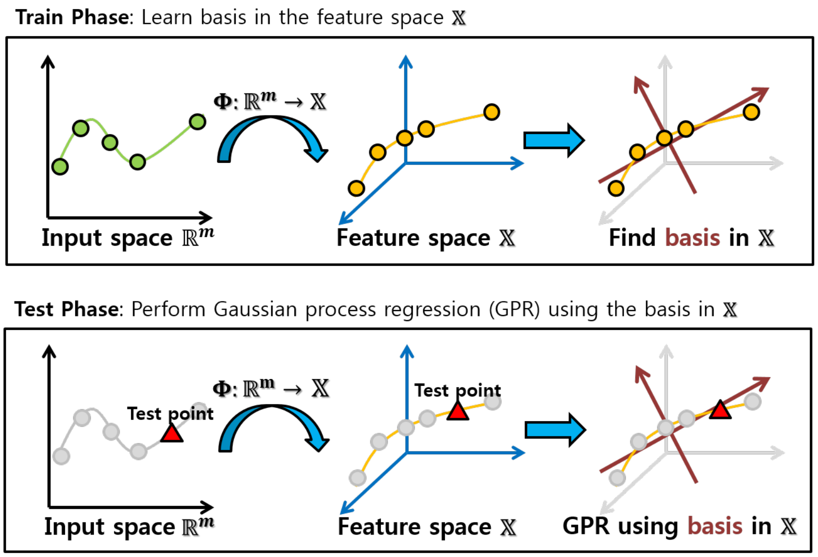


Figure 4.16: A graphical illustration of low-rank kernel matrix approximation. We can perform the prediction step of Gaussian process regression in the dimensionality reduced feature space.

a sparse solution, which are more robust against outliers [10, 25, 21]. Recently, Kim et al. [97] approximated a kernel matrix using l_1-norm based kernel matrix factorization for robust autoregressive Gaussian process motion model:

$$\min_{U,V} \quad J(U,V) = \|K - UV\|_1, \tag{4.74}$$

where $K \in \mathbb{R}^{n\times n}$, $U \in \mathbb{R}^{n\times r}$, and $V \in \mathbb{R}^{r\times n}$ are the kernel, projection, and coefficient matrices, respectively. Here, we want to find a low-rank representation UV of K with sparse approximation errors, such that the effects of outliers can be reduced. However, the optimization technique in [97] may not be proper when approximating a kernel matrix since the low-rank representation is a bilinear multiplication and thus may not satisfy the positive semi-definiteness of a kernel matrix.

4.3.2 Structured kernel subspace learning in GPR

In this section, we first propose a structured kernel subspace method for approximating a kernel matrix by making sure that the approximated matrix is positive semi-definite. Then, we describe the overall framework using Gaussian process regression for modeling motion.

Problem Formulation

For robustness of the proposed method in the presence of erroneous data, we use robust measures in a cost function. Instead of methods based on the l_2-norm, the proposed method is based on the recent advances in nuclear-norm and l_1-norm minimization, which is also called robust principal component analysis (RPCA) [35], to reduce the effect of outliers with an automatic rank search.[20] Hence, we approximate a kernel matrix using a nuclear-norm regularized l_1-norm minimization problem for robust approximation.

We formulate the problem of nuclear-norm regularized l_1-norm minimization as shown below:

$$\min_{P,M} \|K - PMP^T\|_1 + \lambda \|PMP^T\|_*, \tag{4.75}$$

subject to positive semi-definite matrix M, where $K \in \mathbb{R}^{n \times n}$ is a kernel or symmetric positive semi-definite matrix and $P \in \mathbb{R}^{n \times r}$ and $M \in \mathbb{R}^{r \times r}$ are optimization variables. $\| \cdot \|_*$ denotes the nuclear-norm or trace-norm, and $\lambda > 0$ is a regularization parameter. In the cost function, we use the nuclear-norm regular-

[20] Note that the original RPCA solves the nuclear-norm based optimization problem by iterative thresholding over singular values obtained from singular value decomposition of a measurement matrix, which leads to the automatic rank search. But, the proposed framework fixes the rank of the target matrix PMP^T. Nonetheless, it has an effect of reducing the rank of the target matrix further from the pre-determined rank.

izer to minimize the rank of PMP^T, an approximation of K, to our desired one by adjusting the parameter λ since the exact rank is not known. The nuclear-norm has been used as a convex surrogate for the rank in many rank minimization problems [98, 35]. This problem is non-convex and its solution can be obtained using the augmented Lagrangian framework [35].

To reduce the computational complexity and make the convergence faster, it is reasonable to enforce an orthogonality constraint to the basis matrix P by shrinking the solution space of P. Based on these observations, we reformulate the low-rank matrix approximation problem as follows:

$$\begin{aligned} &\min_{P,M} \; \|K - PMP^T\|_1 + \lambda \|M\|_* \\ &\text{s.t.} \;\; P^T P = I_r, \quad M \succeq 0, \end{aligned} \tag{4.76}$$

where I_r is an $r \times r$ identity matrix and M is a positive semi-definite matrix. By enforcing the orthogonal constraint on P, we can compute only small matrix M instead of PMP^T when computing the nuclear-norm. Figure 4.17 shows an overview of the proposed structured low-rank matrix approximation method. Due to the difficulty of solving the problem (4.76) directly, we introduce two auxiliary variables, D and $\widehat{M}$, and solve the following problem:

$$\begin{aligned} &\min_{P,M,D,\widehat{M}} \|K - D\|_1 + \lambda \|M\|_* \\ &\text{s.t.} \;\; D = P\widehat{M}P^T, \; \widehat{M} = M, P^T P = I_r, M \succeq 0. \end{aligned} \tag{4.77}$$

The augmented Lagrangian framework [35] is used to solve (4.77) by converting the constrained optimization problem into the following unconstrained problem:

$$\begin{aligned} &\mathcal{L}(K, P, M, D, \widehat{M}) = \|K - D\|_1 + \lambda \|M\|_* \\ &+ \operatorname{tr}\left(\Lambda_1^T (D - P\widehat{M}P^T)\right) + \operatorname{tr}\left(\Lambda_2^T (\widehat{M} - M)\right) \\ &+ \frac{\beta}{2}\left(\|D - P\widehat{M}P^T\|_F^2 + \|\widehat{M} - M\|_F^2\right), \end{aligned} \tag{4.78}$$

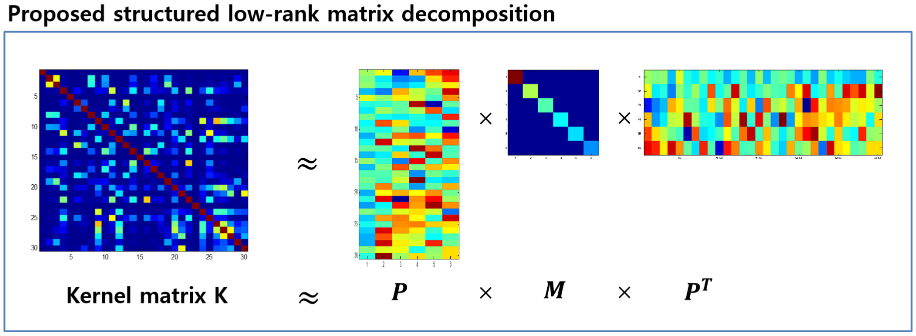


Figure 4.17: A graphical illustration of the proposed method. A kernel matrix K can be approximated by multiplication of P, M, and P^T. We can predict future motions of moving objects using AR-GP based on the rank reduced kernel matrix.

subject to the constraints $P^TP = I_r$ and $M \succeq 0$, where $\Lambda_1, \Lambda_2 \in \mathbb{R}^{n \times n}$ are Lagrange multipliers and $\beta > 0$ is a small penalty parameter. We apply the alternating minimization approach iteratively, which estimates one variable while other variables are held fixed. Each step of the proposed algorithm is described in the following section.

Algorithm

To solve for M, we fix the other variables and solve the following optimization problem:

$$\begin{aligned} M_+ &= \arg\min_M \frac{\lambda}{\beta}\|M\|_* + \frac{1}{2}\left\|\widehat{M} - M + \frac{\Lambda_2}{\beta}\right\|_F^2, \\ &= \arg\min_M \frac{\lambda}{\beta}\|M\|_* + \frac{1}{2}\|M - A\|_F^2, \text{ s.t. } M \succeq 0, \end{aligned} \tag{4.79}$$

where $A = \widehat{M} - \frac{\Lambda_2}{\beta}$. If A is not a symmetric matrix, we make it a symmetric matrix by $A \leftarrow \frac{A+A^T}{2}$ and find M_+. Then, this problem can be solved using eigenvalue thresholding (EVT) [99] and its solution is

$$M_{+} = Q\text{diag}\left[\max\left(\gamma - \frac{\lambda}{\beta}, 0\right)\right] Q^T, \tag{4.80}$$

where Q and Γ are matrices, which contain eigenvectors and eigenvalues, respectively, from the eigenvalue decomposition of A, i.e., $A = Q\Gamma Q^T$ and $\Gamma = \text{diag}(\gamma)$.

For D, we solve the following problem:

$$\begin{aligned} D_{+} &= \arg\min_{D} \|K - D\|_1 + \text{tr}\left(\Lambda_1^T(D - P\widehat{M}P^T)\right) \\ &\quad + \frac{\beta}{2}\|D - P\widehat{M}P^T\|_F^2, \\ &= \arg\min_{D} \|K - D\|_1 + \frac{\beta}{2}\left\|D - P\widehat{M}P^T + \frac{\Lambda_1}{\beta}\right\|_F^2, \end{aligned} \tag{4.81}$$

and the solution can be computed using the shrinkage (soft-thresholding) operator [35]:

$$D_{+} \leftarrow K - \mathcal{S}\left(K - P\widehat{M}P^T + \frac{\Lambda_1}{\beta}, \frac{1}{\beta}\right), \tag{4.82}$$

where $\mathcal{S}(x, \tau) = \text{sgn}(x) \cdot \max(|x| - \tau, 0)$ for a variable x.

With other variables fixed, we have the following optimization problem for finding P:

$$\begin{aligned} P_{+} &= \arg\min_{P} \text{tr}\left(\Lambda_1^T(D - P\widehat{M}P^T)\right) + \frac{\beta}{2}\|D - P\widehat{M}P^T\|_F^2, \\ &= \arg\min_{P} \frac{\beta}{2}\left\|D + \frac{\Lambda_1}{\beta} - P\widehat{M}P^T\right\|_F^2, \end{aligned} \tag{4.83}$$

subject to $P^TP = I_r$. The above problem is a least square problem with an orthogonality constraint. Let $R = D + \frac{\Lambda_1}{\beta}$ and $L = P\widehat{M}$, then L can be represented by $L = R(P^T)^+ = R(P^T)^T = RP$, where $(P^T)^+$ is the pseudo-inverse of the matrix P^T. Therefore, from [100], we can obtain the orthogonal matrix P using the QR factorization of L.

To update $\widehat{M}$, we consider the following equation:

$$\begin{aligned}\widehat{M}_+ = \arg\min_{\widehat{M}} \operatorname{tr}\left(\Lambda_1^T(D - P\widehat{M}P^T)\right) + \operatorname{tr}\left(\Lambda_2^T(\widehat{M} - M)\right) \\ + \frac{\beta}{2}\left(\|D - P\widehat{M}P^T\|_F^2 + \|\widehat{M} - M\|_F^2\right),\end{aligned} \tag{4.84}$$

and its solution is computed by taking a derivative as

$$\widehat{M}_+ = \frac{1}{2}\left(P^T DP + \frac{1}{\beta}P^T\Lambda_1 P + M - \frac{1}{\beta}\Lambda_2\right). \tag{4.85}$$

Finally, we update the Lagrange multipliers Λ_1 and Λ_2 as follows:

$$\begin{aligned}\Lambda_1 &\leftarrow \Lambda_1 + \beta(D - P\widehat{M}P^T), \\ \Lambda_2 &\leftarrow \Lambda_2 + \beta(\widehat{M} - M).\end{aligned} \tag{4.86}$$

The proposed structured kernel subspace learning algorithm is summarized in Algorithm 7. Since it is a symmetric positive semi-definite matrix factorization algorithm, it is named as FactSPSD. In the algorithm, we have assumed a normalized observation matrix. Hence, the output matrices are obtained by rescaling them using the scaling factor. The alternating minimization order of optimization variables can be different, but we have empirically found that the order given in Algorithm 7 shows better results than other orders. We set the initial values to all zero matrices since the algorithm is not sensitive to the choice of initial values. We set the parameters of the algorithm as $\lambda = 10^{-3}$, $\beta = 10^{-5}$, and $\rho = 2$. The number of inner iterations of the algorithm (lines 5–10) was set to 10 since it is enough to converge to a local solution. The stopping criterion (line 13 of Algorithm 1) is chosen as

$$\|D - P\widehat{M}P^T\|_1 < \epsilon \;\text{ or }\; \|\widehat{M} - M\|_1 < \epsilon, \tag{4.87}$$

and $\epsilon = 10^{-5}$, which shows good results in our experiments. Although it is difficult to guarantee the convergence to a local optimal solution, an empirical evidence

---

Algorithm 7 FactSPSD($K, r, \lambda, \beta, \rho$)

Input: $K \in \mathbb{R}^{n\times n}$, rank r, λ, β, and ρ
Output: $P \in \mathbb{R}^{n\times r}$ and $M \in \mathbb{R}^{r\times r}$
Initialization: $M = P = D = \widehat{M} = 0$ and $\beta_{max} = 10^{10}$
while not converged **do**
    while not converged **do**
        Update M by (4.80)
        Update $P \leftarrow QR(RP)$ where $R = D + \frac{\Lambda_1}{\beta}$
        Update $\widehat{M}$ by (4.85)
        Update D by (4.82)
    end while
    Update the Lagrange multipliers Λ_1 and Λ_2 by (4.86)
    Update $\beta = \min(\rho\beta, \beta_{max})$
    Check the convergence condition
end while

---

suggests that the proposed algorithm has a strong convergence behavior and converges with about 30 iterations of the outer loop.

Based on the structured low-rank approximation of a kernel matrix, we can derive a robust motion model using Gaussian process regression and it is shown in Algorithm 8. The algorithm is named as FactGP$_M$ since it is based on factorization-based low-rank kernel matrix approximation applied to Gaussian process regression. In Algorithm 8, we perform the standard PCA to the resulted low-rank kernel matrix L (line 7), to remove the inverse operation as in (4.4), reducing the computational complexity from $O(n^3)$ to $O(rn^2)$. We precompute the kernel matrix and its principal components in lines 4–8, and test a new input $\boldsymbol{x}_*$ given the principal components R in lines 10–11.

Algorithm 8 FactGP$_M$

Input: $X, \boldsymbol{y}$, rank r, and $\boldsymbol{x}_*$

Output: $\overline{\boldsymbol{y}}_*$

// Training

  Compute $\Lambda = K + \sigma_w^2 I$

  Perform kernel subspace learning:

    $[P, M] = \text{FactSPSD}(\Lambda, r, \lambda, \beta, \rho)$

  $L \leftarrow PMP^T$

  Compute R and Σ by performing PCA to L

// Testing

  Compute $k_* = k(\boldsymbol{x}_*, X)$

  Compute $\overline{\boldsymbol{y}}_*$ by (4.4)

4.3.3 Experimental results

In this section, we evaluate the performance of the proposed method, FactGP$_M$, by experimenting with various datasets and comparing with other well-known Gaussian process regression methods (SPGP[21] [101], PITC [102], GPLasso[22] [94], and PCGP-l_1 [97]) along with the standard GP. In our experiments, we used the radial basis kernel function for all GP methods and hyperparameters are learned using a conjugate gradient method [66]. The prediction or regression accuracy is measured by the root mean squared error (RMSE).

Regression problems

First, we tested the proposed structured low-rank matrix approximation method on a synthetic regression problem. We compared FactGP$_M$ to a sparse GP (PITC

[21] Available at `http://www.gatsby.ucl.ac.uk/~snelson/`.

[22] Available at `https://www.cs.purdue.edu/homes/alanqi/softwares/softwares.htm`.

[102]) and the full GP [66] to observe how different methods perform in the presence of corruptions.

Figure 4.18 shows the results from the regression problem with two outlier levels: no outliers and 20% outliers. We also compared the low-rank approximation methods, FactGP$_M$ and PITC, at two different ranks[23] (20% and 40% of the size of the kernel matrix). When there are no outliers, the full GP exactly fits the reference field but FactGP$_M$ and PITC show smooth lines with 20% low-rank components as shown in Figure 4.18(a). However, the low-rank approximation methods try to fit the reference field with the larger rank (40%) as shown in Figure 4.18(b). However, PITC still does not fit the reference very well as it misses some samples. The proposed method shows its competitiveness compared with the other GP methods in this regression problem. When we add outliers to randomly selected 20% of data as shown in Figure 4.18(c) and Figure 4.18(d), the full GP and PITC try to fit outliers, showing large fluctuations. But FactGP$_M$ is less affected by outliers, showing its robustness against outliers. From this experiment, we can see a clear benefit of the proposed low-dimensional learning method to a regression problem when the training set contains outliers.

We also tested the proposed method using real-world datasets, Pumadyn-8nm and Kin-8nm[24] [94]. Pumadyn-8nm is a dataset which consists of puma forward dynamics of eight inputs and Kin-8nm consists of the forward kinematics of an eight-link robot arm. For each dataset, we randomly collected 1,000 training and 800 test samples. To verify the robustness of the proposed method under the

[23] While PITC is a sparse GPR method, we treat it as a low-rank approximation method since the rank can be considered as a generalization of sparsity for two-dimensional data.

[24] Available at `http://www.cs.toronto.edu/~delve/methods/mars3.6-bag-1/mars3.6-bag-1.html`. Both datasets are frequently used to measure the performance of different Gaussian process regression methods.

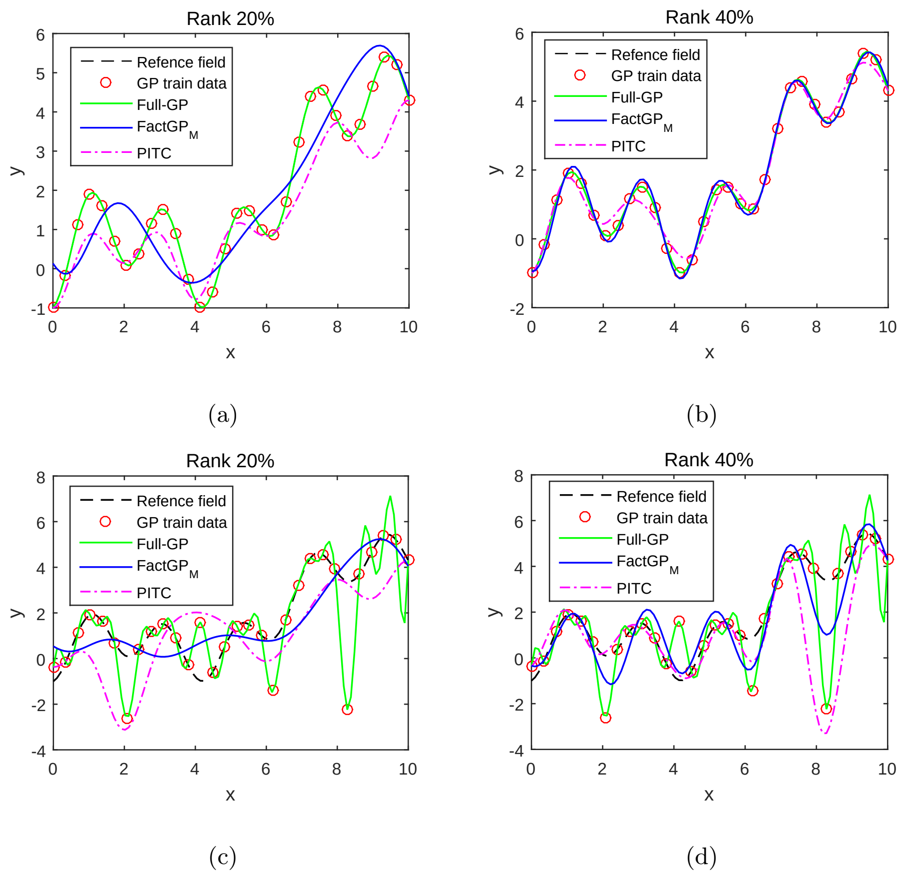
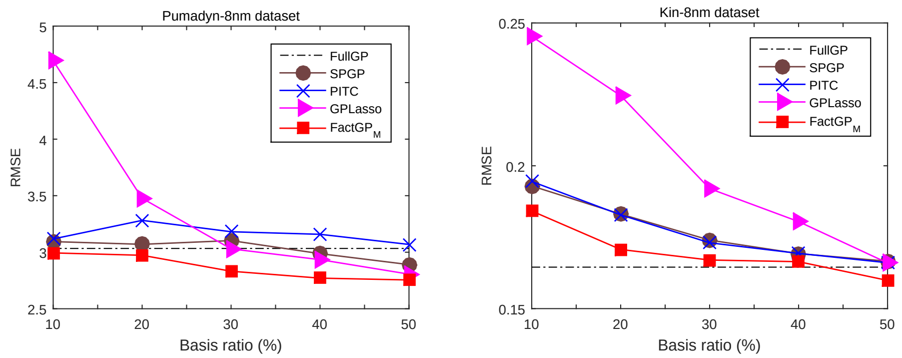


Figure 4.18: Simulation results on a synthetic example with and without outliers. FactGP$_M$ and PITC use kernel matrices whose ranks are either 20% or 40% of the size of the original kernel matrix. (a) No outliers with 20% low-rank. (b) No outliers with 40% low-rank. (c) 20% outliers with 20% low-rank. (c) 20% outliers with 40% low-rank.



(a) (b)

Figure 4.19: Regression results of the proposed method compared with other GP methods for two benchmark datasets: (a) Pumadyn-8nm, (b) Kin-8nm.

existence of various outliers, we added 30% outliers which are randomly selected from [-25, 25], whereas data values are usually in the range of [-2, 2]. The simulation results of the proposed method compared with other sparse GPR methods (SPGP [101], PITC [102], and GPLasso [94]) for various basis ratios (from 10% to 50%) are shown in Figure 4.19. As shown in Figure 4.19(a), the proposed method gives the lowest error among the methods regardless of the basis conditions. Especially, it shows better performance than the full GP, whereas sparse GPR methods show higher error than the full GP for some cases when the basis ratio is small. In Figure 4.19(b), the proposed method also gives lower errors than other sparse GPR methods.

Motion prediction of human trajectories

For the motion prediction experiment, we collected trajectories of moving pedestrians using a Pioneer 3DX differential drive mobile robot and a Microsoft Kinect

(a) (b) (c)

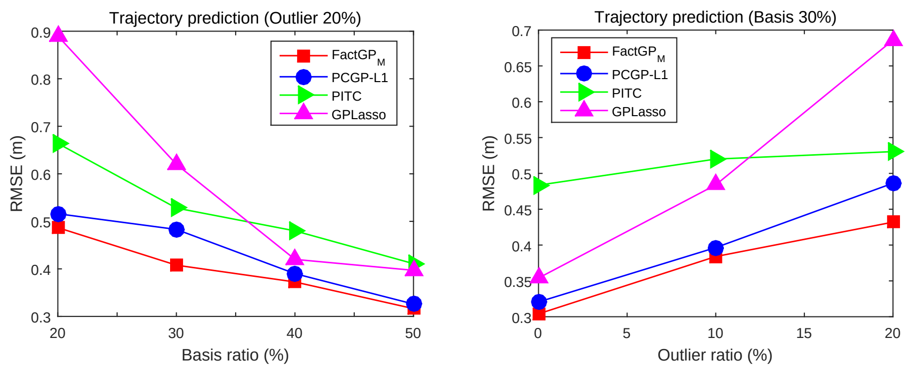
Figure 4.20: (a) A Pioneer 3DX mobile robot with two Kinect cameras and a notebook. (b) Snapshots from an experiment in a human-robot environment. First column: a third-person view. Second column: the egocentric view of a robot. (c) Collected trajectories from Kinect. We show a few trajectories in thick lines for better visualization.

camera,[25] which is mounted on top of the robot as shown in Figure 4.20(a). All algorithms are written in MATLAB with the mex-compiled ARIA package[26] on a notebook with a 2.5 GHz quad-core CPU and 8 GB RAM. The position of a pedestrian is detected using the skeleton grab API for Kinect.

We performed experiments in our laboratory to predict the future position of a person. To model the future positions of a pedestrian, our algorithm is applied to autoregressive Gaussian process (AR-GP) motion model [67]. Let $D_t \in \mathbb{R}^2$ be the position of a moving human at time t. The current velocity, $\Delta D_t = D_t - D_{t-1}$,

[25]For the motion prediction experiment, we collected human trajectories using one Microsoft Kinect camera and the experimental results are shown in Figure 4.21. But, for other experiments, we used two Kinect cameras to increase the field of view of the robot.

[26]Available at `http://robots.mobilerobots.com/wiki/ARIA`.



(a) (b)

Figure 4.21: Motion prediction simulation results using a Kinect camera based human trajectories: (a) Various basis ratio with 30 percent outliers. (b) Various outlier ratio with 30 percent basis vectors.

is modeled in AR-GP as follows [67], with an appropriate time scaling:

$$\begin{aligned} \Delta D_t &= f(D_{t-1}, D_{t-2}, \cdots, D_{t-p}) \\ &\sim GP_f(D_{t-1}, D_{t-2}, \cdots, D_{t-p}). \end{aligned} \tag{4.88}$$

Hence, the AR-GP motion model can find the position of a pedestrian at time t based on p recent positions of the pedestrian with this nonlinear model of an autoregressive process under the Gaussian process framework.

Figure 4.20(b) shows snapshots from the third-person view and the egocentric view from a robot. We collected a diverse set of trajectories of pedestrians and obstacles, which are in the field of view of a robot as shown in Figure 4.20(c). To make a training set from the collected trajectories, we uniformly sampled positions to have about ten samples in a trajectory when a trajectory has many detected positions. From a trajectory which has n positions, we obtain $n-p+1$ input samples where p is the order of an autoregressive motion model, i.e., the number of past positions. One can model it as a Hankel matrix by shifting one point in a trajectory.

We compared the proposed method, FactGP$_M$, with the state-of-the-art approaches (PCGP-l_1 [97], GPLasso [94], and PITC [102]). We divided the collected trajectories into training and test sets with autoregressive order $p = 3$. Using the dataset, we experimented for two cases: (1) various rank (basis) conditions with a fixed outlier ratio and (2) various outlier conditions with a fixed rank. We added outliers to randomly selected positions of collected trajectories from $[-10, 10]$, whereas the datasets are in the range of $[-5, 5]$. Figure 4.21 shows prediction errors by tested algorithms for two cases. As shown in Figure 4.21(a), the proposed FactGP$_M$ shows the best results compared to other methods in all cases. PCGP-l_1 gives the second best results regardless of the basis ratios. We can interpret that the proposed algorithm approximates the positive semi-definite (PSD) matrix better than PCGP-l_1, since the proposed algorithm can guarantee the positive semi-definiteness, whereas PCGP-l_1 cannot. The RMSE error results for a fixed rank ($r/n \times 100 = 30\%$) under various outlier conditions are shown in Figure 4.21(b). As shown in the figure, the proposed method gives the best results regardless of outlier conditions. From two figures, we can see that the proposed method shows the robustness against outliers, by recovering from measurement noises and erroneous trajectories. Figure 4.22 shows some snapshots from the motion prediction experiment using two Microsoft Kinect cameras (field of view of about 110°) in our laboratory. The robot performed the nearly exact prediction of the future positions of pedestrians in real-time.

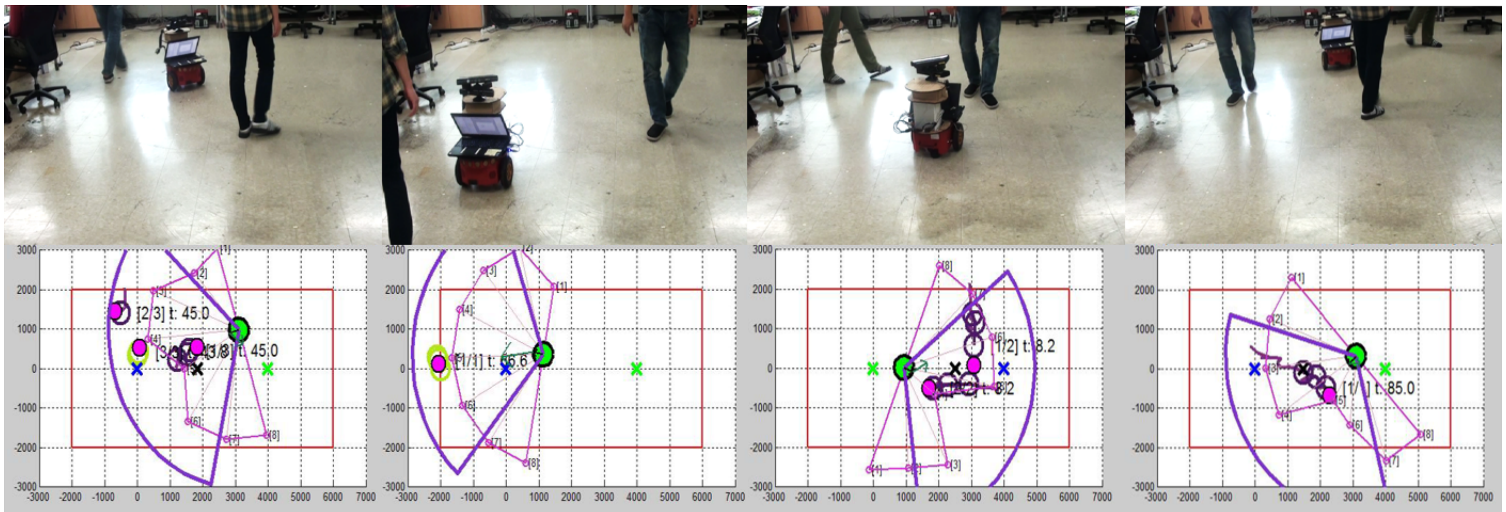


Figure 4.22: Motion prediction experiments using the proposed motion model, FactGP$_M$. A pink circle is the prediction made by FactGP$_M$ given past pedestrian positions (purple or yellow-green circles). The violet fan-shaped region is the field of view of two Kinect sensors and the pink fan-shaped region shows sensing responses from sonar sensors of the Pioneer robot. Each column consists of a photo taken by a camera and the internal state of the robot. Best viewed in color.

4.4 Summary

In this chapter, we have proposed several low-rank representation from unstructured matrix approximation to structured approximation. We have first proposed two novel gradient-based methods, l_1-ARG$_A$ and l_1-ARG$_D$, using the alternating rectified gradient method. For the dual method l_1-ARG$_D$, we have proved the convergence of the algorithm to the subspace-wise local minimum using the global convergence theorem. We have shown the superiority of the proposed methods compared to existing algorithms for large-scale problems.

To overcome the previous unregularized algorithms, we have also proposed a method, ROMF, for efficient fixed-rank factorization with the Frobenius-norm regularizer and orthogonality constraint. ROMF is constructed under the augmented Lagrangian framework and can address the rank uncertainty issue by a rank estimation strategy for practical real-world problems. The experimental results have shown that ROMF outperforms other existing methods including l_1-ARG methods in terms of the approximation error and running time.

Lastly, we have presented a novel optimization formulation for a structured matrix which is generally symmetric positive semi-definite matrix and finds low-rank kernel subspace by minimizing a nuclear-norm regularized l_1-norm objective function. The proposed method is applied to various regression and motion prediction problems in real-world environments. The experimental results have shown the efficiency and robustness of the proposed method against outliers and measurement errors.

Chapter 5

Robust Lower-Rank Subspace Representations

[1]In this chapter, our goal is to develop a robust and stable algorithm for finding subspace structures of grossly corrupted data. For this objective, we propose elastic-net subspace representation based on elastic-net regularization of singular values of data. The elastic-net method embraces the benefits of both lasso and ridge regression methods [104, 105, 106, 103], such as automatic variable selection, continuous shrinkage and thresholding, and selection of groups of correlated variables. We show that the propose framework allows more stable and efficient algorithms for subspace representation in the presence of corruptions or missing entries, due to the strong convexity enforced by the elastic-net regularization.

It is worthwhile to note that while both the proposed method and our main competitor, lasso-based method [22], use an alternative definition of the nuclear-

[1]This chapter is based on the following papers:
"Elastic-Net Regularization of Singular Values for Robust Subspace Learning,", *CVPR* [19],
"Robust Elastic-Net Subspace Representation", *IEEE TIP* [103].

norm regularizer in order to speed up the algorithms, there are clear differences. First, the proposed framework is more general than [22] for rank-related problems, since it can further shrink the singular values under the fixed-rank constraint by introducing strong convex regularizer, whereas [22] does not perform shrinkage as it simply employs the alternative variation of the nuclear-norm regularizer, which makes [22] unstable in the presence of corruptions and produces incorrect results (see Figure 5.1 for an example). Second, it is possible for the proposed method to conduct automatic rank estimation by shrinking and suppressing singular values from the maximum user-defined rank based on the elastic-net regularization of singular values, whereas it is difficult to conduct elaborate rank estimation using [22], making it less applicable in practice.

Based on the proposed elastic-net subspace representation framework, we propose two algorithms: *FactEN* and *ClustEN*. FactEN solves a low-rank subspace learning problem, where data lie in a single low-dimensional subspace, for rank-specific problems [13, 14, 12]. It is a holistic approach to deal with both bilinear factorization and rank minimization using elastic-net regularization. ClustEN is a joint optimization algorithm to solve a general problem, in which data are drawn from a union of subspaces. It jointly solves subspace clustering and subspace learning. The advantages of the elastic-net subspace representation algorithms compared to the state-of-the-art subspace representation algorithms are demonstrated in an extensive set of experiments.

5.1 Elastic-Net Subspace Representation

The methods described in the previous section solve various instances of subspace representation problems. Given an observation matrix $Y = [\boldsymbol{y}_1, ..., \boldsymbol{y}_n] \in \mathbb{R}^{m \times n}$, where samples are drawn from a single subspace or a union of multiple subspaces, the goal of subspace representation is to find the underlying subspace structure of an observation data.

In this chapter, we propose a new approach to solve various subspace representation problems using elastic-net regularization. The general framework of our proposal, where data samples are assumed to be drawn from a union of multiple subspaces, can be formulated as the following optimization problem under noisy scenarios to learn a dictionary or clean matrix D, an error matrix E, and a subspace representation matrix C, simultaneously:

$$\min_{D,E,C} f_W(E) + \lambda \Omega_{EN}(D, C), \quad \text{s.t. } D, E, C \in \mathcal{C}_{EN}, \tag{5.1}$$

where $f_W(E) = \|W \odot E\|_1$ is a weighted l_1-norm loss function to handle outliers, occlusions, and missing entries, and W is a weighting matrix, whose element w_{ij} is 1 if y_{ij} is known, and 0 if y_{ij} is unknown. When there are no missing entries, we can also solve the problem by setting the values of all elements of W to one. $\Omega_{EN}(D, C)$ and $\mathcal{C}_{EN}$ are defined as

$$\Omega_{EN}(D, C) = \|D\|_* + \alpha \|D\|_F^2 + \beta \|C\|_1, \tag{5.2}$$

$$\mathcal{C}_{EN} = \{D, C, E \mid Y = D + E, D = DC, \text{diag}(C) = 0\}.$$

Here, Ω_{EN} consists of the elastic-net regularization over singular values of D and a subspace representation matrix C to represent the subspace membership by sparse representation, and $\mathcal{C}_{EN}$ is used to enforce the low-rank and noise matrices separation from the observation matrix Y and self-expressiveness of D. $\odot$ is the component-wise multiplication or the Hadamard product.

From the subspace representation problem (5.1), we can consider an important special case, in which data samples are drawn from a single subspace with fixed basis vectors (or fixed-rank) by constraining $C = I$, where I is the identity matrix.

Then, (5.1) can be reduced to the following problem:

$$\min_{D,E} \ f_W(E) + \lambda\overline{\Omega}_{EN}(D), \quad \text{s.t. } D, E \in \overline{\mathcal{C}}_{EN}, \tag{5.3}$$

where $\overline{\Omega}_{EN}(D) = \|D\|_* + \alpha\|D\|_F^2$ and $\overline{\mathcal{C}}_{EN} = \{D, E \mid Y = D+E, \text{rank}(D) = r\}$. In this problem, we enforce the rank of D to r. A fixed-rank approximation problem appears frequently in rank-related applications, such as background modeling [12], structure from motion [13], and photometric stereo [14]. The detailed analysis of this problem (5.3) will be the focus of the next section.

In order to compare with other subspace representation algorithms, we can consider the following general form:

$$\min_{V} \ f_{loss}(V) + \lambda\Omega_{reg}(V), \quad \text{s.t. } V \in \mathcal{C}, \tag{5.4}$$

where f_{loss}, Ω_{reg}, and $\mathcal{C}$ are a loss function, regularization function, and constraint set, respectively. V is a set of optimization variables. By substituting terms in (5.4), we can represent different problems, such as low-rank matrix factorization [10], sparse and low-rank matrix separation [35], and subspace clustering [15], to name a few. For example, with the following substitutions in (5.4), we have RPCA [35].

$$f_{loss} = \|E\|_1, \Omega_{reg} = \|D\|_*, \mathcal{C} = \{D, E \mid Y = D + E\}, \tag{5.5}$$

where Y is an observation matrix and D and E are optimization variables. Table 5.1 shows the comparison of well-known subspace learning and clustering problems including the proposed subspace representation algorithms according to the loss function, regularizer, and constraint set. The main difference between the proposed algorithms and existing methods is that ours are based on singular value analysis using the elastic-net regularization to estimate exact singular values with their corresponding singular vectors and reconstruct a clean low-rank matrix from a corrupted observation.

Table 5.1: Comparison of the cost functions of the existing subspace learning and clustering algorithms used in this work including the proposed methods with respect to loss function f_{loss}, regularizer Ω_{reg}, and constraint set $\mathcal{C}$.

Algorithms	f_{loss}	Ω_{reg}	$\mathcal{C}$
[22]	$\\|W \odot E\\|_1$	$\\|D\\|_*$	$\{D, E \mid Y = D + E, \mathrm{rank}(D) = r\}$
RPCA [35]	$\\|E\\|_1$	$\\|D\\|_*$	$\{D, E \mid Y = D + E\}$
SSC [16]	$\\|E\\|_1$	$\\|C\\|_1$	$\{C, E \mid Y = YC + E, \mathrm{diag}(C) = 0\}$
LRR [75]	$\\|E\\|_{2,1}$	$\\|C\\|_*$	$\{C, E \mid Y = YC + E\}$
LRSC [107]	$\\|E\\|_1$	$\\|C\\|_*$	$\{D, C, E \mid Y = D + E, D = DC\}$
Ours_1 (Section 5.2)	$\\|W \odot E\\|_1$	$\\|D\\|_* + \alpha\\|D\\|_F^2$	$\{D, E \mid Y = D + E, \mathrm{rank}(D) = r\}$
Ours_2 (Section 5.3)	$\\|W \odot E\\|_1$	$\\|D\\|_* + \alpha\\|D\\|_F^2 + \beta\\|C\\|_1$	$\{D, C, E \mid Y = D + E, D = DC, \mathrm{diag}(C) = 0\}$

In the subsequent sections, we will give detailed analysis for the proposed algorithms, formulated in (5.1) and (5.3).

5.2 Robust Elastic-Net Subspace Learning

5.2.1 Problem formulation

In this section, we first address a low-rank and sparse matrices separation problem [35, 22], considering missing entries in an observation matrix, based on convex envelopes of rank and sparsity functions as follows:

$$\min_{D} \; f_1(D) + \lambda\Omega_{reg}, \quad \text{s.t. } D \in \mathcal{C}_0, \tag{5.6}$$

where $f_1(D) = \|W \odot (Y - D)\|_1$ and $\Omega_{reg} = \|D\|_*$. Here, $\mathcal{C}_0 = \emptyset$. $\|\cdot\|_1$ and $\|\cdot\|_*$ denote the entry-wise l_1-norm and the nuclear-norm, which are convex relaxation[2] of the l_0-norm and the rank function, respectively. Note that the regularization term in (5.6), $\|D\|_*$, can be interpreted as a sum of singular values, $\sum_i^r |\sigma_i|$, where σ_i is the ith singular value of a low-rank matrix D and r is the rank of D. The nuclear-norm based subproblem in (5.6) can be solved by singular value thresholding [108], which has both thresholding and shrinkage effect over singular values of D.

Here, we would like to note that the problem (5.6) can find a suboptimal solution where the rank of the target matrix is pre-defined as a constant, such as structure from motion [13], background modeling [12], and photometric stereo [14]. Furthermore, there is an issue in regard to the computational complexity due to the SVD operation performed at each iteration to solve a nuclear-norm

[2]Since a problem based on the l_0-norm or rank function is NP-hard, a convex surrogate of the function is used in practice.

based cost function. In order to address these issues efficiently, one can consider the following property of the nuclear-norm [87]:

Lemma 2 ([87])**.** *For any matrix $D \in \mathbb{R}^{m\times n}$, the following holds:*

$$\|D\|_* = \min_{P,X} \frac{1}{2}\left(\|P\|_F^2 + \|X\|_F^2\right) \quad s.t. \quad D = PX. \tag{5.7}$$

If the rank of D is $r \leq \min(m,n)$, then the minimum solution above is attained at a factor decomposition $D = P_{m\times r}X_{r\times n}$.

Using Lemma 2, we make an equivalent form of (5.6) as follows:

$$\min_{P,X,D} f_1(D) + \frac{\lambda}{2}\left(\|P\|_F^2 + \|X\|_F^2\right), \quad \text{s.t. } D \in \mathcal{C}, \tag{5.8}$$

where $\mathcal{C} = \{D, P, X \mid D = PX\}$. However, by using the lemma, we have lost the effect of shrinkage since the singular value thresholding operation is no longer needed. Even though we have lost the effect of thresholding, the effect remains in the problem by fixing the rank to r. Moreover, (5.8) is a lasso-based approach which has weak convexity and, hence, can make an iterative minimization routine unstable when highly corrupted data are presented. To improve the stability of the algorithm and give the shrinkage and thresholding effects on the singular values of D, we introduce a strong convex regularizer for the original cost function (5.6)[3] using the l_2-norm penalty of singular values:

$$\min_{D} \ f_1(D) + \lambda\overline{\Omega}_{EN}(D), \quad \text{s.t. } D \in \mathcal{C}, \tag{5.9}$$

where $\overline{\Omega}_{EN}(D) = \|D\|_* + \frac{\alpha}{2}\|D\|_F^2$. Although (5.9) is slightly modified from (5.3), they are equivalent. Using the fact that $\|D\|_* = \sum_i |\sigma_i|$ and $\|D\|_F^2 =$

[3]Here, we first give an equivalent form of the original problem (5.6), for a while, instead of (5.8), to analyze the problem from a theoretical perspective.

$\mathrm{tr}(V\Sigma U^T U\Sigma V^T) = \mathrm{tr}(\Sigma^2) = \sum_i |\sigma_i|^2$, where $D = U\Sigma V^T$ is SVD of D, we have the following equivalent form to (5.9):

$$\min_{D} \; f_1(D) + J_{\lambda_1,\lambda_2}(\Sigma), \tag{5.10}$$

where

$$J_{\lambda_1,\lambda_2}(\Sigma) \triangleq \lambda_1 \sum_i^r |\sigma_i| + \frac{\lambda_2}{2} \sum_i^r |\sigma_i|^2, \tag{5.11}$$

$\lambda_1 = \lambda$, and $\lambda_2 = \alpha\lambda$.

In (5.11), we have elastic-net regularization of singular values of D, which has shown its superiority compared to lasso [34] in many applications [104, 106, 105]. It is capable of stabilizing a lasso-type method due to its strong convexity, owing to the Frobenius norm [104, 106, 109]. By incorporating with Lemma 2, we have the following equivalent formulation of (5.9):

$$\min_{P,X,D} \; f_1(D) + \frac{\lambda_1}{2}\left(\|P\|_F^2 + \|X\|_F^2\right) + \frac{\lambda_2}{2}\|D\|_F^2, \tag{5.12}$$

and it has both a thresholding effect over singular values from the alternative definition of Lemma 2[4] and a shrinkage effect from the l_2 regularizer to make a parsimonious and stable model. In summary, we can achieve both thresholding and shrinkage effects without performing SVD by introducing a strong convex regularizer, called elastic-net, to accelerate the computation speed and stably solve problems.

Note that, without these regularization terms, the problem (5.11) can be solved using the augmented Lagrangian alternating direction method (ALADM) [11]. There is another approach using a nuclear-norm regularized l_1-norm cost function [49]. It is extended using the alternative definition of the nuclear-norm given in

[4] Actually, it also gives a hard thresholding effect due to the matrix factorization by the pre-defined rank.

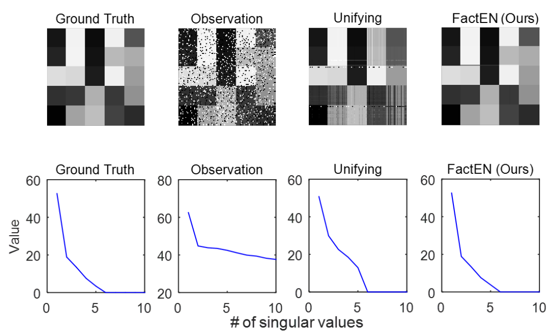
Lemma 2 (Unifying[5]) [22], which does not contain the smoothness term given in (5.12). However, these methods can find a suboptimal solution since these alternating minimization based approaches with weak convexity may lead to a poor solution in the presence of highly corrupted data (see Section 5.4.1). Figure 5.1 shows results of the proposed method compared to Unifying [22], a lasso-based method, and ground-truth on a simple example (100 $\times$ 100) with 20% outliers. The rank of the ground-truth is five. From the figure, the proposed method gives a stable result against outliers and eliminates noises by suppressing the singular values, whereas Unifying finds relatively inaccurate and larger singular values and shows a poor reconstruction result compared to the proposed method and the ground-truth.

In general, the problem (5.12) with the low-rank constraint $D = PX$ is a non-convex and non-smooth problem, making it difficult to find a solution efficiently and exactly. To solve the problem efficiently, a common strategy is to use an alternating minimization approach which solves for one variable while other variables are fixed [10]. Hence, we give an equivalent formulation of (5.11) by introducing an auxiliary variable $\widehat{D}$ and solve the following problem instead.

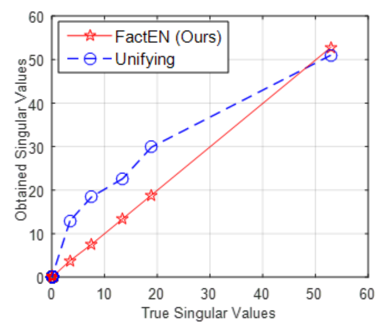
$$\begin{aligned} \min_{P,X,D,\widehat{D}} \; & f_1(\widehat{D}) + \frac{\lambda_1}{2}\left(\|P\|_F^2 + \|X\|_F^2\right) + \frac{\lambda_2}{2}\|D\|_F^2 \\ \text{s.t.} \;\; & D = PX, \quad \widehat{D} = D. \end{aligned} \tag{5.13}$$

To solve (5.13), we utilize the augmented Lagrangian framework which converts (5.13) into an unconstrained problem with Lagrange multipliers $\Lambda_1, \Lambda_2 \in \mathbb{R}^{m \times n}$.

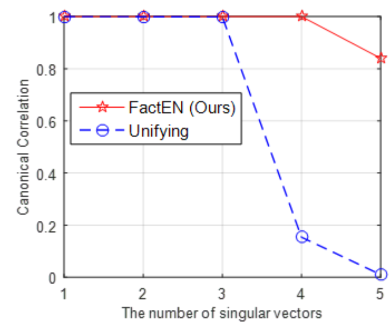
[5] We call the method in [22] as Unifying for simplicity.



(a)



(b)



(c)

Figure 5.1: Evaluation of the proposed subspace learning method (FactEN) and a lasso-based method (Unifying [22]) for a toy example. (a) Data matrices and corresponding 10 largest singular values from the ground truth and noisy observation are shown in two left columns. Recovered matrices and corresponding 10 largest singular values by Unifying and FactEN are shown in two right columns. (b) Ordered singular values recovered by two algorithms against true singular values. (c) Canonical correlations between singular vectors from applied algorithms and true singular vectors.

5.2.2 Algorithm: FactEN

Algorithm

Based on the previous formulation, we develop a method based on the augmented Lagrangian framework and solve it using an alternating minimization technique [11]. To solve for P, we fix the other variables and solve the following optimization problem:

$$P_+ = \arg\min_P \ \lambda_1 \|P\|_F^2 + \beta \|D - PX + \frac{\Lambda_1}{\beta}\|_F^2, \tag{5.14}$$

where $\beta > 0$ is a small penalty parameter. This optimization problem is a least square problem and the solution is

$$P_+ = (\Lambda_1 + \beta D) X^T (\lambda_1 I + \beta X X^T)^{-1}, \tag{5.15}$$

where I denotes an identity matrix. Similar to (5.14), X and D can be solved as follows:

$$X_+ = (\lambda_1 I + \beta P^T P)^{-1} P^T (\Lambda_1 + \beta D), \tag{5.16}$$

$$D_+ = \frac{\beta PX + \beta \widehat{D} + \Lambda_2 - \Lambda_1}{\lambda_2 + 2\beta}. \tag{5.17}$$

We obtain the following equation to solve for $\widehat{D}$,

$$\widehat{D} = \arg\min_{\widehat{D}} f_1(\widehat{D}) + \mathrm{tr}\left(\Lambda_2^T (\widehat{D} - D)\right) + \frac{\beta}{2} \|\widehat{D} - D\|_F^2, \tag{5.18}$$

and the solution can be computed using the absolute value thresholding operator [43, 35, 49]:

$$\begin{cases} W \odot \widehat{D}_+ = W \odot \left(Y - \mathcal{S}\left(Y - D + \frac{\Lambda_2}{\beta}, \frac{1}{\beta} \right) \right), \\ \overline{W} \odot \widehat{D}_+ = \overline{W} \odot \left(D - \frac{\Lambda_2}{\beta} \right), \end{cases} \tag{5.19}$$

Algorithm 9 FactEN by ALM for optimizing (5.13)

1: **Input:** $Y \in \mathbb{R}^{m \times n}$, r, β, ρ, and $\lambda_1, \lambda_2 = 10^{-3}$
2: **while** not converged **do**
3:     **while** not converged **do**
4:         Update $P, X, D, \widehat{D}$, respectively
5:     **end while**
6:     Update the Lagrange multipliers Λ_1, Λ_2 using (5.20)
7:     $\beta = \min(\rho\beta, \beta_{max})$
8: **end while**
9: **Output:** $P \in \mathbb{R}^{m \times r}$, $X \in \mathbb{R}^{r \times n}$, and $D \in \mathbb{R}^{m \times n}$

where $\mathcal{S}(x, \tau) = \text{sgn}(x) \max(|x| - \tau, 0)$ for a variable x and $\overline{W} \in \mathbb{R}^{m \times n}$ is a complementary matrix of W whose element $\overline{w}_{ij}$ is 0 if y_{ij} is known, and is 1 if y_{ij} is unknown.

Finally, we update the Lagrange multipliers as

$$\Lambda_{1+} = \Lambda_1 + \beta(D - PX), \quad \Lambda_{2+} = \Lambda_2 + \beta(\widehat{D} - D). \tag{5.20}$$

Based on the previous analysis, we derive a robust elastic-net regularized low-rank matrix factorization algorithm and it is summarized in Algorithm 9. Since the algorithm is constructed based on elastic-net regularization and solved using a matrix factorization approach, the proposed method is named as *FactEN*. In the algorithm, we have assumed a normalized observation matrix. Thus, the output matrices P and X can be later re-scaled based on initial scaling factor. We initialize the optimization variables with the Gaussian distribution $\mathcal{N}(0, 10^{-3})$.[6]

The computational complexity of the inner loop (line 4 in Algorithm 9) is

[6]Note that we have empirically found that our algorithm is not sensitive to initial values and finds similar solutions with different initial values.

$O(mnr)$ for the proposed method, where m, n, and r denote dimensionality, sample size, and rank, respectively, which is the same as those of Unifying [22] and ALADM [11]. Since IALM [43] and Regl_1-ALM [49] perform an SVD operation at every iteration, their computational complexities are $O(\min(m,n)\max(m,n)^2)$ and $O(r\max(m,n)^2)$, respectively, requiring more computational efforts than FactEN. In the algorithm, we can choose β_{max} by following several works [43, 49] (e.g., 10^{20}) as a real-valued choice of β for the positive infinite number or very high upper bound. However, since our algorithm converges within a small number of iterations (see Figure 5.2), the exact value does not influence the performance of the proposed method.

Note that the proposed method can be easily extended to speed up the algorithm with linear complexity at each iteration by sampling sub-matrices from a measurement matrix as described in [12].

Convergence analysis of FactEN

In this section, we analyze the convergence property of the proposed method. Although it is difficult to guarantee its convergence to a local minimum, an empirical evidence suggests that the proposed algorithm has a strong convergence behavior (see Figure 5.2). Nevertheless, we provide a proof of weak convergence of FactEN by showing that under mild conditions any limit point of the iteration sequence generated by the algorithm is a stationary point that satisfies the Karush-Kuhn-Tucker (KKT) conditions [110]. The KKT conditions are first order conditions to be an optimal solution in constrained optimization problems. It is worth proving that any converging point satisfies the KKT conditions because they are necessary conditions to be a local optimal solution and give the minimum guarantee about the convergence behavior of an algorithm when it is nonconvex and thus difficult to show the complete convergence. This result provides an assurance about the behavior of the proposed algorithm.

We rewrite the cost function of FactEN by assuming the fully-observed data model of (5.13), i.e., $W_{ij} = 1$ for all i, j, as follows:

$$\begin{aligned} \min_{P,X,D,\widehat{D}} \; & f_2(\widehat{D}) + \frac{\lambda_1}{2}\left(\|P\|_F^2 + \|X\|_F^2\right) + \lambda_2\|D\|_F^2 \\ \text{s.t.} \quad & D = PX, \; \widehat{D} = D. \end{aligned} \tag{5.21}$$

where $f_2(\widehat{D}) = \|Y - \widehat{D}\|_1$. However, a similar result can be derived for the partially-observed data model.

Let us assume that the proposed algorithm reaches a stationary point. The KKT conditions for (5.21) are derived as follows:

$$\begin{aligned} & D - PX = 0, \; \widehat{D} - D = 0, \; \frac{\partial \mathcal{L}}{\partial P} = \lambda_1 P - \Lambda_1 X^T = 0, \\ & \frac{\partial \mathcal{L}}{\partial X} = \lambda_1 X - P^T \Lambda_1 = 0, \; \frac{\partial \mathcal{L}}{\partial D} = \lambda_2 D + \Lambda_1 - \Lambda_2 = 0, \\ & \Lambda_2 \in -\partial_{\widehat{D}}(||Y - \widehat{D}||_1). \end{aligned} \tag{5.22}$$

Here, we can obtain the following equation from the the last relationship in (5.22):

$$\begin{aligned} Y - D + \frac{\Lambda_2}{\beta} & \in Y - D - \frac{1}{\beta}\partial_{\widehat{D}}(||Y - \widehat{D}||_1) \\ & = Y - \widehat{D} - \frac{1}{\beta}\partial_{\widehat{D}}(||Y - \widehat{D}||_1) \triangleq Q_\beta(Y - \widehat{D}), \end{aligned} \tag{5.23}$$

where scalar function $Q_\beta(t) \triangleq t - \frac{1}{\beta}\partial|t|$ is applied element-wise to $Y - \widehat{D}$. From [11], we can obtain the following relation:

$$Y - \widehat{D} = Q_\beta^{-1}\left(Y - D + \frac{\Lambda_2}{\beta}\right) \equiv \mathcal{S}\left(Y - D + \frac{\Lambda_2}{\beta}, \frac{1}{\beta}\right), \tag{5.24}$$

where $\mathcal{S}(x, \tau) = \mathrm{sgn}(x)\max(|x| - \tau, 0)$. Based on these conditions, we prove the convergence to a point which satisfies the KKT conditions.

Theorem 2. *Let* $G \triangleq (P, X, D, \widehat{D}, \Lambda_1, \Lambda_2)$ *and* $\{G^j\}_{j=1}^{\infty}$ *be generated by FactEN. Assume that* $\{G^j\}_{j=1}^{\infty}$ *is bounded and* $\lim_{j\to\infty}\{G^{j+1} - G^j\} = 0$*. Then, any accumulation point of* $\{G^j\}_{j=1}^{\infty}$ *satisfies the KKT conditions. In particular, whenever* $\{G^j\}_{j=1}^{\infty}$ *converges, it converges to a KKT point.*

Proof. See Appendix E □

In our algorithm, we set the stopping criterion as

$$\frac{||D^{(t)} - P^{(t)}X^{(t)}||_1}{||Y||_1} < \theta, \tag{5.25}$$

where t is the number of iterations and θ is a small positive number. Since it is enough for the algorithm to achieve a nearly stationary point when the difference between the terminating cost of adjacent iterations becomes small, we set the stopping condition as $\theta = 10^{-5}$ in our experiments in Section 5.4.1. Figure 5.2 shows scaled cost values[7] of the proposed method at each iteration for four examples from 500 × 500 to 3,000 × 3,000 with outliers as described in Section 5.4.1. Each point denotes a cost value at each iteration. As shown in the figure, the cost value of FactEN decreases fast and converges to a stationary point in a small number of iterations.

[7]We have scaled cost values as $(f_1(\widehat{D}) + \frac{\lambda_1}{2}(\|P\|_F^2 + \|X\|_F^2) + \frac{\lambda_2}{2}\|D\|_F^2)/\|W \odot Y\|_1$ in order to display four cases under the same scale.

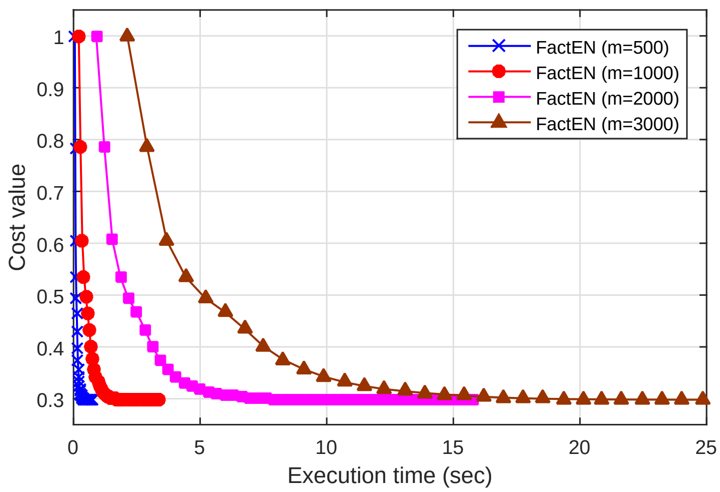


Figure 5.2: Scaled cost values of the proposed algorithm at each iteration for four synthetic examples. The termination of the cost value means the algorithm reaches to a stationary point in the cost function, which gives an empirical justification, showing that the proposed algorithm converges to an accumulation point.

5.3 Joint Subspace Estimation and Clustering

5.3.1 Problem formulation

The subspace learning method described in the previous section only considers a single subspace and cannot be applied to datasets, in which data samples are drawn from a union of multiple subspaces. Hence, in this section, we consider the general elastic-net subspace representation framework given in (5.1). Handling a union of multiple subspaces is closely related to a subspace clustering problem [15], where the goal is to estimate the structure of multiple subspaces by a method based on a user-defined regularizer, such as the l_1-norm for sparse representation [16], the nuclear-norm for low-rank representation [4], and the Frobenius-norm for least square regression [111]. While there are many algorithms to identify the exact structure under noiseless scenarios [16, 4], it is still difficult to find the precise structure under grossly corrupted scenarios. As a remedy of the issue, we propose a new joint optimization framework handling both subspace learning and clustering under the presence of corruptions.

The problem formulation of the unified framework for both subspace learning and clustering in the presence of corruptions is as follows:

$$\min_{D,C} \; f_1(D) + \lambda\Omega_{EN}(D,C), \quad \text{s.t. } D, C \in \mathcal{C}_{EN}, \tag{5.26}$$

where $\Omega_{EN}(D,C) = \overline{\Omega}_{EN}(D) + \beta\Omega_C(C)$ and $C \in \mathbb{R}^{n\times n}$ is a latent matrix to reveal the structure of multiple subspaces. $\Omega_C(C) = \|C\|_1$ and β is a weighting parameter. Here, $\mathcal{C}_{EN} = \{D, C \mid D = DC,\ \mathrm{diag}(C) = 0\}$. The last constraint in $\mathcal{C}_{EN}$, $\mathrm{diag}(C) = 0$, is used to avoid a trivial solution, i.e., $C = I$, where I is the identity matrix. The problem (5.26) can be reduced to the problem (5.9) when we ignore C and enforce the rank constraint for D. In (5.26), we jointly learn the outlier-reduced low-rank matrix D and the subspace representation matrix C. A similar approach to (5.26) is low-rank subspace clustering (LRSC) [107], which pursues both subspace estimation and clustering in the presence of outliers.

Notice that we do not factorize the data matrix into basis and coefficient matrices, unlike FactEN in Section 5.2, since the rank of a subspace clustering problem is generally unknown or difficult to estimate reliably. Hence, we do not apply the Lemma 2 to (5.26), which means that we do not obtain the computation advantage for this problem. But, the effect of the elastic-net regularization is still valid for subspace clustering since the elastic-net over singular values is used in the joint optimization procedure to find a noise-reduced data stably in the presence of corruptions.

Let us consider a case where all data are observable. But, we can easily extend to a scenario with missing data. The equivalent problem of (5.26) for a non-missing scenario with two auxiliary variables $\breve{D}$ and J is as follows:

$$\begin{aligned} &\min_{M,D,\breve{D},J,C} f_c(M) + \lambda_1 \|D\|_* + \frac{\lambda_2}{2} \|\breve{D}\|_F^2 + \lambda_3 \|J\|_1 \\ &\text{s.t.} \quad \breve{D} = M, M = \breve{D}C, J = C, D = \breve{D}, \operatorname{diag}(J) = 0, \end{aligned} \tag{5.27}$$

where $f_c(M) = \|Y - M\|_1$ and $\lambda_3 = \beta\lambda$.

5.3.2 Algorithm: ClustEN

From the above formulation, we derive another method based on the augmented Lagrangian framework with Lagrange multipliers Π_1, Π_2, Π_3, and Π_4. and solve it using the alternating minimization approach of optimization variables as discussed in the previous subspace learning section.

To solve for M, we have the following problem:

$$\begin{aligned} M_+ = &\min_M f_c(M) + \\ &\frac{\gamma}{2}\left(\|\breve{D} - M + \frac{\Pi_1}{\gamma}\|_F^2 + \|M - \breve{D}C + \frac{\Pi_2}{\gamma}\|_F^2 \right), \end{aligned} \tag{5.28}$$

where $\gamma > 0$ is a small penalty parameter and its solution is computed by an absolute value shrinkage operator [43]:

$$M_{+} = Y - \mathcal{S}\left(Y - K, \frac{1}{2\gamma}\right), \tag{5.29}$$

where $K = \frac{1}{2\gamma}(\gamma(\breve{D} + \breve{D}C) + \Pi_1 + \Pi_2)$ and $\mathcal{S}(x,\tau) = \mathrm{sgn}(x)\max(|x| - \tau, 0)$.

To find D, we have the following problem:

$$D_{+} = \min_{D} \lambda_1 \|D\|_* + \frac{\gamma}{2}\left\|D - \breve{D} + \frac{\Pi_4}{\gamma}\right\|_F^2, \tag{5.30}$$

which can be solved by singular value shrinkage [43, 35]

$$D_{+} = U_D \mathcal{S}_{\tau}(S_D) V_D, \tag{5.31}$$

where $\tau = \frac{\lambda_1}{\gamma}$ and $[U_D, S_D, V_D] = svd\left(\breve{D} - \frac{\Pi_4}{\gamma}\right)$, where $svd(\cdot)$ is the singular value decomposition (SVD) operator.

The update of C and $\breve{D}$ are constructed by least square problems and their solutions are

$$C_{+} = (\breve{D}^T\breve{D} + I)^{-1}\left(\breve{D}^T M + \frac{1}{\gamma}\breve{D}^T\Pi_2 + J + \frac{\Pi_3}{\gamma}\right), \tag{5.32}$$

and

$$\breve{D}_{+} = \left(\gamma(M + MC^T + D) - \Pi_1 + \Pi_2 C^T + \Pi_4\right)\Gamma^{-1}, \tag{5.33}$$

respectively, where $\Gamma = \lambda_2 I + 2\gamma I + \gamma CC^T$.

Lastly, the update of J is constructed as

$$J_{+} = \widehat{J} - \mathrm{diag}(\widehat{J}), \tag{5.34}$$

where J is computed as follows:

$$\widehat{J} = \mathcal{S}(C - \frac{\Pi_3}{\gamma}, \frac{\lambda_3}{\gamma}). \tag{5.35}$$

Algorithm 10 ClustEN by ALM for optimizing (5.27)

1: **Input:** $Y \in \mathbb{R}^{m\times n}$, γ, ρ, and λ_1, λ_2, and λ_3
2: **while** not converged **do**
3:     **while** not converged **do**
4:         Update $M, D, C, \breve{D}, J$, respectively
5:     **end while**
6:     Update the Lagrange multipliers Π_1, Π_2, Π_3, and Π_4
7:     $\gamma = \min(\rho\gamma, \gamma_{max})$
8: **end while**
9: **Output:** $D \in \mathbb{R}^{m\times n}$ and $C \in \mathbb{R}^{n\times n}$

We also have the same update strategies of the Lagrange multipliers, Π_1, Π_2, Π_3, and Π_4, as described in Section 5.2.

In conclusion, for the problem described in (5.27), we have derived a new algorithm, named *ClustEN*, and it is described in Algorithm 10. In this algorithm, we set the initial values of optimization variables to zero. We solve for the problem (5.27) with respect to the five optimization variables using the alternating Lagrangian framework whose convergence properties are similar to those in [43]. While it is difficult to prove the convergence in general, there exist some guarantees for ensuring the convergence with mild technical conditions when we optimize three or more variables [43]. We set the stopping criterion of the algorithm to the following:

$$\begin{aligned} \|M^{(t)} - M^{(t-1)}\|_\infty < \varepsilon \ \wedge \ \|C^{(t)} - C^{(t-1)}\|_\infty < \varepsilon, \\ \wedge \ \|J^{(t)} - J^{(t-1)}\|_\infty < \varepsilon, \end{aligned} \tag{5.36}$$

where t is the number of iterations in the inner loop and ε is a small positive number. Since it is enough to obtain a nearly stationary point of the optimization

Algorithm 11 Subspace segmentation by ClustEN

1: **Input:** $Y \in \mathbb{R}^{m \times n}$, the number of subspaces k

2: Obtain C in (5.27) using Algorithm 10

3: Construct Z by performing post-processing [16] on C

4: Perform NCut on Z and segment data samples into k clusters

5: **Output:** cluster memberships of data samples

variables like FactEN, we set the stopping condition of the proposed method as $\varepsilon = 10^{-7}$ in all subspace clustering experiments.

The computational complexity of the inner loop is $O(\min(m,n)\max(m,n)^2)$ for ClustEN, which is the same as SSC [16], LRSC [107], and LRR [4].[8] Although the proposed algorithm have more optimization variables than other methods, the difference of running time among them are not significant (see section 5.4.2).

After finding the structure of multiple subspaces in the subspace representation matrix, the next stage is to perform post-processing, which is used for most of the subspace clustering algorithms and gives a definite effect on the clustering performance. In our experiments, we use a post-processing technique described in [16], which reduces the noise effect in a subspace representation matrix while preserving the sparsity. Finally, we use the well-known spectral clustering algorithm, Normalized Cuts (NCut) [52], to segment data samples to their respective subspaces. The whole procedure of the subspace segmentation based on the proposed method is summarized in Algorithm 11.

[8]Note that we compare the accelerated version of LRR described in [4] in this work.

5.4 Experiments

We evaluated the performance of the proposed subspace learning method, FactEN, by experimenting with various synthetic and real-world problems, such as non-rigid motion estimation [46, 49], photometric stereo [14, 22], and background modeling [12]. We compared FactEN to the state-of-the-art low-rank approximation methods, ALADM [11], Regl_1-ALM [49], and Unifying [22], and rank estimation methods, IALM [43] and ROSL [12]. We set the parameters of FactEN as follows: $\rho = 1.2$ for all cases, except for Giraffe and Static Face datasets, in which $\rho = 1.05$; and $\beta_0 = 0.5$ for all cases, except for non-rigid motion estimation problems, in which $\beta_0 = 10^{-2}$. Note that $\beta = \beta_0 / \|Y\|_\infty$.

We also compared the another proposed method, ClustEN, with the state-of-the-art subspace segmentation algorithms, SSC [16], LRR [4], LRSC [107], LSR [111], and SMR [59], for well-known subspace clustering problems, such as motion segmentation [55], face clustering [4], and handwritten digits clustering [59]. For ClustEN, we focus on the comparison of methods for clustering accuracy and running time. We set the parameters of ClustEN as follows: $\rho = 1.2$ for face clustering, 1.7 for handwritten digit clustering, and 1.5 for motion segmentation; and $\gamma = 10^{-2}$ for face and handwritten digits clustering, 10^{-1} for motion segmentation, respectively. We report the setting of remaining parameters, $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \lambda_3)$, for each dataset in Section 5.4.1. Parameters of all compared algorithms are set as reported in their papers and tuned to achieve the best performance for each task. In this work, we used an inexact version of ALM [43] in the proposed algorithms for all experiments, since the inexact version generally gives the comparable performance with faster computation than exact ALM [43, 4].

5.4.1 Subspace learning problems

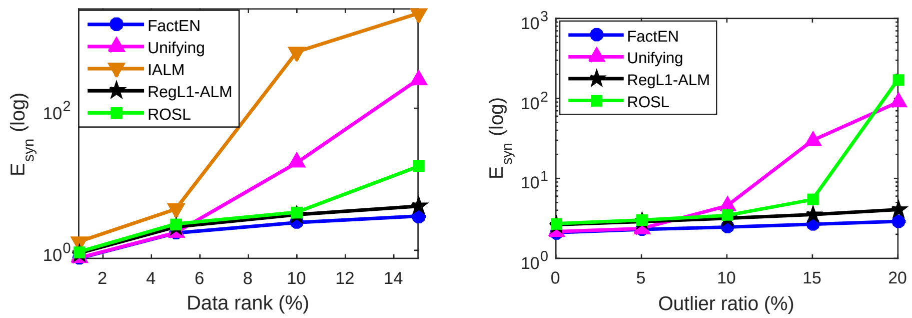
Synthetic data

First, we applied the proposed method to synthetic examples. We generated six test sets from 500×500 to $10,000 \times 10,000$ with Gaussian noises which were sampled from $\mathcal{N}(0, 10^{-2})$. In the matrices, we added outliers for randomly selected entries, which were uniformly distributed in the range of $[-15, 15]$. All entries of the weight matrix W are one in this problem. We set the rank of each test data matrix as $r = \lceil \min(m, n) \times 0.01 \times \kappa \rceil$. In the experiment, the average reconstruction error E_{Syn} is calculated as $E_{Syn} = \frac{1}{n}||M^{gt} - \widehat{M}||_1$, where M^{gt} is the ground truth and $\widehat{M}$ is the low-rank matrix approximated by the applied algorithm.

Figure 5.3 shows average performances on a synthetic example (500×500) with various data ranks[9] and various outliers ratios to verify the robustness under various conditions. We did not perform IALM for experiments using different outlier ratios, since it gives much poorer performance than compared methods. Overall, the proposed method outperforms other methods with respect to the reconstruction error for both scenarios. Regl_1-ALM follows the proposed method with slight error difference. Unifying gives similar performance to FactEN, but its reconstruction error becomes higher as the data rank or outlier ratio increases. IALM and ROSL show unsatisfactory results when data rank or outlier ratio is large, restricting their applications in practice. From Figure 5.3(b), we can see that the proposed method is robust to outliers regardless of the outlier ratio.

To verify the ability of the proposed method compared to Unifying with respect to the rank and sparsity, we conducted an experiment for a 1,000×1,000 synthetic example. Figure 5.4 plots the fraction of correct recoveries at different

[9] Note that the data rank means the percentage of the true rank over the maximum possible rank of the data matrix.



(a) (b)

Figure 5.3: Average performances on a synthetic example (500×500) with various conditions. (a) Average reconstruction errors at different observation data rank ratios (10% outliers). (b) Average reconstruction errors at different outlier ratios (10% data rank).

rank and sparsity ratios. The region which is correctly recovered by the proposed method appears to be broader than that of Unifying. From the figure, the proposed method is more capable of handling corruptions than Unifying.

Figure 5.5(a) and 5.5(b) show average reconstruction errors and execution times of different algorithms, respectively, for various matrix sizes with 8% fixed data rank and 4% outliers which were uniformly distributed in the range of $[-20, 20]$. We could not evaluate IALM and Regl_1-ALM for a large-scale problem ($10,000 \times 10,000$) because of their heavy computational complexity. The proposed method outperforms the other methods with respect to the reconstruction error in all cases. Although Regl_1-ALM shows the similar performance compared with the proposed method for small-scale datasets, it takes a longer computation time to get a good solution and shows poor performance for large-scale problems. The computing time of ALADM is faster than FactEN, but it performs poorer than

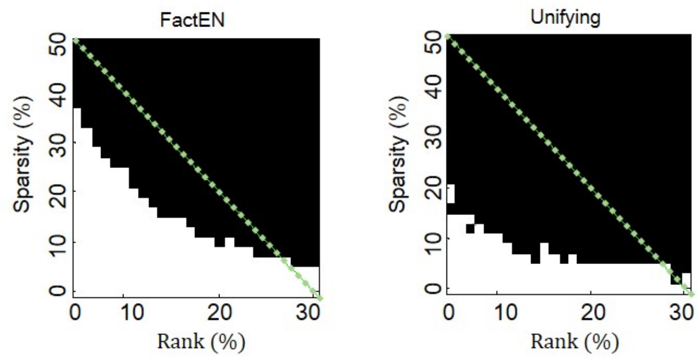


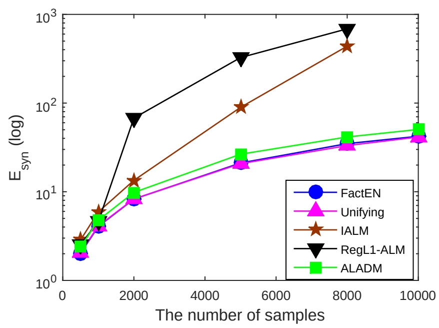
Figure 5.4: Phase transition in rank and sparsity for a synthetic example (1,000×1,000) using the proposed method and Unifying. Correct recovery (white region) is achieved when a recovered low-rank matrix $\widehat{M}$ satisfies $\|M^{gt} - \widehat{M}\|_1/\|M^{gt}\|_1 \leq 5 \times 10^{-4}$.

FactEN.

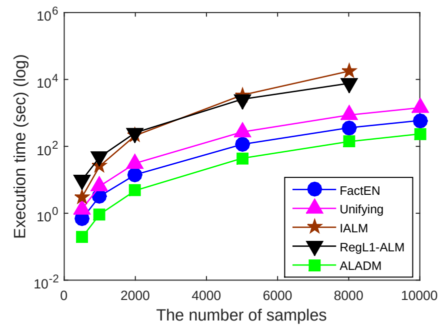
To compare the proposed algorithm in realistic conditions, we changed the outliers to block corruptions with missing entries in a synthetic example. For a similarly constructed 300×300 example, we added occlusions with various sizes with 20% missing data. Figure 5.5(c) shows reconstruction errors of different methods. As shown in the figure, the proposed method robustly reconstructs corruptions while other methods except ALADM give poor reconstruction results when there are large-sized block corruptions. It is interesting to note that Unifying is not robust against heavy corruptions including missing data compared to the proposed method.

Non-rigid motion estimation

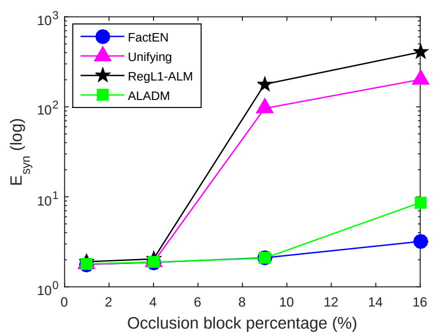
We evaluated the proposed method for real-world problems, which are summarized in Table 5.2. For these problems, we computed the mean absolute error



(a)



(b)



(c)

Figure 5.5: Average performances for synthetic problems in the presence of corruptions. (a) Average reconstruction errors with random outliers for various data sizes. (b) Average execution times for various data sizes. (c) Average reconstruction errors with various block corruption sizes and 20% missing for an example of 300×300 in size.

Table 5.2: Summary of real-world problems with known rank r.

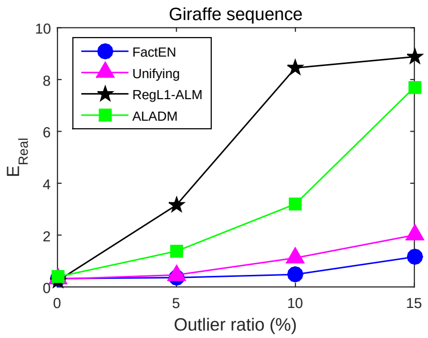
Datasets	Size	Rank r	Missing
Giraffe [74]	240 × 167	6	30 %
Shark [13]	91 × 240	6	10 %
Static Face [74]	4,096 × 20	4	42 %
PETS 2009 [89]	110,592 × 221	2	0 %

(MAE) over the observed entries as

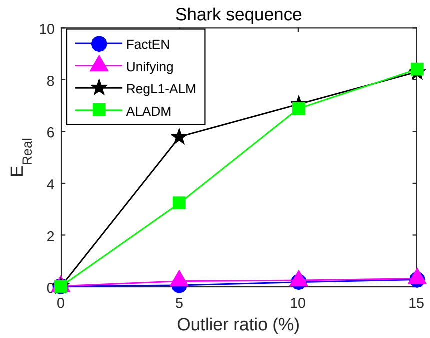
$$E_{Real} = \frac{||W \odot (M^{gt} - \widehat{M})||_1}{\|W\|_1}. \tag{5.37}$$

First, we conducted a non-rigid motion estimation experiment using Giraffe sequence [74]. The non-rigid motion estimation in the presence of missing data from image sequences can be considered as a low-rank approximation problem. In this problem, low-rank matrix factorization can be applied to restore 2D tracks contaminated by outliers and missing data. To demonstrate the robustness of the proposed method, we replaced 5% of the randomly selected points in a frame by outliers in the range of $[0, 100]$ whereas the data points are in the range of $[127, 523]$. In this setting, we performed several experiments by changing outlier ratio in the data.

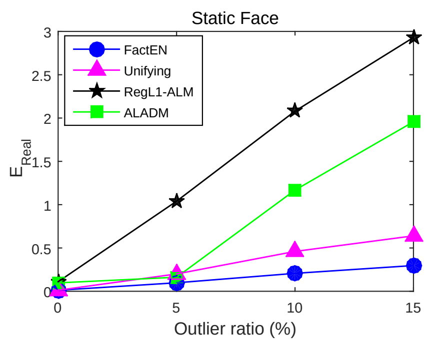
The result for the Giraffe sequence in the presence of various outlier levels is shown in Figure 5.6(a). The figure also includes the case when no outliers are added. As shown in the figure, FactEN gives the best performance regardless of the outlier ratio. Although Unifying gives similar reconstruction performance when the outlier ratio is small, the performance gets worse as the outlier ratio increases. Regl_1-ALM and ALADM show worse performance compared to other state-of-the-art methods. Figure 5.7 shows how the average reconstruction error



(a)



(b)



(c)

Figure 5.6: Average performances on real-world problems (non-rigid motion estimation, photometric stereo) in the presence of outliers and missing data. (a) Giraffe sequence. (b) Shark sequence. (c) Static face.

is affected by the choice of λ_1 for FactEN and Unifying [22]. The proposed method shows more stable results under different values of λ_1 and λ_2, whereas Unifying is sensitive to the choice of λ_1.

We also performed the motion estimation problem using the Shark sequence [13]. In this data, we examine how robust the proposed method is for various outlier ratios in the presence of missing data. We randomly dropped 10% of points in each frame as missing data. We set from 0% to 15% of tracked points

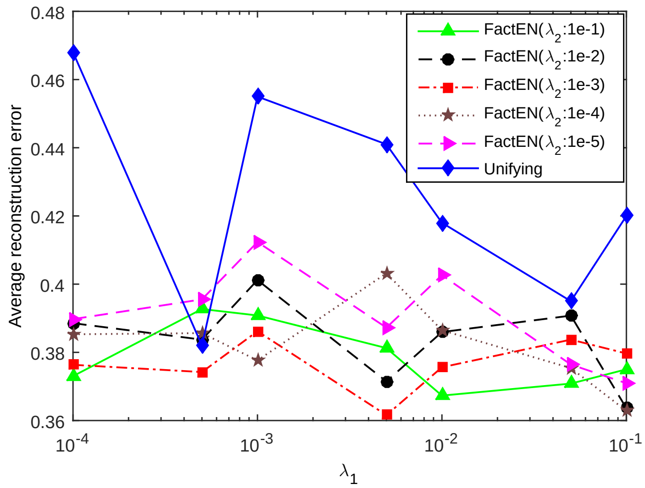


Figure 5.7: Comparison between the proposed method and Unifying [22] at different values of λ_1 for the Giraffe sequence. $(\cdot)$ denotes a value of λ_2.

as outliers in each frame in the range of $[-1000, 1000]$, whereas the data points were located in the range of $[-105, 105]$.

Average reconstruction errors at various outlier ratios by different methods are shown in Figure 5.6(b). As shown in the figure, FactEN and Unifying both give outstanding reconstruction results. However, the proposed method gives the better reconstruction results than Unifying on average. Similar to the previous example, Regl_1-ALM and ALADM show the bad reconstruction performances when there exist outliers. The reconstruction results of the three selected algorithms, the proposed method, Unifying, and l_1-ARG$_D$, for selected three frames in the presence of 15% outliers are shown in Figure 5.8. From the figure, we can observe excellent reconstruction results by the proposed method against missing data and outliers compared to the other approaches. Even though Unifying shows the similar reconstruction, it sometimes fails to estimate the exact reconstruction

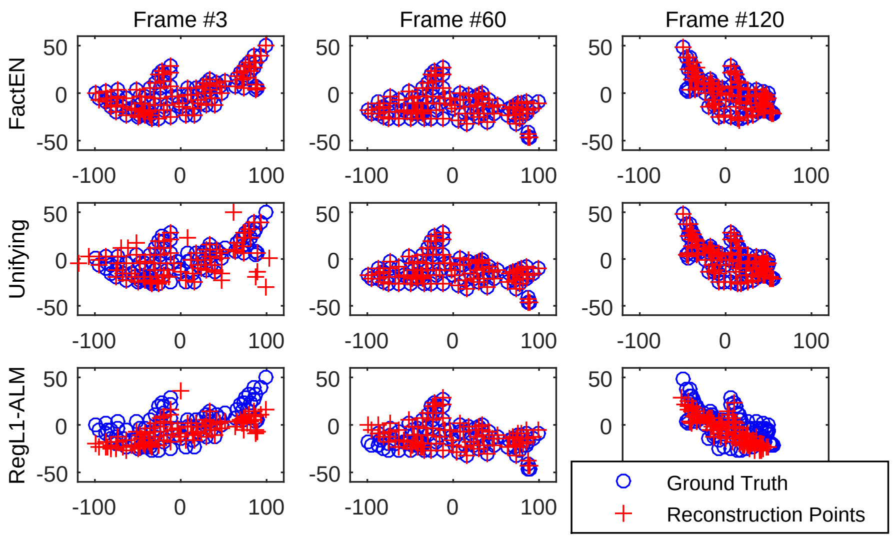


Figure 5.8: Reconstruction results from the shark sequence by three methods: FactEN, Unifying [22], and RegL1-ALM [49]. $'\bigcirc'$ means the ground truth and $'+'$ means the reconstruction point.

point as shown in the figure.

Photometric stereo

Photometric stereo [14] is another well-studied problem to estimate the surface normal of an object given multiple images of the object under different lighting conditions. It can be shown that the observation matrix has rank at most 3 [14]. In this work, we used the Static Face sequence [22] for the problem which has 20 images consisting of 64×64 pixel per image. We examine how robust the proposed method is for various outlier ratios in the presence of missing data. We set from 0% to 15% of tracked points as outliers in each frame in the range of $[0, 100]$.

The overall results are represented in Figure 5.6(c). From the figure, the proposed method gives the obvious distinction compared to other methods regardless of the outlier ratio. Following the proposed method, Unifying presents the second best performance. Although ALADM shows the satisfactory performance when there exist small elements corrupted by outliers or no outliers, the reconstruction error gets larger as the outlier ratio increases. Regl_1-ALM gives the vulnerability for outliers in this problem.

Background subtraction

Modeling background from a video sequence is an important step to separate foreground objects from background and applied to many applications, including video surveillance, traffic monitoring, and abnormal behavior detection [92]. A background modeling task can be considered as a low-rank matrix approximation problem [35]. We have used a benchmark video dataset, PETS2009 [89], which exists many walking people from a static overhead camera. For the task, we used PETS2009 [89] which is a sequence of 221 frames. Since the original image frame size is 576 $\times$ 768, which is very high dimensional, we rescaled each frame to 288 $\times$ 384 for computational tractability and thus the stacked data size is is 110,592 $\times$ 221. We performed the proposed method compared with the state-of-the-art methods: Unifying [22] and ROSL [12]. We added 30% random noises in randomly selected frames.

Figure 5.9 shows the background modeling results on two selected frames. As shown in the figure, FactEN and Unifying correctly separated foreground from background. The rank estimation method, ROSL, fails to find a good solution in the presence of heavy corruptions. The computation times are 186.37 sec for the proposed method, 497.46 sec for Unifying, and 145.93 sec for ROSL. Although ROSL gives the slightly faster computation time than FactEN, it did not provide

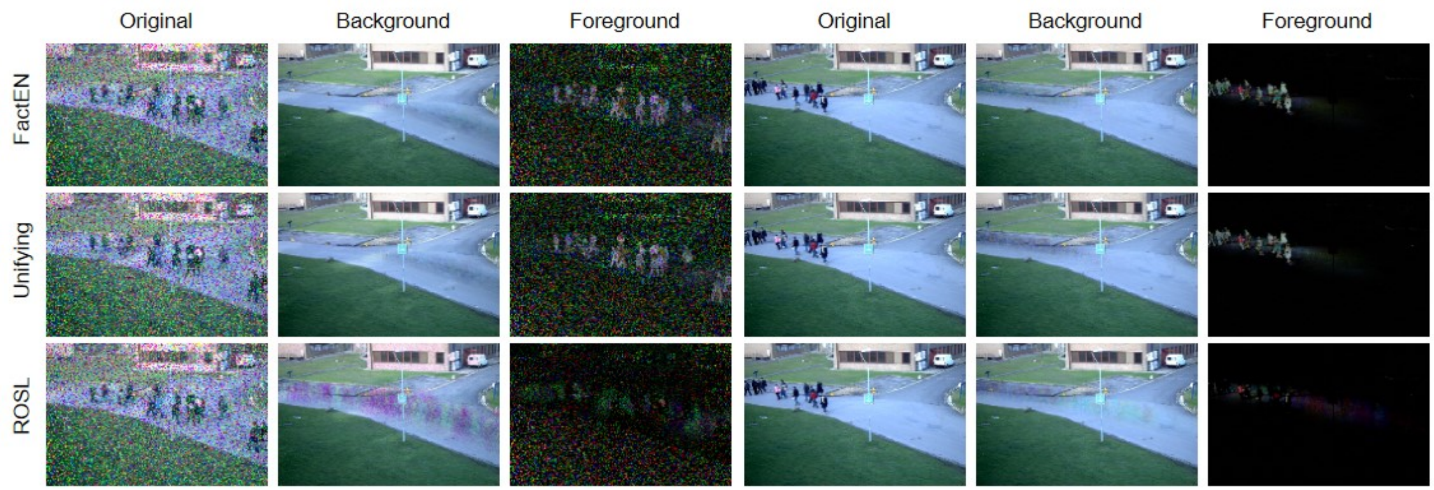


Figure 5.9: Background modeling results of the methods for two selected frames in the PETS2009 dataset. The first frame is corrupted by noise in all channels, whereas the second frame is not contaminated by noise. Each algorithm decomposes the original image into background and foreground images.

satisfying results.

In order to compare the algorithms quantitatively, we used the Bootstrapping sequence [90]. The dataset has a foreground ground-truth image which is used to compare the performance of algorithms in terms of precision and recall.[10] We used the whole 300 frames, where each frame is 160×120, and converted them into gray-scale images. In the dataset, we inserted uniform noises from [0,1] for randomly selected 10% of entries. We extracted final foreground images of different algorithms by performing pixel-wise thresholding with mathematical morphology (closing). Two low-rank approximation algorithms, Regl_1-ALM and ALADM, were included in this experiment. Figure 5.10 represents the precision-recall curve by varying the threshold level for final foreground images. From the figure, the proposed method shows the higher performance compared to other algorithms. While Regl_1-ALM gives higher performance than FactEN when the recall is low, it performs poorer than FactEN as we require higher recalls. In this problem, Unifying gives the worst performance among the tested methods. The running times of the compared methods are 11.9 sec for FactEN, 24.5 sec for Unifying, 11.7 sec for ROSL, 211.4 sec for Regl_1-ALM, and 3.1 sec for ALADM.

5.4.2 Subspace clustering problems

The proposed subspace clustering method, ClustEN, is compared in this section. We evaluate the method along with other state-of-the-art algorithms for three subspace clustering problems using the clustering accuracy and execution time. The clustering accuracy is computed as $\frac{1}{n}\sum_{i=1}^{n}\varphi(p_i, map(q_i))$, where n is the

[10] The precision and recall are computed as follows: Precision = *TP/(TP+FP)* and Recall = *TP/(TP+FN)*, where *TP* is the number of correctly estimated foreground pixels, *FP* is the number of wrongly estimated background pixels, and *FN* is the number of wrongly estimated foreground pixels.

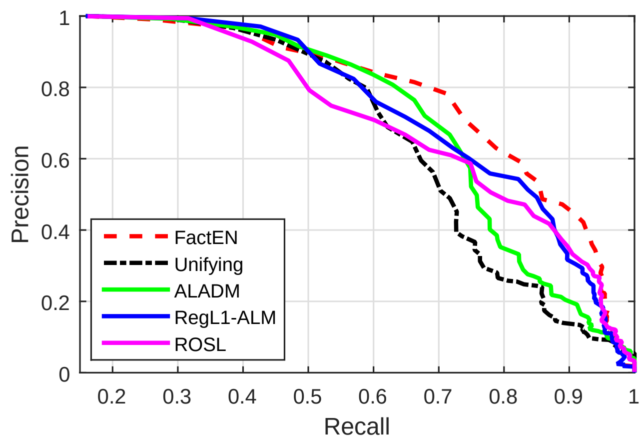


Figure 5.10: Precision-recall curve for the Bootstrapping sequence [90].

number of samples, p_i and q_i are the ground-truth and estimated cluster labels from the tested method, respectively, $\varphi(a, b)$ is the Kronecker delta function, and $map(\cdot)$ is a mapping function to permute estimated labels to match with the ground-truth labels, which is computed by the Kuhn-Munkres algorithm [77].

Motion segmentation

Motion segmentation [55] is the process of separating tracked points of moving objects from a video sequence into their underlying independent subspaces. Since trajectories associated with a rigid motion lie in a low-dimensional subspace, we regard motion segmentation as a subspace clustering problem. We performed the proposed subspace clustering method compared with the state-of-the-art algorithms, SSC [16], LRR [4], LRSC [107], LSR [111], and SMR [59], for the well-known benchmark dataset, Hopkins 155 [55]. Hopkins 155 dataset contains 155 video sequences along with features of two or three motions in all frames. Typical examples of the Hopkins 155 dataset are described in Figure 5.11. Motivated from

Figure 5.11: Typical examples in the Hopkins 155 dataset.

the work in [16], we computed four measures for the accuracies of 155 sequences: mean, standard deviation (Std), minimum, and median values. The parameters of the proposed algorithm are set to $\boldsymbol{\lambda} = (10^{-2}, 10^{-1}, 5 \times 10^{-3})$.

The experimental results of different methods are shown in Table 5.3. From the table, the proposed method gives the state-of-the-art performance. Although SMR shows better clustering accuracy than ClustEN, their performance gap is insignificant. It is interesting to note that the proposed method is based on the joint optimization using sparse representation similar to SSC [16], hence, SSC can be considered as a baseline method of ours. In this respect, the proposed method outperforms SSC with respect to all measures. Hence, we can see that the subspace learning part in the proposed joint learning procedure can improve clustering performance. LRSC, which has the similar strategy as ours, gives worse performance than ours.

Face clustering

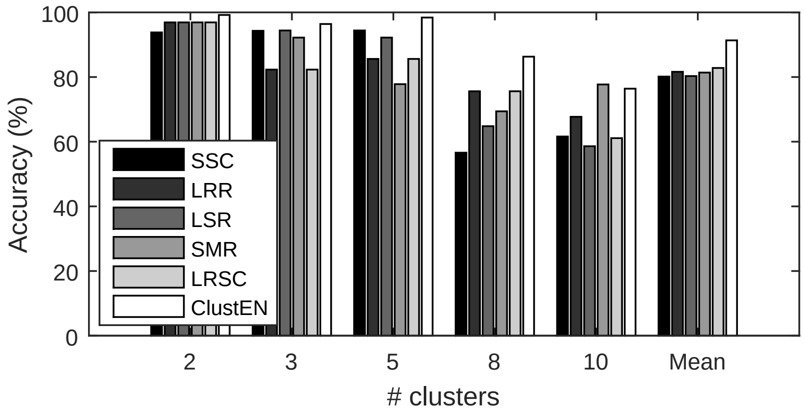
Face clustering [78] is a task to segment face images collected from multiple subjects into their corresponding identities under various illumination conditions. To evaluate the performance of the proposed method, we use the Extended Yale B dataset [78], which contains 38 subjects each of which has 64 aligned frontal

Table 5.3: Motion segmentation results (%) on the Hopkins 155 dataset.

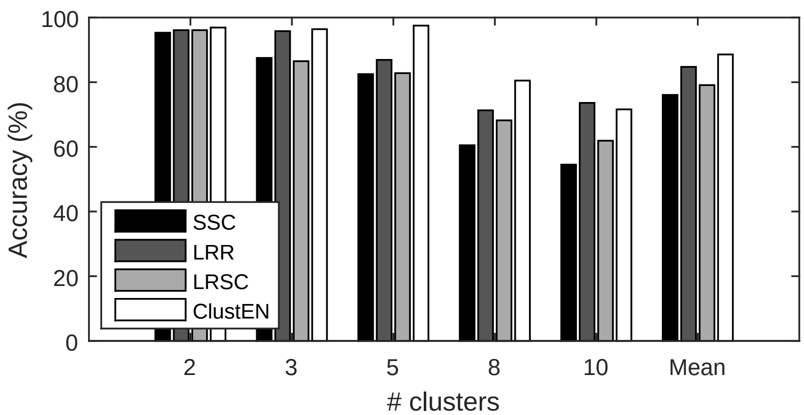
Algorithms	Mean	Std	Min	Median
SSC	96.2	9.34	52.2	**100**
LRR	96.9	7.73	59.9	99.7
LRSC	96.5	8.08	**60.3**	99.5
LSR	95.9	10.2	52.1	99.6
SMR	**97.7**	**6.7**	58.2	**100**
ClustEN	97.4	7.19	57.6	**100**

face images under various illumination conditions. Following the works [16], we evaluated different methods for five scenarios by collecting the first c subjects, where $c \in \{2, 3, 5, 8, 10\}$. We created a dataset by reducing the dimension of each image to $9c$ by PCA. Hence, we have a dataset, whose size is $9c \times 64c$, for each scenario. We set the parameters of ClustEN to $\boldsymbol{\lambda} = (10^2, 50, 5 \times 10^{-2})$.

The clustering accuracies of different methods are shown in Figure 5.12. From Figure 5.12(a), the proposed algorithm outperforms existing methods on average. Even though SSC performs better than existing algorithms except ClustEN, it degrades when the number of clusters is large. SMR shows good performance for $c = 10$, but it gives unsatisfactory results on average compared to the proposed method. LRR and LRSC show the similar clustering accuracies across the scenarios. We also compared the proposed method with respect to the running time. The running times of different methods for a scenario when the number of subjects is 10, are 5.68 sec for SSC, 1.45 sec for LRR, 0.73 sec for LRSC, 0.16 sec for LSR, 1.2 sec for SMR, and 7.13 sec for ClustEN, respectively. The proposed method gives the competitive computing time compared to other methods, even though it has many variables to learn in a joint optimization problem.



(a)



(b)

Figure 5.12: Clustering accuracy (%) on the (a) Extended Yale B dataset and (b) Yale-Caltech dataset.

Figure 5.13: Examples from the Yale-Caltech dataset. First and second rows show facial and non-facial (outlier) images, respectively.

To evaluate the robustness of the proposed algorithm, we created a dataset motivated from the work in [4]. The dataset, which we call Yale-Caltech, consists of the Extended Yale B dataset [78] and Caltech 101 dataset [112]. We collected 101 images from Caltech 101 dataset, where we randomly selected an image for each class, and regarded them as outlying samples. The typical examples from the Yale-Caltech dataset are shown in Figure 5.13. As described in the previous experiment, we selected the first c subjects from the Extended Yale B dataset. We made the dataset by blending Extended Yale B and Caltech data sets, each of which has dimension of $9c$ by projecting it to a basis matrix extracted from the Extended Yale B dataset using PCA. We compared our proposal with existing methods, SSC [16], LRR [4], and LRSC [107], which address outliers. In the dataset, we did not compare LSR and SMR since they cannot handle outliers. We set the parameters of the proposed method to $\boldsymbol{\lambda} = (10^2, 50, 8 \times 10^{-2})$.

Figure 5.12(b) shows the clustering accuracy of the compared methods for the Yale-Caltech dataset. Similar to the previous problem, the proposed method gives the best performance outperforming existing algorithms. Whereas, LRSC, which is another joint optimization method, performs poorer than the proposed algorithm. Even if LRR can handle outlying samples due to its group sparsity regularizer, it dose not show satisfying results compared to the proposed algorithm. Average accuracies of the methods are 8.25 for SSC, 2.78 for LRR, 0.98 for LRSC, and 8.96 for ClustEN. As shown in Figure 5.12, the proposed algorithm shows its excellent performance for problems with and without corruptions.

Handwritten digits clustering

The proposed algorithm was also applied to handwritten digits clustering problems using the USPS dataset [113], which consists of 9,298 16×16 grayscale images. The number of classes is ten, which contains digits from 0 to 9. We tested the proposed algorithm compared with existing methods for two scenarios by selecting the first 500 and 1,000 samples, which contains image samples from all classes, from the dataset. The parameters of ClustEN are as follows: $\boldsymbol{\lambda} = (5 \times 10^2, 5 \times 10^2, 10^{-1})$.

Table 5.4 shows the segmentation accuracy (%) and running time (sec) of different algorithms. As shown in the table, the proposed algorithm, ClustEN, gives the state-of-the-art performance on average for both scenarios. SMR gives the comparable performance to the proposed method. Note that all algorithms, except SMR and ClustEN, show unsatisfactory results when the number of samples are large ($n = 1,000$). Another joint optimization approach, LRSC, shows poor performance for this problem. When it comes to the running time, the proposed algorithm shows the decent running time, which is faster than SSC, and LRR. Although LSR shows the fastest running time due to the closed-form solution, its clustering accuracy is lower than that of ours.

Table 5.4: Handwritten digit clustering results on the USPS dataset.

	n=500		n=1,000	
Algorithm	Acc	Time	Acc	Time
SSC	71	9.13	61.3	33.3
LRR	75.8	18.06	66	31.9
LRSC	47.8	4.03	50.3	9.49
LSR	72.2	**0.19**	66.2	**0.86**
SMR	73.4	0.94	**74.8**	8.75
ClustEN	**76.0**	3.34	73.4	12.7

5.5 Summary

Throughout this chapter, we have proposed a new subspace representation framework based on elastic-net regularization of singular values. The introduced elastic-net is shown to stabilize the proposed algorithms in the presence of heavy corruptions due to the strong convexity. The proposed algorithms can find a robust solution more efficiently and is stable against missing data and outliers. Two algorithms are developed under the proposed framework. FactEN is proposed to robustly identify a low-rank matrix approximating the given data matrix. For the general problem of subspace clustering and estimation, ClustEN is proposed. The proposed algorithms have been applied to a number of applications for subspace learning and clustering, including non-rigid motion estimation, photometric stereo, and background modeling problems for subspace learning, and motion segmentation, face clustering, and digit clustering for subspace clustering. The experimental results show that the proposed algorithms outperform the state-of-the-art methods in terms of the approximation error, clustering accuracy, and execution time.

Chapter 6

Robust Group Subspace Representations

As mentioned in the previous chapter, subspace clustering assumes that a data sample can be represented by other samples drawn from the same subspace. While many recent studies are based on sparse or low-rank representation for robustness, the grouping effect among similar samples has not been often considered with sparse or low-rank representation. In this chapter, we introduce *group subspace representation* to handle highly correlated data samples. It is motivated by the well-known regularizer introduced in Chapter 5,[1] called elastic-net [104], which has the grouping effect with variable selection. Based on the representation using the elastic-net regularization, we propose two robust subspace clustering algorithms: *group sparse representation (GSR)* and *group low-rank representation (GLR)* which are based on sparse and low-rank representation, respectively. GSR

[1]While the elastic-net is introduced in Chapter 5 for the purpose of stabilizing the proposed algorithm by regularizing singular values, we use the regularizer from a grouping perspective for subspace segmentation by regularizing coefficient elements.

is devised to reveal grouping effect in sparse representation due to the strictly convexity of the proposed representation. While LRR has the grouping effect as discussed earlier, GLR is proposed to overcome the non-strict convexity of LRR and to demonstrate the effectiveness of the proposed group subspace representation over existing methods.

The main contributions of the proposed methods are summarized as follows. First, the proposed group subspace representation generalizes sparse and low-rank representation problems with strictly convexity promoting the subspace grouping effect. It accelerates the grouping capability for both representations by capturing the similarity among data samples collected from the same cluster, even in the presence of noises or corruptions. We also show that our two proposals, GSR and GLR, reveal a block-diagonal structure if subspaces are independent. In addition, we verify the grouping capability of our proposals when highly correlated data are presented, theoretically and empirically. Lastly, the proposed methods outperform the state-of-the-art methods, without introducing an additional computational complexity from their baseline methods, on well-known benchmark subspace clustering tasks, such as motion segmentation and face clustering with and without corruptions.

6.1 Group Subspace Representation

The well-known subspace clustering approaches, SSC [53, 16] and LRR [54, 4], work well for many problems, but they have limitations when performing a clustering task as discussed in the previous section. In order to overcome the weaknesses, we introduce a generalized approach, named *group subspace representation*, to improve both methods. Motivated by the grouping effect discussed in [104], we define:

Definition 3 (Group subspace representation)**.** *Given a set of sample vectors $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$, where samples are drawn from k subspaces. The task of group subspace representation is to find a subspace representation matrix $Z = [\boldsymbol{z}_1, ..., \boldsymbol{z}_n] \in \mathbb{R}^{n\times n}$, where $\|\boldsymbol{z}_i - \boldsymbol{z}_j\| \to 0$ if $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \to 0, \forall i \neq j$, to segment the samples according to the underlying subspaces they are drawn from.*

From the definition, we can consider the following problem to find a subspace representation matrix Z:

$$\min_{Z} \|Z\|_s + \frac{\lambda}{2}\|Z\|_F^2, \quad s.t.\ X = XZ,\ \mathrm{diag}(Z) = 0, \tag{6.1}$$

such that $\lambda > 0$. Here, $\|Z\|_s$ can be the l_1 norm for finding sparse representation of Z or the nuclear norm for finding low-rank Z (with the last constraint, $\mathrm{diag}(Z) = 0$, removed). This formulation promotes sparsity by the l_1 norm or the nuclear norm and enforces grouping effects on a subspace representation matrix Z from the Frobenius norm regularizer over Z, which allows grouping of highly correlated samples in X. This is due to the strict convexity property of the group subspace representation in (6.1), unlike the sparse representation in (2.8) in Chapter 2, which is non-strict convex. Furthermore, it shrinks the subspace representation matrix to parsimonious one by the l_1 norm of Z. The distinct difference between strict and non-strict convexity can be seen from the following lemma [104]:

Lemma 3. *In a linear regression model, $\boldsymbol{x} = X\boldsymbol{z}$, where $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n]$ is a set of sample vectors and $\boldsymbol{z}$ is a coefficient vector, assume that $\boldsymbol{x}_i = \boldsymbol{x}_j$, for some $i, j \in \{1, ..., n\}$. (a) If we use the group subspace representation in (6.1), then $z_i = z_j$. (b) If we use the sparse representation in (2.8), then $z_i z_j \geq 0$ and z^* is* *another minimizer, where*

$$z_k^* = \begin{cases} z_k & \textit{if} \;\; k \neq i \;\; \textit{and} \;\; k \neq j, \\ (z_i + z_j) \cdot (s) & \textit{if} \;\; k = i, \\ (z_i + z_j) \cdot (1 - s) & \textit{if} \;\; k = j, \end{cases}$$

for any $s \in [0, 1]$.

The strict convexity guarantees the grouping effect in the ideal situation with the same samples drawn from a cluster, whereas the sparse representation approach does not provide a unique solution because of its non-strict convexity. Although Lemma 3 shows an ideal case where samples are exactly the same, we can infer the weakness of the sparse representation in (2.8) from Lemma 3. Based on the above analysis, we propose two methods: group sparse representation (GSR) and group low-rank representation (GLR), which are based on the group subspace representation defined in Definition 3. Figure 6.1 shows the clustering evaluation of the proposed methods, GSR and GLR, and their corresponding baseline algorithms, SSC and LRR, using a synthetic example with small corruptions. From the figure, the proposed methods find the subspace structure better than the baseline algorithms, which can fail to find the exact clusters when there are corruptions (see the second cluster). Our proposals accelerate the cluster grouping which prevents an unnecessary segmentation within a cluster.[2]

Theorem 3. *Suppose that the data sampling is sufficient and samples are drawn from a union of* k *independent linear subspaces. Let us define a function* f *satisfying* $f(Z) = f(ZP)$*, for any permutation matrix* P*. Then, the optimal solution* $Z^* \in \mathbb{R}^{n \times n}$ *to the problem (6.1) is block-diagonal.*

[2] While an affinity matrix is not perfect block-diagonal, an application of spectral clustering can provide a better segmentation result by cleaning up disturbances in the imperfect affinity matrix [114].

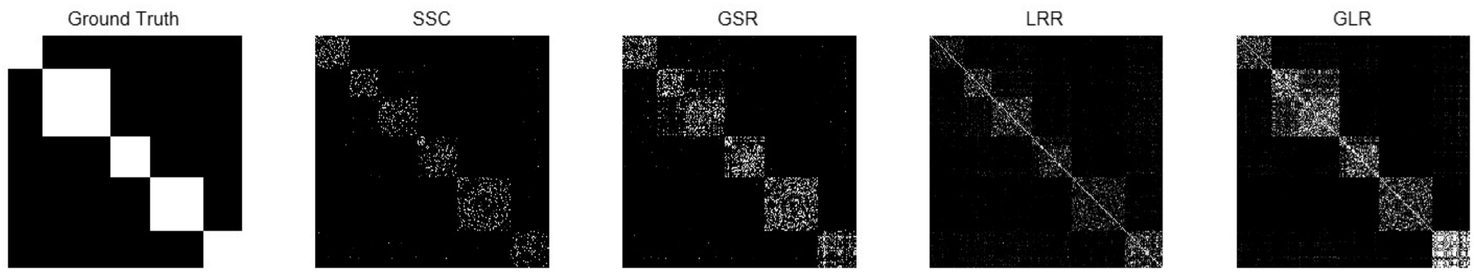


Figure 6.1: An evaluation of the proposed methods, GSR and GLR, and their baseline methods, SSC [16] and LRR [4], for a synthetic example with corruptions. Figures show a ground truth affinity matrix and affinity matrices computed from different algorithms.

Proof. See Appendix F.1 □

Theorem 3 shows that the optimal solution of a linear combination of any functions satisfying $f(Z) = f(ZP)$, for any permutation matrix P, such as the l_1 norm, Frobenius norm, and nuclear norm, achieves the block-diagonal condition. In the following subsections, we introduce two algorithms based on the group subspace representation.

6.2 Group Sparse Representation (GSR)

6.2.1 GSR with noisy data

In practice, there exist noises in real data sets. Now, we modify the cost function (6.1) to consider noises as follows:

$$\min_{Z} \ \xi_F(Z) + \lambda_1 \|Z\|_1 + \frac{\lambda_2}{2}\|Z\|_F^2, \quad s.t. \ \mathrm{diag}(Z) = 0, \tag{6.2}$$

where $\xi_F(Z)$ is the Frobenius norm loss function to reflect the Gaussian noises, i.e., $\frac{1}{2}\|X - XZ\|_F^2$, and λ_1 and λ_2 are weighting parameters. The problem (6.2), which we name group sparse representation (GSR), is a method based on the well-known elastic-net regularizer [104] with self-dictionary X. Elastic-net is a generalization of ridge and Lasso regression methods with a grouping effect by applying both the l_1 norm and Frobenius norm regularization on Z [104]. Hence, GSR can prevent the sparsest representation of Z by grouping clusters properly. When there exist closely related samples drawn from the same cluster, GSR encourages the subspace representation matrix Z to have the same membership for the closely related samples, as stated by the following theorem [104]:

Theorem 4. *Given a sample $\boldsymbol{x}_k \in \mathbb{R}^d$, a dataset $X \in \mathbb{R}^{d \times n}$, and parameters (λ_1, λ_2), and assume that X is normalized. Let $\boldsymbol{z}^* \in \mathbb{R}^n$ be the optimal solution* *to following problem:*

$$\min_{\boldsymbol{z}} \frac{1}{2}\|\boldsymbol{x}_k - X\boldsymbol{z}\|_2^2 + \lambda_1\|\boldsymbol{z}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{z}\|^2, \tag{6.3}$$

where $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_{k-1}, \boldsymbol{x}_{k+1}, ..., \boldsymbol{x}_{n+1}]$. *Supposed that* $z_i z_j > 0$*, we have the following relation:*

$$\mu(z_i^*, z_j^*) \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)}, \tag{6.4}$$

where $\mu(z_i^*, z_j^*) = \|z_i^* - z_j^*\|_2/\|\boldsymbol{x}_k\|_2$ *and* $\rho = \boldsymbol{x}_i^T\boldsymbol{x}_j$ *is the sample correlation.*

Proof. See Appendix F.2. □

Theorem 4 says that when $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are highly correlated up to a sign change when negatively correlated, i.e., $\rho \simeq 1$ ($\rho \simeq -1$ if negatively correlated, then consider $-\boldsymbol{x}_j$), the difference between the corresponding coefficients in $\boldsymbol{z}$ is almost 0, leading to the same subspace membership.

6.2.2 GSR with corrupted data

Now, we consider a problem where collected data are faced with unwanted corruptions, such as outliers and occlusion blocks. Since the problem (6.2) with the Frobenius norm cannot handle the corruptions, a robust loss function, such as the l_1 norm, is a better choice to deal with corruptions

$$\min_{Z,E} \ \|Z\|_1 + \frac{\lambda_1}{2}\|Z\|_F^2 + \lambda_2\xi_1(Z), \ s.t. \ \mathrm{diag}(Z) = 0, \tag{6.5}$$

where $\xi_1(Z) = \|X - XZ\|_1$ is the element-wise l_1 norm of $X - XZ$.

Optimization for solving (6.5)

The problem (6.5) can be solved by the alternating minimization approach under the augmented Lagrangian framework. Let E be a corruption matrix, which is modeled by $X - XZ$. Then we have the following Lagrangian:

$$\begin{aligned}\mathcal{L}(Z,C,E) = &\|Z\|_1 + \frac{\lambda_1}{2}\|Z\|_F^2 + \lambda_2\|E\|_1 \\ &+ \mathrm{tr}(\Pi_1^T(X - XC - E)) + \mathrm{tr}(\Pi_2^T(C - Z)) \\ &+ \frac{\beta}{2}(\|X - XC - E\|_F^2 + \|C - Z\|_F^2),\end{aligned} \tag{6.6}$$

such that $\mathrm{diag}(Z) = 0$, where C is an auxiliary variable for Z, and Π_1 and Π_2 are Lagrange multipliers and β is a penalty parameter. We have optimization problems to update the variables Z, C, and E using the alternating direction method of multipliers (ADMM) [115]. First, we solve Z by the following equation

$$Z = \widehat{Z} - \mathrm{diag}(\widehat{Z}), \tag{6.7}$$

where $\widehat{Z}$ is obtained by solving the following problem

$$\widehat{Z} = \min_Z \|Z\|_1 + \frac{\lambda_1}{2}\|Z\|_F^2 + \frac{\beta}{2}\left\|C - Z + \frac{\Pi_2}{\beta}\right\|_F^2, \tag{6.8}$$

and the solution of (6.8) can be computed by the absolute value shrinkage operator [43]:

$$\widehat{Z} = \mathcal{S}_{\frac{1}{\lambda_1+\beta}}\left(\frac{1}{\lambda_1+\beta}(\beta C + \Pi_2)\right), \tag{6.9}$$

where $\mathcal{S}_\nu(x) = \mathrm{sgn}(x)\max(|x| - \nu, 0)$ for a variable x.

For solving C and E, we have the following problems:

$$\min_C \left\|X - XC - E + \frac{\Pi_1}{\beta}\right\|_F^2 + \left\|C - Z + \frac{\Pi_2}{\beta}\right\|_F^2, \tag{6.10}$$

$$\min_E \lambda_2\|E\|_1 + \frac{\beta}{2}\left\|X - XC - E + \frac{\Pi_1}{\beta}\right\|_F^2, \tag{6.11}$$

where (6.10) is a least-square problem whose solution is

$$C = \Delta^{-1}\left(X^TX - X^TE + Z + \frac{X^T\Pi_1}{\beta} - \frac{\Pi_2}{\beta}\right) \tag{6.12}$$

Algorithm 12 GSR or GLR for subspace clustering

1: **Input:** data matrix $X \in \mathbb{R}^{d \times n}$ lying in a union of k linear subspaces

2: Solve an optimization problem of GSR or GLR to obtain a subspace representation matrix Z

3: Form a similarity graph $\breve{Z}$ from Z

4: Apply a clustering method to $\breve{Z}$ in order to segment the data samples to k clusters

5: **Output:** a similarity graph $\breve{Z}$ and k clusters

with $X^T X + I = \Delta$ and (6.11) is computed in a closed form using the absolute value shrinkage operator:

$$E = \mathcal{S}_{\frac{\lambda_2}{\beta}} \left(X - XC + \frac{\Pi_1}{\beta} \right). \tag{6.13}$$

Note that we have the same optimization strategy to that of [16], which solves Z and E simultaneously using ADMM, whose convergence to the optimal solution for two variables are guaranteed in [43]. In summary, we derive a group sparse representation (GSR) algorithm, based on the group subspace representation discussed in Section 6.1, for robust subspace segmentation, which is described in Algorithm 12. In the algorithm, we solve an optimization problem (6.2) or (6.5) according to the case when there are noises or outliers, respectively. After finding a subspace representation matrix Z from the optimization, we construct an undirected similarity graph $\breve{Z}$ as stated in [53]. Finally, we assign a cluster label for each sample based on a clustering algorithm.

6.3 Group Low-Rank Representation (GLR)

6.3.1 GLR with noisy or corrupted data

The proposed group subspace representation (6.1) can be applied to the nuclear norm based subspace clustering problems, such as LRR [54] and LatLRR [116]. Like the sparse representation [53], the nuclear norm based clustering algorithms can sometimes encourage the within-cluster segmentation due to their non-strict convexity when there exist corruptions as shown in Figure 6.1. Hence, our group subspace representation can help the nuclear norm based clustering methods to improve the subspace grouping effect in a within-cluster. The new formulation to consider the grouping capability is as follows (noiseless case):

$$\min_{Z} \|Z\|_* + \frac{\lambda}{2}\|Z\|_F^2 \quad s.t.\ X = XZ. \tag{6.14}$$

The problem (6.14), which we call group low-rank representation (GLR), also satisfies the block-diagonal condition in Theorem 3. Based on results given in [59], we can show the grouping effect of GLR as follows:

Theorem 5. *The optimal solution of GLR has grouping effect, i.e., given a set of data samples $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$ and a subspace representation matrix $Z \in \mathbb{R}^{n\times n}$, a solution to the optimization problem of GLR using X, if $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \to 0$, then $\|\boldsymbol{z}_i - \boldsymbol{z}_j\| \to 0$ for all $i \neq j$.*

Proof. See Appendix F.3. □

When data samples contain noises, i.e., $X = XZ + E$, where elements in E have the independent and identically distributed Gaussian distribution, we can easily make an optimization problem by inserting a loss function $\xi_F(Z)$ to the formulation instead of the equality constraint.

Now, we consider a more realistic scenario when data have some corruptions. As stated in Section 6.2, we introduce a loss function $\xi_1(Z)$ to the problem (6.14) to have the following robust subspace clustering problem:

$$\min_{Z} \xi_1(Z) + \lambda_1 \|Z\|_* + \frac{\lambda_2}{2} \|Z\|_F^2, \tag{6.15}$$

where λ_1 and λ_2 are weighting parameters.

Optimization for solving (6.15)

The problem (6.15) can be solved by ADMM of the following problem with two auxiliary variables:

$$\begin{aligned} \min_{Z,D,M} & \|X - D\|_1 + \lambda_1 \|Z\|_* + \frac{\lambda_2}{2} \|Z\|_F^2, \\ s.t. & \quad D = XM, \ Z = M, \end{aligned} \tag{6.16}$$

and its corresponding augmented Lagrangian is

$$\begin{aligned} \mathcal{L}(Z, D, M) = & \|X - D\|_1 + \lambda_1 \|Z\|_* + \frac{\lambda_2}{2} \|Z\|_F^2 \\ & + \operatorname{tr}(\Pi_1^T (D - XM)) + \operatorname{tr}(\Pi_2^T (Z - M)) \\ & + \frac{\beta}{2} \left(\|D - XM\|_F^2 + \|Z - M\|_F^2 \right), \end{aligned} \tag{6.17}$$

where $\Pi_1 \in \mathbb{R}^{d \times n}$ and $\Pi_2 \in \mathbb{R}^{n \times n}$ are Lagrange multipliers and $\beta > 0$ is a small penalty parameter. Then, we solve for each variable while other variables held fixed.

First, we form the following optimization problem to solve for Z:

$$\min_{Z} \lambda_1 \|Z\|_* + \frac{\lambda_2}{2} \|Z\|_F^2 + \frac{\beta}{2} \left\| Z - M + \frac{\Pi_2}{\beta} \right\|_F^2, \tag{6.18}$$

and the solution of (6.18) can be computed by the singular value shrinkage operation [43] in a closed form as follows:

$$Z = U_1 \mathcal{S}_\tau (S_1) V_1^T, \tag{6.19}$$

where $\tau = \frac{\lambda_1}{\lambda_2+\beta}$ and

$$[U_1, S_1, V_1] = svd\left(\frac{1}{\lambda_2+\beta}(\beta M - \Pi_2)\right), \tag{6.20}$$

where svd is the singular value decomposition operator.

For finding D, we consider the following problem:

$$\min_{D} \|X - D\|_1 + \frac{\beta}{2}\left\|D - XM + \frac{\Pi_1}{\beta}\right\|_F^2, \tag{6.21}$$

and it has a closed-form solution using the absolute value shrinkage operator:

$$D = X - \mathcal{S}_{\frac{1}{\beta}}\left(X - XM + \frac{\Pi_1}{\beta}\right). \tag{6.22}$$

Lastly, the update of M is computed by the simple least squares

$$M = \Gamma^{-1}(\beta X^T D + X^T \Pi_1 + \beta Z + \Pi_2), \tag{6.23}$$

where $\Gamma = \beta(X^T X + I)$.

The overall procedure is to update the optimization variables via the alternating minimization until convergence. After finding the output Z by solving the problem (6.16), we build an undirected graph and apply a clustering algorithm to obtain k clusters as stated in Algorithm 12. In the ADMM procedure, we set β to an increasing sequence to a maximum point, i.e., $\beta_{t+1} = \min(\rho\beta_t, \beta_{max})$, following [43, 4]. It is interesting to note that the convergence behavior of GLR can be ensured by [43], since we have two-step optimization procedure in every iteration where Z and D are optimized independently when M is held fixed. Hence, the convergence of ADMM with two blocks can be guaranteed by [43]. A similar proof can be applied to the well-known previous work, LRR [4], where two variables in LRR can be optimized independently and simultaneously while another variable is fixed, even though they said that it is difficult to ensure the convergence of ADMM with three or more blocks [4].

6.4 Experimental Results

In this section, we evaluate the proposed methods, GSR and GLR, for various subspace segmentation tasks such as synthetic problems, motion segmentation [55, 16], and face clustering [78, 4]. In the experiments, we formulate the proposed methods based on the l_1 norm loss function, i.e., we solve GSR and GLR for problems (6.5) and (6.15), respectively. We compare the proposed methods with state-of-the-art subspace clustering methods: SSC [53, 16], LRR[3] [54, 4], LRSC [107], LatLRR [116], LSR [111], CASS [56], and SMR [59]. For the comparing algorithms, we use the codes released by their authors. The parameters for each method are tuned to have the best performance for each task. In experiments, clustering accuracy and running time are used to evaluate the performance of methods, where the clustering accuracy are calculated using the metric from [59]. Since k-means can give an unsatisfactory result when the number of clusters is large as it can be biased to the initial condition [117, 118], we take another approach described in [118], which avoids such problem, to segment data samples into k clusters for all methods. Nonetheless, we also provide results using spectral clustering (with k-means) in Table 6.2.

Synthetic Examples

First, we performed clustering experiments on synthetic examples. We generated an example where the number of clusters and the number of samples were chosen randomly in the range of [3, 10] and [30, 70], respectively. The dimension of each sample is set to 50. For each cluster, we drew samples from a linear subspace which was generated by obtaining orthogonal basis vectors from Gaussian

[3]We used an accelerate version of LRR [4], which gives a speed-up over the original LRR [54].

Table 6.1: Average performance on synthetic problems over 100 independent runs. From the first to the third row, we have the names of algorithms, clustering accuracies (%), and running times (sec), respectively.

Algorithms	SSC	LRR	LSR	CASS	SMR	GSR	GLR
Accuracy	83.31	86.55	84.78	85.68	86.60	89.05	**91.61**
Time	1.43	0.52	**0.044**	247.7	0.21	0.46	2.08

random vectors whose mean is zero and standard deviation is chosen randomly. The number of basis vectors is randomly selected to be less than the half of the number of samples. When generating a synthetic dataset, which consists of k clusters, we added a noise matrix whose elements had the Gaussian distribution with zero-mean and variance of 0.2. In this problem, we compared with SSC and LRR, and methods addressing the grouping issue (LSR, CASS, and SMR) to demonstrate the performance of the proposed group subspace representation.

The average clustering accuracy and running time over 100 different synthetic examples are shown in Table 6.1. From the table, the proposed methods achieve the best clustering accuracy with a competitive running time. GSR outperforms SSC on both clustering performance and computing time. Although LSR is faster than ours, the clustering performance is lower than those of GSR and GLR. Since CASS gives a much longer running time than other methods (over 100 times longer) because of its expensive operation to solve the trace Lasso based optimization problem, it is hard to be used for large-scale problems in practice. SMR gives the best performance among the existing methods, but it gives lower clustering accuracy than the proposed methods.

Figure 6.2 shows affinity matrices computed from different algorithms for an

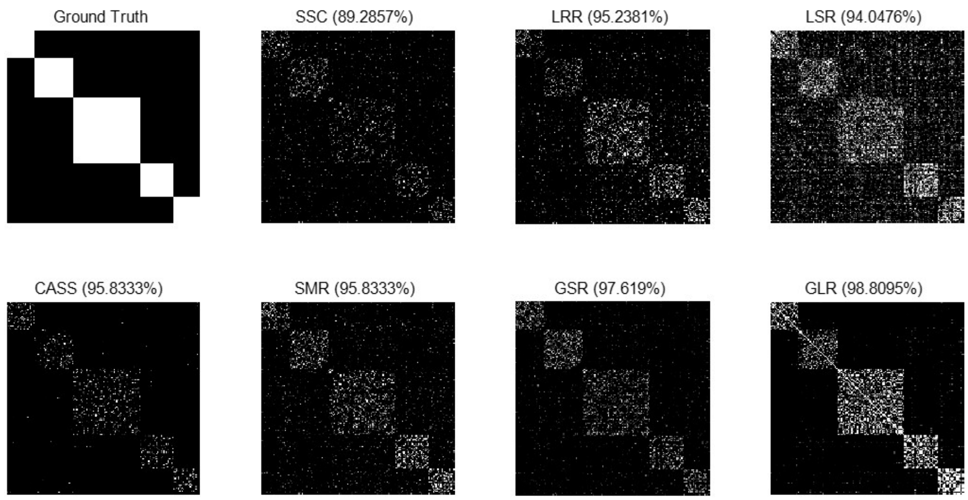


Figure 6.2: Clustering evaluation of the proposed methods and other state-of-the-art methods, SSC, LRR, LSR, CASS, and SMR, for a synthetic example with Gaussian noises. Figures show a ground truth affinity matrix and affinity matrices computed from different algorithms. $(\cdot)$ denotes the clustering accuracy.

example where the number of clusters is 5 and the number of samples in each subspace was chosen randomly in the range of [30, 70] with Gaussian noises. As shown in the figure, the proposed two methods show the clear representations over the affinity matrix with higher clustering accuracies than other methods, whereas other methods represent somewhat noisy affinity matrices with poorer performance than ours.

We also generated an example with the same setting to the previous example, and added a corrupted matrix which consists of a square occlusion block whose area is $n^2/10$ and Gaussian noises. The optimization results from different algorithms are shown in Figure 6.3. In the figure, most of the affinity matrices give the noisy representation due to the corruption. Among the methods, our pro-

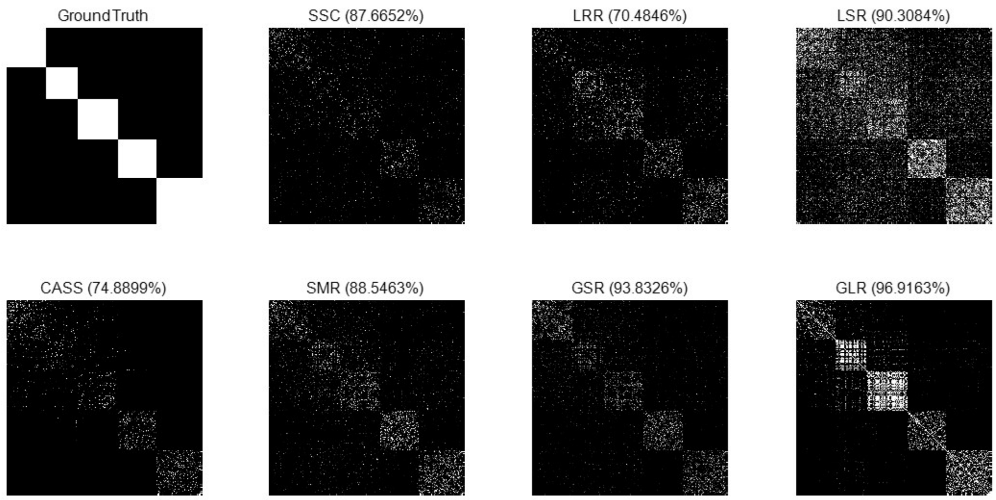


Figure 6.3: Clustering evaluation of the proposed methods and other state-of-the-art methods, SSC, LRR, LSR, CASS, and SMR, for a synthetic example with corruptions. Figures show a ground truth affinity matrix and affinity matrices computed from different algorithms. $(\cdot)$ denotes the clustering accuracy.

posals represent more clean results than the compared methods including their baseline algorithms, SSC and LRR. Specifically, GLR improves LRR by preventing the inter-cluster grouping and outperforms other methods significantly by its robust group subspace representation. CASS gives poor performance because it did not capture the resemblance among similar samples in some subspaces under the noisy scenario.

To verify the robustness of the proposed methods under the various of noise conditions, we added various percentages of corrupted elements, from 0% to 100%, to synthetic examples whose elements are drawn from a uniform distribution in the range of $[-1, 1]$. In this experiment, we compared our proposals with the corresponding baseline methods, SSC and LRR, to investigate the robustness of the

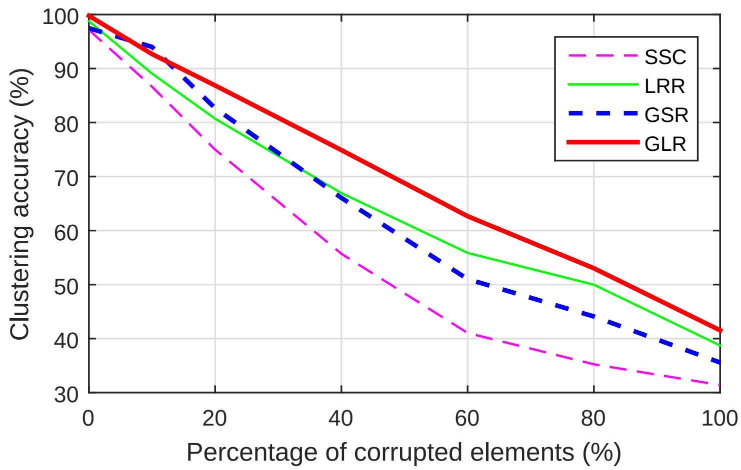


Figure 6.4: Average clustering performance on synthetic examples under various noise ratios.

proposed subspace grouping. The average clustering performances of the methods over 100 independent scenarios are shown in Figure 6.4. Note that even though SSC gives much lower accuracy than LRR, GSR reduces the gap considerably and even surpasses LRR when the corruption ratio is lower than about 35%. As shown in the figure, we can see that the proposed grouping methods outperform their baseline algorithms.

Motion Segmentation

Motion segmentation [55] is a task for clustering trajectories of rigidly moving objects based on tracked points along the frames. Since all trajectories associated with a single rigid motion lie in a low-dimensional subspace, it is considered as a subspace clustering task over the point trajectories. We applied the proposed methods to the well-known benchmark dataset, Hopkins 155 database[4].

[4] `http://www.vision.jhu.edu/data/hopkins155`

We compared the proposed methods with seven state-of-the-art subspace clustering methods for evaluation, including LRSC [107] and LatLRR [116]. Table 6.2 describes the results of two measures (mean, standard deviation (Std)) over segmentation accuracy of the methods for the Hopkins 155 dataset. We compared the methods via two clustering methods, a method in [118] and spectral clustering [58], as discussed in the previous section. From the table, GSR outperforms other methods including SSC with respect to the mean and standard deviation of 155 motion segmentation tasks. GLR shows competitive results for both cases and it also has a higher accuracy than LRR, which is the baseline algorithm of GLR. LatLRR and SMR show better performance than GLR, but not as good as GSR. LSR and CASS, which address the grouping issue, do not give satisfactory results compared to the proposed methods. We can see that the proposed group subspace representation method helps the baseline methods, SSC and LRR, significantly. Note that GSR can have a denser membership representation than SSC because of its subspace grouping, which can balance between sparse and dense representation. Although LRR gives a dense representation by minimizing a nuclear norm based optimization problem, our group representation using GLR further enhances the clustering accuracy. Note that the two clustering methods [118, 58], which are used after affinity matrices are found, give similar clustering performance for most of the methods in this problem.

Face Clustering

Face clustering without outliers. Face clustering [78, 54] is a task for segmenting face images into their identities. We tested the proposed algorithms for the face clustering task under unfavorable conditions. We used the Extended Yale B [78], which consists of 38 subjects placed in order where each subject has about

Table 6.2: Motion segmentation results (%) on the Hopkins 155 dataset.

	Method in [118]		Spectral clustering	
Algorithms	Mean	Std	Mean	Std
SSC	96.42	8.99	96.46	9.11
LRR	96.59	7.67	96.53	8.04
LRSC	96.43	7.85	96.5	7.94
LatLRR	97.51	6.19	97.53	6.12
LSR	95.86	10.45	95.62	10.89
CASS	94.67	9.89	94.35	10.55
SMR	97.25	7.44	97.25	7.44
GSR	**98.4**	6.42	**98.37**	6.58
GLR	96.64	7.45	96.73	7.66

60 manually aligned frontal face images under illumination variations. Following the experimental setting in [119, 120], we make 8 scenarios by taking the first c subjects from the dataset, where $c \in \{2, 3, 5, 8, 10, 20, 30, 38\}$ is the number of subjects, to verify the clustering performance for various subjects. Similar to the setup in [119], face images were projected into $9 \times c$-dimensional subspace by PCA [9].

Table 6.3 shows the clustering accuracy with respect to the number of clusters. The proposed methods, GSR and GLR, outperform other methods on average. Especially, GSR gives much higher accuracy than others when the number of clusters is larger than three. GLR gives the second best performance on average and it outperforms other methods when $c = 2$. Following our proposals, LatLRR and CASS perform better than others but their performance are unsatisfactory. LRR and LRSC give similar clustering accuracy and lower than that of LatLRR on average. LSR and SMR show the poor performance when the number of clusters is over 30. Although CASS gives the satisfactory results for small subject cases, its performance gets worse when the number of subjects increases. SSC shows the worst performance on average and especially it gives unsatisfactory results when the number of clusters is large. From the table, we can see that the proposed methods based on the group subspace representation work well for all cases and show the superiority over the face clustering experiment without outliers, even though their formulations are based on the l_1 norm loss function.

Face clustering with outliers. To verify the robustness of the proposed methods, we created a dataset, Yale-Caltech, which combines Extended Yale B and Caltech-101[5] [112], motivated by [4]. Unlike the dataset described in [4], we randomly collected an image from each category of Caltech-101 as outliers. Hence, we added 101 outlier images, which are converted into gray-scale images, to a dataset consisting of the first c subjects. We resized both face and outlier images to 20×20 to make all images have the same size and to reduce the computational cost and memory requirement. We performed face clustering experiments for $c \in \{10, 20, 30, 38\}$ to investigate the clustering performance of the proposed methods when the number of clusters is large.

Figure 6.5 shows the clustering performance of methods, SSC, LRR, LRSC, LatLRR, GSR, and GSR, which can handle outliers. Even though LSR, CASS, and SMR are not robust against non-Gaussian noises, we provide the results of them in the following experiment. In this experiment, the clustering accuracy is computed only for the facial images without the outlier images. The proposed method, GSR, achieves the highest accuracy for all cases. SSC gives the competitive results compared to the proposed methods. Although GLR gives less

[5] `http://www.vision.caltech.edu/feifeili/Datasets.htm`

Table 6.3: Face clustering results (%) on the Extended Yale B dataset.

Algorithms		SSC	LRR	LRSC	LatLRR	LSR	CASS	SMR	GSR	GLR
No. subjects	2	97.7	96.9	96.9	96.9	95.3	96.1	96.9	97.7	**98.4**
	3	95.8	85.4	85.9	91.1	86.9	94.3	92.2	**98.4**	96.9
	5	71.6	88.4	88.8	90.6	89.7	93.8	79.4	**97.8**	91.9
	8	71.7	84.6	84.6	87.3	84.8	76.6	83.4	**96.7**	91.2
	10	75.8	75.9	76.1	77.3	74.7	68.6	85.2	**96.9**	82.7
	20	69.8	75.3	75.5	76.4	75.1	74.6	71.3	**86.1**	80.5
	30	61.0	72.8	73.1	69.7	69.6	79.8	67.6	**81.4**	67.9
	38	50.8	63.3	64.3	63.4	54.7	63.8	60.0	**69.2**	66.5
Average		74.2	80.3	80.6	81.6	78.9	80.9	79.5	**90.5**	84.5

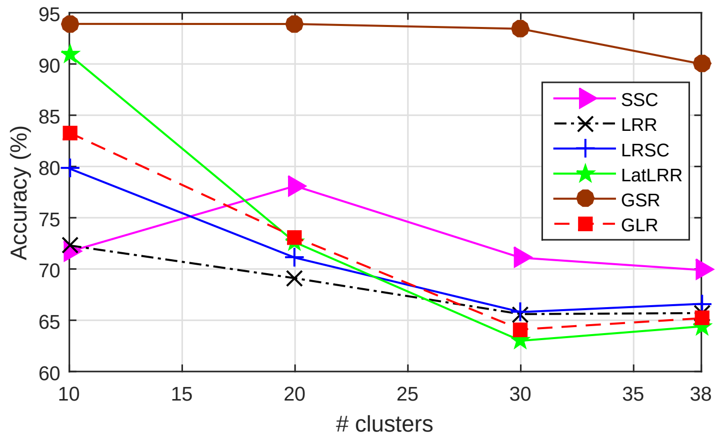


Figure 6.5: Face clustering results on the Yale-Caltech dataset.

accuracy than SSC and LatLRR, it outperforms LRR and LRSC on average. Note that like the previous example, it is meaningful to compare our proposals, GSR and GLR, with their baseline methods, SSC and LRR. In this perspective, the proposed methods show a significant improvement. The running times of the methods are 173.4 sec for SSC, 80.3 for LRR, 98.4 for LRSC, 108.2 for LatLRR, 182.7 for GSR, and 490.4 for GLR, for the case of $c = 38$.

For the Yale-Caltech dataset, we provide the experimental result for the case when $c = 10$ for all compared methods including LSR, CASS, and SMR, as mentioned before. Table 6.4 shows the clustering performance and running time of different methods. Similar to the previous example, the proposed methods give the best performance among the methods with competitive running time. Three methods, LSR, CASS, and SMR, which cannot handle outliers, show poor performance in this case. In addition, CASS shows an extremely long computation time, making it infeasible for large-scale problems.

Table 6.4: Face clustering accuracies (%) and running times (sec) on the Yale-Caltech dataset. (# clusters: 10)

Algorithms	SSC	LRR	LRSC	LatLRR	LSR	CASS	SMR	GSR	GLR
Accuracy	71.7	72.3	79.8	90.9	70.8	63.7	40.2	**93.9**	83.3
Time	16.8	25.8	7.87	43.8	**0.51**	15,680.3	2.55	18.3	12.3

6.5 Summary

In this chapter, we have proposed two subspace clustering algorithms, group sparse representation (GSR) and group low-rank representation (GLR), using the group subspace representation. The proposed methods simultaneously address sparsity-based representation and the grouping issue by introducing a strong convex regularizer, since a grouping capability is important for improving the subspace clustering performance. Our proposals encourage the grouping effect by capturing the resemblance among data samples drawn from the same subspace. The proposed methods have been applied to various subspace clustering tasks, such as synthetic problems, motion segmentation, and face clustering under the existence of various noise and illumination conditions. Experimental results show that our methods provide favorable performance compared to existing methods.

Chapter 7

Scalable Low-Rank Subspace Clustering

In this chapter, we address another important issue of the subspace clustering task. While existing subspace clustering algorithms have been successfully applied to various clustering problems, they are still challenges in terms of scalability and an ability to handle out-of-samples. These methods compute an affinity matrix using all observed samples in a batch mode. Hence, if an out-of-sample is introduced, the affinity matrix has to be recomputed using all samples. Hence, they are not scalable and their applications are limited. Furthermore, since most of the methods are iterative approaches or need heavy complexity when constructing an affinity matrix, they are not suitable for large-scale problems. There is an additional factor to consider. After an affinity matrix is computed, there are two remaining steps, post-processing and spectral clustering, whose time complexities are also significantly high (in general, over cubic complexity).

To reduce the complexity everywhere in subspace clustering, in this chapter, we propose an end-to-end[1] integrated pipeline for scalable subspace clustering. We first introduce a scalable learning framework for subspace clustering which seeks to find an affinity matrix incrementally without degrading the performance from its baseline algorithm. The complexity of the introduced incremental learning framework is further reduced by proposing *summary representation* based on the motivation that a subspace can be well represented by sparse representative basis vectors [121]. But there still remains post-processing[2] and spectral clustering steps before the final clustering result if obtained. These additional steps can sometimes demand more computation than the affinity learning step. To reduce the complexity of the overall algorithm, we propose an efficient integration of post-processing and spectral clustering into the proposed scalable low-rank representation framework, named *scalable low-rank representation (SLR)*. It is interesting to note that even our method is based on the l_2-norm, the proposed summary representation enforces the affinity matrix to be low-rank and has sparse connections due to its selection strategy. To conclude, the proposed learning framework achieves not only the competitive performance but also robustness to outliers, as well as the fairly reduced time complexity. The main contributions of the proposed method are as follows.

- The proposed method constructs an affinity matrix incrementally using the summary representation, which gives an efficient and robust representation of data with low complexity.

[1] We would like to note that the term "end-to-end" is used in this chapter to describe the fully scalable framework in the entire process from the front-end to the back-end, even though the meaning of recently used end-to-end pipelines in the deep learning literature is slightly different from our intention.

[2] Since it has an impact on the clustering performance, many algorithms usually contain a post-processing step.

- More importantly, the proposed affinity learning strategy is integrated in a complete pipeline of subspace clustering, including post-processing and spectral clustering, to reduce the overall time complexity to linear in the number of samples.

- The proposed method can be integrated with kernel methods for handling challenging problems where data lie in nonlinear manifolds. Thus, the proposed framework can address both linear and nonlinear clustering problems.

- The clustering accuracy of the proposed method is satisfactory with an order-of-magnitude speed-up compared to the existing subspace clustering algorithms on various benchmark tasks.

7.1 Incremental Affinity Representation

The goal of this work is to develop an efficient scalable algorithm for subspace segmentation since many recently developed methods are not suitable for handling streaming samples. To handle this issue, we develop a scalable method based on least squares regression (LSR) [111]. LSR utilizes an l_2-norm regularizer for enforcing grouping effects among the samples of the same subspace, and it shows the state-of-the-art performance on various datasets. The l_2-norm regularizer in LSR makes it highly efficient and adequate for incremental processing, but at the same time, it can make the method vulnerable to outliers or ill-conditioned subspaces. This disadvantage will be addressed by using the *robust summary representation* later in this chapter. Before introducing the proposed method, we present an incremental approach of LSR in this section, since the incremental concept is used in the proposal in the next section. First, we reformulate the LSR problem under the noisy case without the diagonal constraint as follows [111]:

$$\min_{C} \|X - XC\|_F^2 + \lambda \|C\|_F^2, \tag{7.1}$$

where λ is a weighting parameter and its analytical solution is

$$C^* = (X^T X + \lambda I)^{-1} X^T X. \tag{7.2}$$

Here, I is the identity matrix. Although the solution consists of simple operations, it is hard to process streaming data, because it involves an inverse operation whose complexity is cubic in the number of samples. To compute the inverse operation efficiently, we introduce an equivalent solution using the matrix inversion lemma [88]:

$$\begin{aligned} \begin{bmatrix} A & U \\ V & D \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -A^{-1}U \\ I \end{bmatrix} \\ \times (D - VA^{-1}U)^{-1} \begin{bmatrix} -VA^{-1} & I \end{bmatrix}, \end{aligned} \tag{7.3}$$

where A and D are invertible and square matrices, and U and V are compatible matrices so that dimensions of A and UDV are the same.

Now, let $X_{n-1} = [\boldsymbol{x}_1, ..., \boldsymbol{x}_{n-1}] \in \mathbb{R}^{d \times (n-1)}$ be a matrix whose samples are collected until time $n-1$, and $\boldsymbol{x}_n \in \mathbb{R}^d$ is a newly observed sample. Then we can update the affinity matrix $C_n \in \mathbb{R}^{n \times n}$ for all n samples as follows:

$$\begin{aligned} C_n &= (X_n^T X_n + \lambda I_n)^{-1} X_n^T X_n \\ &= \begin{bmatrix} X_{n-1}^T X_{n-1} + \lambda I_{n-1} & X_{n-1}^T \boldsymbol{x}_n \\ \boldsymbol{x}_n^T X_{n-1} & \boldsymbol{x}_n^T \boldsymbol{x}_n + \lambda \end{bmatrix}^{-1} \cdot X_n^T X_n \\ &=: \begin{bmatrix} A & U \\ V & D \end{bmatrix}^{-1} \cdot \begin{bmatrix} \breve{A} & U \\ V & \breve{D} \end{bmatrix}, \end{aligned} \tag{7.4}$$

Algorithm 13 Incremental LSR (ILSR)

```
Input: streaming data X_n = [x_1, ..., x_n]
for i = 1, ..., n do
    Solve the problem (7.5) for each sample x_i
end for
Perform post-processing [4]
Apply spectral clustering [58] to C to obtain k clusters
```

where I_n denotes an $n \times n$ identity matrix, $\breve{A} = X_{n-1}^T X_{n-1} \in \mathbb{R}^{(n-1)\times(n-1)}$, and $\breve{D} = \boldsymbol{x}_n^T \boldsymbol{x}_n \in \mathbb{R}$. From (7.4), we have the complexity of $O(nd)$ for the inverse operation when computing with the new sample $\boldsymbol{x}_n$. Likewise, the last term $X_n^T X_n$ in (7.4) is constructed incrementally. Using (7.3) and (7.4), we compute the solution sequentially

$$
\begin{aligned}
C_n = &\begin{bmatrix} C_{n-1} & C_U \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -C_U \\ 1 \end{bmatrix} \\
&\times (D - VC_U)^{-1} \begin{bmatrix} -VC_{n-1} + V & -VC_U + \breve{D} \end{bmatrix},
\end{aligned} \tag{7.5}
$$

where $C_{n-1} = A^{-1}\breve{A}$ and $C_U = A^{-1}U$. We can see that the incremental learning of an affinity matrix in (7.5) is an incremental LSR (ILSR) approach, whose algorithm is summarized in Algorithm 13. By obtaining the affinity matrix with proper post-processing such as [4] to have more clear representation, we can find cluster memberships using spectral clustering [58]. The computational complexity for computing C_n in (7.5) is $O(n^2 d)$, since we do not need to re-compute C_{n-1}. Hence, the overall complexity of ILSR is $O(n^3 d)$, which is higher than the batch LSR method.

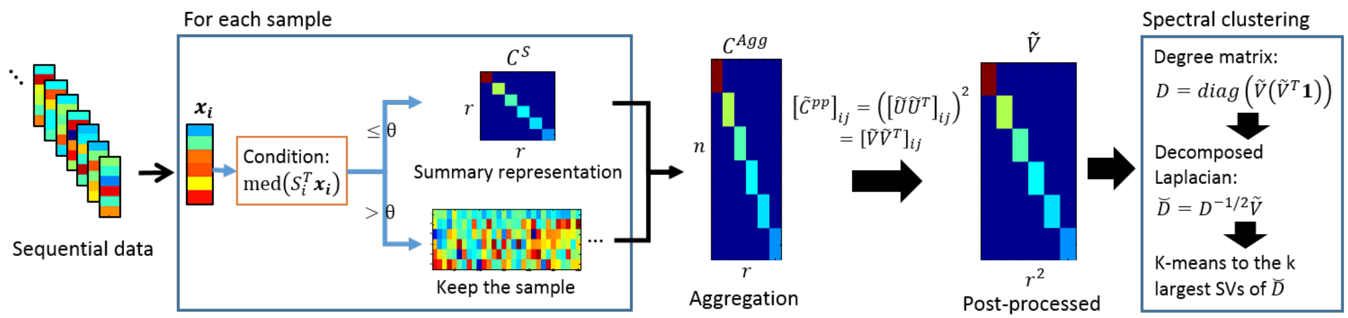


Figure 7.1: Graphical representation of the proposed end-to-end scalable subspace clustering pipeline. When i-th samples comes in, the first step is to construct a summary affinity matrix C^S incrementally by checking the condition, $\mathrm{med}(S_i^T \boldsymbol{x}_i) \leq \theta$, where S_i is a stacked matrix consisting of samples which satisfy the thresholding test until i-th time. Then, we aggregate the summary matrix with remaining samples and compute $\tilde{V}$ as a post-process matrix. Finally, we apply k-means to the k largest singular vectors (SVs) of the decomposed Laplacian matrix $\breve{D}$ to obtain cluster memberships of involved samples.

7.2 End-to-End Scalable Subspace Clustering

7.2.1 Robust incremental summary representation

To reduce the unsatisfying complexity, we propose a new approach using the concept of the widely used representative learning [121, 122]. The basic idea is derived from the fact that a subspace can be efficiently constructed based on sparse representative basis vectors, in other words, a sample in a subspace is represented by linear combination of a small number of effective basis vectors constructing the subspace. This goes along the lines of sparse representation in subspace clustering [53], which reveals a data sample by other sparse essential samples. It is interesting to note that though the proposed method is based on the l_2-norm, we can represent the features of SSC and LRR on the affinity matrix indirectly by using a low-rank approximation matrix. From this motivation, we construct a small-sized summary matrix which can represent most of samples instead of constructing an overall affinity matrix, which we named *summary representation* of the observed data.

The first step is to construct a summary affinity matrix, C^S, sequentially based on incoming samples. Assume that data samples are normalized. We can construct the summary matrix using a small subset or summary set S of data matrix X as follows:

$$C^S = \arg\min_{C} \|S - SC\|_F^2 + \lambda \|C\|_F^2, \tag{7.6}$$

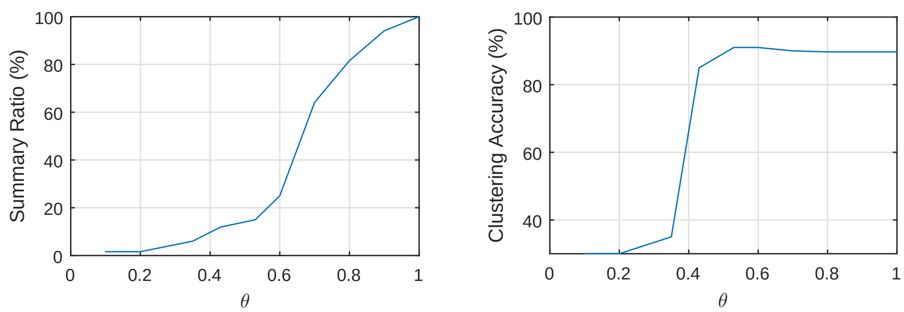
where $S = [\boldsymbol{x}_i]_{i \in \mathcal{Q}}$ is a matrix constructed by stacking $\boldsymbol{x}_i$, the i-th sample (or column) of X, for all $i \in \mathcal{Q}$. $\mathcal{Q}$ is defined as a set of indices where i is selected by examining the correlation of $\boldsymbol{x}_i$ and the previous samples indexed by the current $\mathcal{Q}$ to ensure that $\mathcal{Q}$ includes diverse samples. Note that this can be interpreted as a sparse coding [34] or a vector quantization procedure, but our selection procedure does not involve a time-consuming task such as LASSO [34]. This is motivated by [123, 63], viewing a sparse coding problem as a linear coding problem. However, this strategy can be vulnerable to outliers due to the l_2 error term. As a remedy of the issue, we add a simple but powerful stochastic outlier detection step to the procedure. The overall procedure is described below.

Let S_i be a matrix consisting of samples used for a summary matrix until time i. Then, $\boldsymbol{x}_i$ is included in S_i if it passes a thresholding test using the median of the coded vector computed from the linear coding scheme, i.e., $\mathrm{med}(S_i^T \boldsymbol{x}_i) \leq \theta$, where θ is a threshold which will affect the size of S_i and $\mathrm{med}(\cdot)$ is a median operator. This step will maximize the diversity of $\mathcal{Q}$. To eliminate the outlying samples during the step, we can further check the correlation with a small set, $\tilde{R}_i$, randomly sampled from previously unselected samples R_i. If the correlation between the current sample $\boldsymbol{x}_i$ and the sampled set $\tilde{R}_i$ is low, i.e., $\mathrm{med}(\tilde{R}_i^T \boldsymbol{x}_i) \leq \theta_0$, where θ_0 is a minimum threshold value to detect outliers, we regard $\boldsymbol{x}_i$ as an outlier. We have found that this simple strategy is highly efficient and provides excellent performance in several scenarios with outliers. (See section 7.3.1 for more details.) By varying θ, we can control the size of the summary matrix and the representation capability of the summary matrix. Hence, when $i \in \mathcal{Q}$, we update a new summary matrix C^S as follows:

$$
\begin{aligned}
C^S &= (S_i^T S_i + \lambda I_i)^{-1} S_i^T S_i \\
&= \begin{bmatrix} S_{i-1}^T S_{i-1} + \lambda I_{i-1} & S_{i-1}^T \boldsymbol{x}_i \\ \boldsymbol{x}_i^T S_{i-1} & \boldsymbol{x}_i^T \boldsymbol{x}_i + \lambda \end{bmatrix}^{-1} \cdot S_i^T S_i.
\end{aligned} \tag{7.7}
$$

Otherwise, we do not modify C^S. The remaining samples are held and later used to construct the overall affinity matrix in order to assign cluster memberships to all samples.

Note on the summary representation. To check how the summary rep-



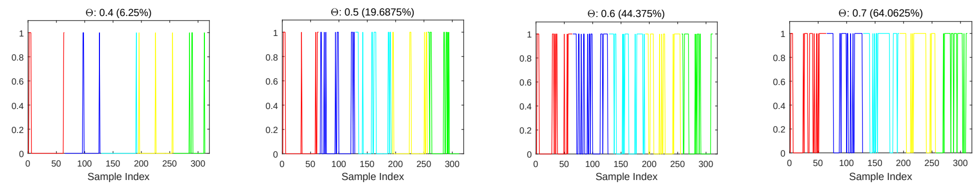
(a) (b)

Figure 7.2: Summary ratio and clustering accuracy according to the thresholding θ for face clustering. (a) Summary ratio (%). (b) Clustering accuracy (%).

resentation works, we performed the proposed method on the Extended Yale B dataset [78], where the number of clusters is 5 for a face clustering task. We varied the value θ from 0.1 to 1. Figure 7.2 shows the summary ratio and its corresponding clustering accuracy using the proposed method, which will be described in Section 7.2.3, according to θ. From the figure, we can observe that the summary ratio increases gradually when θ increases and the clustering accuracy converges to a stationary point when θ is larger than 0.5 (summary ratio is larger than 15%), which is not sensitive to the choice of θ once the accuracy reaches at a stationary point. Selected summary samples (by varying θ) are represented in Figure 7.3, which reveals that the proposed summary representation selects diverse samples in every class by its sparse selection nature.

7.2.2 Efficient affinity construction

The next step is to develop an affinity matrix based on the summary matrix C^S and the remaining set R. Let the size of the summary matrix be r. Then, we



(a) (b) (c) (d)

Figure 7.3: Graphical representation of selected summary samples (represented by 1) of the proposed method according to θ for face clustering (# cluster is 5). Each class in the dataset has 64 samples and the samples are in general position. $(\cdot)$ denotes the summary ratio for the corresponding threshold value.

form an aggregation matrix, $C^{Agg} \in \mathbb{R}^{n\times r}$, which consists of a summary matrix, $C^S \in \mathbb{R}^{r\times r}$, and a latent matrix, $C^R \in \mathbb{R}^{r\times(n-r)}$, computed using the remaining set R:

$$C^{Agg} = [C^S, C^R]^T \quad s.t.\ C^R = [\boldsymbol{c}_k]_{k\notin\mathcal{Q},\forall k}, \tag{7.8}$$

where $\boldsymbol{c}_k = (S^T S + \lambda I)^{-1} S^T \boldsymbol{x}_k$ is a latent vector with $\boldsymbol{x}_k \in R$. Now, we can obtain an overall affinity matrix as $\widetilde{C} = C^{Agg} {C^S}^{\dagger} {C^{Agg}}^T$, where $A^{\dagger}$ is the pseudo-inverse of a matrix A. Note that the subspace clustering based on the summary representation using a small number of representative samples can be guaranteed under mild conditions:

Theorem 6. *Suppose that noiseless data samples are sufficiently collected from a union of k independent linear subspaces and basis vectors constructing the summary matrix cover the remaining samples. Let us define a function f which satisfies $f(C) = f(C\mathcal{P})$ for any permutation matrix $\mathcal{P}$. Then, the problem (7.1) based on the summary representation solves the subspace clustering problem exactly with a block-diagonal structure of $\widetilde{C}$.*

Proof. See Appendix G.1. □

As mentioned earlier, however, we may consider another important step to make a final affinity matrix, i.e., post-processing to reduce noisy representations of affinity matrices. Most of the subspace clustering methods utilize a post-processing step to reduce the effect of noise before performing spectral clustering. One of popular post-processing techniques is described in [4], which acts like a singular value shrinkage [108] over a latent matrix by discarding low-impact singular values. In this post-processing step, the main computational cost is from singular value decomposition (SVD), which has $O(n^3)$ complexity and thus is not suitable for scalable learning. To reduce the complexity, instead of conducting post-processing on $\widetilde{C}$, we directly conduct SVD on the $n \times r$ rectangular matrix $C^{Rec} \triangleq C^{Agg}\widehat{U}$, where $\widehat{U} \triangleq U\Sigma^{\frac{1}{2}} \in \mathbb{R}^{r\times r}$ is computed from eigenvalue decomposition (EVD) over $C^{S^\dagger}$ such that $C^{S^\dagger} = \widehat{U}\widehat{U}^T$,[3] whose complexity is $O(nr^2)$, and follow the steps stated in [4] (please see the paper for more details). Thus, we can reconstruct an affinity matrix using outer product of $\tilde{U}$, i.e., $\widetilde{C} = \tilde{U}\tilde{U}^T$, where $\tilde{U} \in \mathbb{R}^{n\times r}$ is the post-processed matrix made from C^{Rec}. Then, we obtain the post-processed affinity matrix $\widetilde{C}^{pp}$ where

$$[\widetilde{C}^{pp}]_{ij} = [\widetilde{C} \odot \widetilde{C}]_{ij} = ([\tilde{U}\tilde{U}^T]_{ij})^2, \tag{7.9}$$

where $\odot$ is the Hadamard product. In the next section, we explore for a scalable algorithm giving an equivalent solution to (7.9) whose time complexity of the entire task is linear in the number of samples.

[3]In practice, we first compute EVD over $C^S = U\Sigma U^T$, and then perform inversion on Σ for computational efficiency.

7.2.3 An end-to-end scalable learning pipeline

Until now, we have discussed how to construct an overall affinity matrix efficiently. But, in order to obtain the cluster membership, we need to perform spectral clustering [58] after obtaining the affinity matrix with post-processing. It is important to note here that constructing an overall affinity matrix based on the thin rectangular matrix $\tilde{U}$ followed by conducting EVD to obtain a new skinny rectangular matrix in spectral clustering is quite wasteful, since handling a full affinity matrix involves heavy computational tasks. Specifically, it is important to maintain a thin matrix structure taking the effect of EVD without constructing a full affinity. to reduce the overall complexity to linear in the number of samples. To do so, we devise a unified framework by integrating the overall procedure from constructing an aggregation matrix to spectral clustering, without building an overall affinity matrix. As discussed before, we perform post-processing [4], which involves element-wise square operation in (7.9), i.e., $\widetilde{C}^{pp} = ([\tilde{U}\tilde{U}^T]_{ij})^2$, to make a clear affinity and thus enhance the clustering performance. To consider the effect of the element-wise square operation in a decomposed matrix, we present a new matrix $\tilde{V}$ using the following result:

Theorem 7. *Suppose that $\widetilde{C} = \tilde{U}\tilde{U}^T \in \mathbb{R}^{n\times n}$ with a matrix $\tilde{U} \in \mathbb{R}^{n\times r}$. Then, for a matrix $\tilde{V} \in \mathbb{R}^{n\times r^2}$ satisfying $[\widetilde{C}^{pp}]_{ij} = ([\tilde{U}\tilde{U}^T]_{ij})^2 = [\tilde{V}\tilde{V}^T]_{ij}$, the following holds:*

$$\tilde{V} = [(\tilde{U}_1 \otimes \tilde{U}_1)^T \; (\tilde{U}_2 \otimes \tilde{U}_2)^T \; \cdots \; (\tilde{U}_n \otimes \tilde{U}_n)^T]^T, \tag{7.10}$$

where $\tilde{U}_i$ is the i-th row of $\tilde{U}$ and $\otimes$ is the Kronecker product.

Proof. See Appendix G.2. □

From Theorem 7, we have an efficient representation of a decomposed matrix considering post-processing and it bridges among the tasks in subspace clustering for scalability. Now, we are ready to perform spectral clustering on small $\tilde{V}$ instead of performing on $\widetilde{C}^{pp}$. Here, we assume that we use $\tilde{V}$ when $n \geq r^2$, which is common for large-scale problems. In the spectral clustering step, we first compute a degree matrix as $D = \text{diag}(\tilde{V}(\tilde{V}^T \mathbf{1})) \in \mathbb{R}^{n \times n}$, which can be computed efficiently with linear complexity. Based on D, a normalized Laplacian matrix L satisfies the following relation:

$$L = I - D^{-\frac{1}{2}} \widetilde{C}^{pp} D^{-\frac{1}{2}} = I - \breve{D}\breve{D}^T, \tag{7.11}$$

where $\breve{D} = D^{-\frac{1}{2}}\tilde{V}$ is a decomposed Laplacian matrix. Let $\breve{D} = U\Sigma V^T$ be SVD of $\breve{D}$, then, $L = U(I - \Sigma^2)U^T$. It is important to note here that the k largest singular vectors of $\breve{D}$ is the same as the k smallest eigenvector of L. Hence, we can also reduce the complexity by directly conducting SVD on $\breve{D}$, instead of computing the square matrix L and then performing EVD over L, whose complexity is $O(n^3)$, occupying the main complexity of spectral clustering. Then, we perform k-means over the singular vectors to obtain the final segmentation result. The overall procedure of the proposed method, named *scalable low-rank representation (SLR)*, is summarized in Algorithm 14. It is recommended that the former approach described in 7.2.2 with spectral clustering can be used for small-scale problems ($n \leq 1,000$, in general) and the solution proposed here is used for large-scale problems.

Proposition 2. *Suppose that we can observe clean data $\overline{X}$, where rank$(\overline{X}) = r^* \leq r$. Then, SLR finds cluster memberships of samples exactly in $O(n)$ time.*

Proof sketch. SLR gives an equivalent solution to al clustering problem with (7.9) followed by spectral clustering, where the block diagonal structure of (7.9) based on the rank-r approximation using $\overline{X}$ is guaranteed based on Theorem 6 and the work in [111]. Therefore, SLR solves the subspace clustering problem exactly

Algorithm 14 Scalable low-rank representation (SLR)

```
Input: normalized streaming data X_n = [x_1, ..., x_n]
for i = 1, ..., n do
    if med(S_i^T x_i) ≤ θ and θ_0 ≤ med(R̃_i^T x_i) then
        S_i ← [S_i, x_i]
        Update the summary matrix C^S using S_i
    else if θ < med(S_i^T x_i) and θ_0 < med(R̃_i^T x_i) then
        R_i ← [R_i, x_i]
    else
        Regard x_i as an outlier
    end if
end for
Construct C^Agg = [C^S, C^R]^T by (7.8)
Compute a post-processed matrix Ṽ by (7.10)
Compute D̆ = D^(-1/2) Ṽ where D is a degree matrix
Apply k-means to the k largest singular vectors of D̆
```

with linear time complexity. □

Complexity analysis. The computational complexity of the subspace clustering algorithms depends on the following three main tasks: (1) construction of an affinity matrix, (2) post-processing, and (3) spectral clustering. The proposed framework, SLR along with SSSC [63] do not perform the conventional spectral clustering step. Moreover, the proposed algorithm as well as LSR do not learn an affinity matrix iteratively (that is, their solutions are computed in closed form). The computational complexity of the proposed unified framework is $O(nr^4)$. This takes the linear complexity over n if r is considered as a constant over various-size samples. In other words, if the number of samples dominates the summary size, i.e., $n \gg r$, we can dramatically reduce the computational complexity (for example, see Table 7.5). The computational complexity of SSSC to $O(tq^3 + nq^2)$ where t is the number of iterations and q is the in-sample size.[4] Even though the complexity of SSC-OMP [65] is $O(ndk)$, where k is the size of the support set used in OMP, it still suffers from the heavy computational complexity due to the spectral clustering task. The memory complexity of the proposed framework is $O(nr^2)$, whereas the memory complexity of existing methods is $O(n^2)$, except SSSC, which has $O(q^2)$ complexity. The time and memory complexities of the proposed method along with existing algorithms are summarized in Table 7.1.

7.2.4 Nonlinear extension for SLR

The proposed framework is applied to more challenging problems where samples lie in a union of nonlinear manifolds, since conventional linear subspace clustering methods are hard to apply for the nonlinear subspace structure. Fortunately, the proposed framework is easy to extend to nonlinear subspace clustering as follows:

$$\min_{C} \|\phi(X) - \phi(X)C\|_F^2 + \lambda \|C\|_F^2, \tag{7.12}$$

where $\phi(\cdot) : \mathbb{R}^d \to \mathcal{H}$ is a nonlinear mapping function to a reproducing kernel Hilbert space $\mathcal{H}$. The optimal solution of the problem (7.12) is computed by using the kernel trick:

$$C_n = (\mathcal{K}_{XX} + \lambda I)^{-1} \mathcal{K}_{XX}, \tag{7.13}$$

where $\mathcal{K}_{XX} \in \mathbb{R}^{n \times n}$ is a kernel matrix such that $[\mathcal{K}_{XX}]_{ij} = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Note that the proposed summary representation with the unified scal-

[4] We have found that q is normally larger than r or similar to r^2 to get the reasonable performance for most problems in Section 4.1.1. Even worse, such a choice still shows unsatisfying performance compared to the proposed method.

Table 7.1: Complexity analysis of the compared algorithms for overall procedure including post-processing and spectral clustering. Time(A) denotes the time complexity for constructing an affinity matrix and Time(S) denotes the time complexity for post-processing and spectral clustering. t is the number of iterations in each iterative algorithm, q is the in-sample size, k is the size of the support set, and r is the summary size. For LRR, we describe the accelerated version [4].

Method	SSC	LRR	LSR	SSSC	SSC-OMP	ILSR	SLR
Time(A)	$O(tn^3)$	$O(t(nd^2+d^3))$	$O(n^2d)$	$O(tq^3+nq^2)$	$O(tndk)$	$O(n^3d)$	$O(nr^4)$
Time(S)	$O(n^3)$	$O(n^3)$	$O(n^3)$		$O(n^3)$	$O(n^3)$	
Memory	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(q^2)$	$O(n^2)$	$O(n^2)$	$O(nr^2)$

able pipeline can be straight-forwardly applied to the kernelized formulation. In this section, we consider two kernel functions: radial basis function (RBF) kernel function, $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|^2/2\sigma^2)$, and polynomial kernel function, $\kappa(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^T\boldsymbol{y} + \alpha)^\beta$, where σ, α, and β are parameters of the kernel functions.

7.3 Experimental Results

In this section, we apply the proposed method, SLR, to five datasets: synthetic data, Hopkins 155 dataset [55] for motion segmentation, Extended Yale B dataset [78] for face clustering, USPS dataset [113] for handwritten digits clustering, and HARUS dataset [124] for action clustering. Selected examples of the datasets are illustrated in Figure 7.4. We compare with well-known batch subspace clustering algorithms, SSC [16], LRR [4], and LSR [111], a nonlinear subspace clustering method, KSSC [125], and scalable methods, SSSC [63] and SSC-OMP [65], and the incremental approach of LSR (ILSR) described in Section 7.1, with respect to clustering accuracy and execution time. Furthermore, we compare with two large-scale spectral clustering algorithms: a spectral clustering method using the Nyström method with orthogonalization (Nyström) [126, 127] and the landmark-based spectral clustering method (LSC) [128] to demonstrate the proposed method with spectral clustering algorithms.

The clustering accuracy is computed as follows:

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\delta(p_i, map(q_i)), \tag{7.14}$$

where p_i and q_i are the i-th true and obtained labels, respectively, $\delta(a, b)$ is the Kronecker delta function, and $map(\cdot)$ is a mapping function to permute the obtained labels to match with the true labels, which is computed by Kuhn-Munkres algorithm [77]. In the experiments, we compute execution times of tested meth-

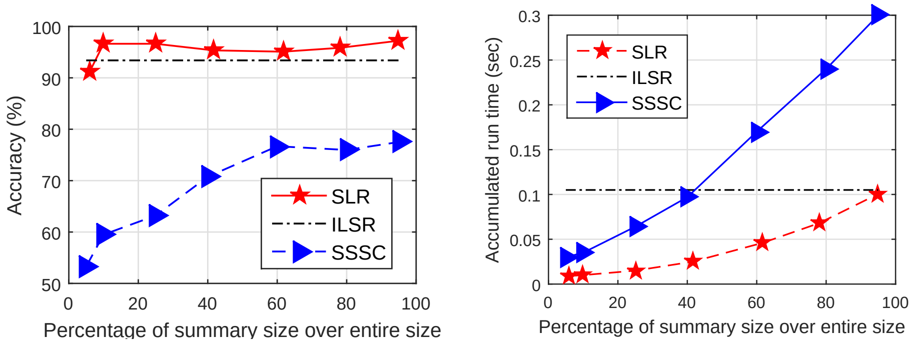
(a) (b) (c)

Figure 7.4: Typical examples from three datasets. (a) Hopkins 155 dataset for motion segmentation, (b) USPS dataset for handwritten digits clustering, and (c) Extended Yale B dataset for face clustering.

ods for whole tasks in subspace clustering, unless stated otherwise. We use the codes of compared methods provided by authors. For fair comparison, we set the parameters of all tested methods to achieve the best performance, unless stated otherwise.

7.3.1 Synthetic data

We first evaluated the performance of the proposed method compared with ILSR and SSSC according to various summary ratios or in-sample ratios, in order to verify the proposed summary representation. We generated an example which has five clusters, where each cluster has 50 samples with dimension of 50 and added Gaussian noises from $\mathcal{N}(0, 0.1)$. Figure 7.5 shows the average clustering accuracy and execution time according to the summary ratio for 50 different examples. We varied the summary and in-sample ratio from 5% to 95%. As shown in Figure 7.5(a), SLR outperforms SSSC for all cases. Furthermore, it gives higher accuracy than ILSR when the summary ratio is larger than about 10%. One possible reason



(a) (b)

Figure 7.5: Performance comparison on a synthetic example according to summary ratio. The example size is 50×250 with 5 clusters where each cluster has 50 samples. (a) Clustering accuracy. (b) Execution time.

is that the summary representation has a denoising effect by discarding noisy or meaningless samples as existing low-rank and sparse representation algorithms do in noisy scenarios. For execution time, the proposed method is much faster than SSSC and the difference gets larger when the summary ratio increases as shown in Figure 7.5(b). From the figures, the proposed summary representation shows its efficiency with excellent performance.

Then, we conducted our proposal, SLR, compared with ILSR and existing algorithms, SSC [16], LRR [16], LSR [111], SSSC [63], and SSC-OMP [65], to verify the efficiency of the proposed algorithms for large-size datasets when the number of samples dominates the summary size. We tested the proposed method on synthetic examples. We constructed a data matrix whose samples are randomly collected from five linear subspaces, where the number of randomly chosen basis vectors in each subspace is five. Then, we added a Gaussian noise matrix whose elements are generated from $\mathcal{N}(0, 0.1)$. We set the summary size to 25 and the

Table 7.2: Average clustering accuracy (%), execution time (sec), and speed-up gain over each compared method for SLR on synthetic problems with a large number of samples.

	n=15,000			n=30,000		
Method	Accuracy	Time	Speed-up	Accuracy	Time	Speed-up
SSC	94.0	>5.5h	$7,071\times$	95.3	>11.9h	$11,577\times$
LRR	96.6	3,279.1	$1,171\times$	99.2	8,617.2	$2,329\times$
LSR	**97.5**	3,706.8	$1,324\times$	**99.5**	7,420.1	$2,005\times$
SSSC	90.4	15.9	5.7×	92.8	43.2	11.7×
SSC-OMP	94.1	1,436.9	513×	96.1	5,479.7	1,481×
SLR	**97.5**	**2.8**	–	99.0	**3.7**	–

in-sample size of SSSC to 500 to get reasonable performance. The parameter λ of the proposed method is set to 500. We performed the proposed method for two scenarios, where $n = 15,000$ and $n = 30,000$. Table 7.2 shows the average performance of different methods from 10 independent runs. From the table, SLR gives the order-of-magnitude speed-up (roughly thousands of times faster for $n = 30,000$) over other methods including SSSC. SSSC is faster than other state-of-the-art algorithms, but it is slower than SLR with relatively poor performance. Even though SSC-OMP shows faster running time than SSC based on the basis pursuit formulation, it still fairly slow compared to ours, making it less applicable for large-scale problems.

In addition, we provide an experiment on robustness of the proposed summary representation. We generated an example with 5 classes each of which has 50 samples with dimension of 100 and added Gaussian noises from $\mathcal{N}(0, 10^{-2})$. In the

Table 7.3: Performance comparison on synthetic problems with outliers.

Method	SSC	LRR	LSR	SSSC	SLR
Accuracy (%)	96.6	91.6	94.9	51.5	**99.7**
Time (sec)	0.90	1.79	**0.02**	0.13	**0.02**

example, we replaces 10% randomly selected samples to outliers whose elements are uniformly generated from $[-25, 25]$. We set the summary ratio to roughly 20% ($\theta = 0.45$) and the minimum threshold θ_0 to 0.13 for the example. Table 7.3 shows the average performance of the methods from 30 different examples and Figure 7.6 illustrates the selected samples used for constructing the summary matrix. From the results, we can observe that the proposed method is robust against outliers by eliminating them and thus gives satisfactory performance.

7.3.2 Motion segmentation

Motion segmentation [55] is a task for clustering trajectories of rigidly moving objects based on tracked points along the frames. We applied the proposed algorithm to the Hopkins 155 database [55], which consists of 155 video sequences where there exist two or three motions. We compared SLR with other methods in terms of clustering performance and execution time for all sequences. Since this task is a small-scale problem, we solve SLR based on the reconstruction approach in (7.9), and we compute execution times of tested algorithm for the affinity construction task. We set the summary ratio of SLR to about 25% and the in-sample ratio of SSSC to 25%. We set the parameter λ of SLR to 5×10^{-4}. In the dataset, we use four measures over the clustering performance (mean, standard deviation (Std.), minimum, and median) motivated by the work in [53]. The average results

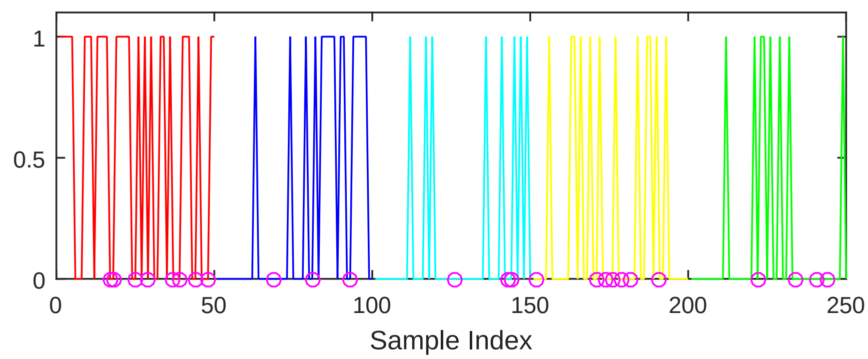


Figure 7.6: Selected samples (represented by 1) in the proposed summary representation to construct the summary matrix for a synthetic example with 10% outliers. A magenta circle indicates an outlier.

of the compared methods are shown in Table 7.4. From the table, we observe that most of the algorithms give the similar performance except SSSC which gives unsatisfactory performance for this problem. The execution time of SLR is much faster than that of SSC and LRR and is relatively faster than that of SSSC and ILSR. While LSR and SSC-OMP run slightly faster than SLR for this small-scale dataset, SLR is much faster than LSR for larger datasets on average as shown in other experiments.

7.3.3 Face clustering

We evaluated our proposal for face clustering on the Extended Yale B dataset [78], which consists of 38 subjects and each subject has 64 frontal face images under various illumination changes. In the dataset, we used the first c classes, where $c \in \{3, 5, 8, 10\}$ with samples of 64 for each class. Then, we reduced each image to a $9c$ dimensional vector using PCA. Similar to the previous problem, this task is also a small-scale problem. We solve SLR by the reconstruction approach with

Table 7.4: Performance comparison with respect to clustering accuracy (%) and execution time (sec) on the Hopkins 155 dataset for motion segmentation.

Method	Mean	Std.	Min	Median	Time
SSC	96.46	9.1	52.8	**100**	1.36
LRR	**96.53**	**8.0**	58.2	99.7	1.03
LSR	95.96	10.4	52.1	99.8	**0.04**
SSSC	80.80	18.0	41.3	84.9	0.18
SSC-OMP	96.33	8.53	58.7	99.8	**0.04**
ILSR	95.96	10.4	52.1	99.8	0.30
SLR	95.98	8.7	**61.9**	**100**	0.06

the same summary ratio and execution time as described in the previous subsection. We set the parameter λ of SLR to 10^{-2}. Figure 7.7 shows the clustering accuracy of all methods at different numbers of clusters with average clustering accuracy. The proposed method performs competitively compared to other methods on average as described in the results. SSC-OMP gives better performance than ours when the number of clusters is small, its results drop sharply when $c \geq 8$. The performance of SSSC gets worse considerably than other algorithms when the number of clusters increases for the in-sample ratio of 25%. Figure 7.8 shows the execution times of different methods for the case when $c = 10$. As shown in Figure 7.8(a), ours shows satisfying execution time for this small-size problem and comparable to SSC-OMP. In addition, we compared the proposed method with the naïve algorithm, ILSR, with respect to the time as more samples are introduced sequentially. As shown in Figure 7.8(b), the proposed method runs in real-time, whereas ILSR gets slower rapidly when the number of samples

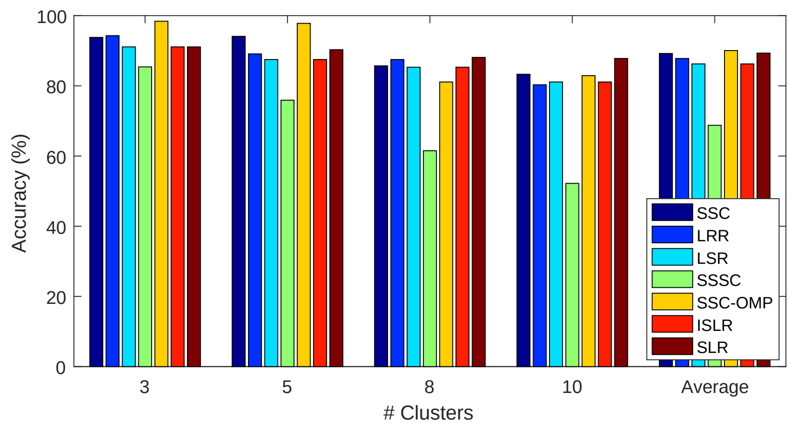


Figure 7.7: Clustering accuracy (%) on the Extended Yale B dataset.

increases.

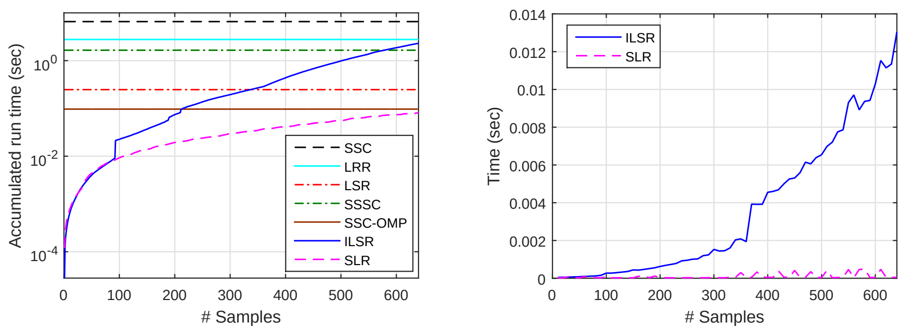
7.3.4 Handwritten digits clustering

We tested the performance of the proposed method for clustering handwritten digits. We used the USPS dataset [113], which consists of 9,298 gray-scale images with 10 classes where each image is represented using a 16×16 matrix. In the dataset, we selected the first 1,000, 3,000, 5,000, and 9,298 samples to verify the performance of the methods with regard to the number of samples from small-scale to large-scale. In addition, we augmented the dataset by duplicating the dataset and shuffling samples in the augmented dataset (a total of 18,596 samples) to perform on a larger dataset. We set the summary size of SLR to 30 which results in $\tilde{V} \in \mathbb{R}^{n\times 900}$ in (7.10) and the in-sample ratio of SSSC to $min(\lceil 0.1n \rceil, 900)$ to get reasonable performance. We set $\lambda = 3$.

Table 7.5 shows the performance of different methods. The proposed algorithm outperforms other methods on average in terms of clustering accuracy and execution time. It gives an accuracy of over 70% on average and is much faster than

Table 7.5: Performance comparison with respect to clustering accuracy (%), execution time (sec), and speed-up gain over each method for SLR on the USPS dataset.

	n=1,000		n=3,000		n=5,000		n=9,298		n=18,596		
Method	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Acc.	Time	Speed-up
SSC	66.6	26.0	58.0	253.6	60.6	1,069.3	**71.1**	5,317.5	30.0	>10.9h	$7,007\times$
LRR	71.6	23.4	66.9	82.5	59.6	268.8	57.5	1,083.6	50.1	6,755.2	$1,206\times$
LSR	72.0	1.4	**70.5**	32.1	66.7	148.1	54.9	905.8	53.9	6,561.2	$1,171\times$
SSSC	29.9	1.2	40.7	3.8	57.1	8.0	60.4	19.3	59.6	17.5	$3.1\times$
SSC-OMP	37.9	**0.7**	37.4	15.5	37.3	59.8	39.0	373.9	19.3	963.1	$172\times$
ILSR	72.0	11.8	70.5	339.2	66.7	1,574.1	54.9	9,952.8	53.9	>30h	$19,286\times$
SLR	**72.4**	0.8	70.1	**1.4**	**70.0**	**2.1**	70.3	**3.3**	**69.4**	**5.6**	–



(a) (b)

Figure 7.8: Execution time (sec) on the Extended Yale B dataset. Time was computed when the number of clusters is 10. (a) Accumulated run time (log scale). (b) Time at each iteration.

existing algorithms with an order-of-magnitude speed-up. Even though SSSC is faster than existing algorithms, it performs poorer than others. Likewise, SSC-OMP give poorer performance than the others in this problem, and even its execution time increases sharply compared to the proposed method. Here, we have found that the clustering accuracy of SSC decreases substantially when the number of samples is 18,596 for a fixed parameter. The reason is that the sparsest representation of SSC may not cover all samples in a subspace, leading to fractions in a subspace. Whereas, our approach provides excellent performance mainly due to its grouping effect with robust representation generated from sparse and low-rank connections.

7.3.5 Action clustering

We also provide the experimental results on more challenging problem, action clustering [124]. We evaluated the proposed method on the HARUS dataset

Table 7.6: Performance comparison with respect to clustering accuracy (%), execution time (sec), and speed-up gain over each method for SLR on the HARUS dataset.

	n=5,000			n=10,299			n=20,598		
Method	Acc.	Time	Speed-up	Acc.	Time	Speed-up	Acc.	Time	Speed-up
SSC	44.0	1,240.9	775×	48.0	8,135.8	2,624×	26.7	>4.8h	3,830×
LRR	48.4	300.9	188×	51.4	1,394.5	450×	50.8	2,725.1	606×
LSR	63.3	143.2	89×	64.1	1,176.9	380×	63.2	2,363.8	525×
SSSC	38.9	9.9	6×	52.3	35.1	11×	51.2	62.5	14×
SSC-OMP	49.3	60.8	38×	46.8	516.7	167×	22.3	1304.6	290×
KSSC	65.2	1,660.1	1,037×	64.6	>3.7h	4,315×	64.5	>4.8h	3,875×
Nyström	62.5	1.9	1.2×	67.7	11.7	3.8×	66.2	54.8	12.2×
LSC	64.1	**1.2**	0.75×	68.3	**2.2**	0.7×	68.1	**2.9**	0.64×
SLR	69.0	1.6	–	66.6	3.1	–	68.9	4.5	–
KSLR(G)	70.1	2.5	1.6×	71.1	5.0	1.6×	71.6	11.3	2.5×
KSLR(P)	**78.1**	2.1	1.3×	**76.6**	3.7	1.2×	**75.2**	7.3	1.6×

[124], which consists of 10,299 samples over six action classes (walking, walking up/down-stairs, sitting, standing, laying). Since a sample in the dataset may not be represented by a linear combination of other samples, we applied the nonlinear extensions of the proposed algorithm: KSLR(P) and KSLR(G) using polynomial kernel and RBF kernel functions, respectively, described in Section 7.2.4. We also tested kernel SSC (KSSC) [125], a recently proposed nonlinear subspace clustering algorithm, and two spectral clustering algorithms, Nyström [126, 127] and LSC [128]. We made three scenarios by selecting first 5,000 and 10,299 samples and augmenting additional scenario, where the number of samples is 20,598. We set the summary and in-sample size to the same value stated in the previous problem. The parameters of the kernel functions are set to $\sigma = 1$, $\alpha = 0$, and $\beta = 5$. We set $\lambda = 10^3$ for SLR and $\lambda = 1$ for KSLR.

Table 7.6 shows the clustering accuracy and execution time of the compared algorithms for the action clustering tasks. From the table, the proposed linear method, SLR, gives better performance than other methods except its nonlinear extensions. Even, they perform better than KSSC for all scenarios. The nonlinear extensions of the proposed method outperform the existing methods. Especially, the extension based on the polynomial kernel function gives the best performance. As for the execution time, existing subspace clustering methods are hundreds or thousands times slower than the proposed method and also SSSC is 14 times slower than SLR when $n = 20,598$. Another scalable subspace clustering algorithm, SSC-OMP, gives unsatisfying results for both clustering accuracy and execution time (290× slower). Even if Nyström and LSC give the competitive execution time, they perform poorer than the proposed method. From the table, we can observe that the proposed framework is scalable, efficient, and can be used for large-scale problems.

7.4 Summary

In this chapter, we have proposed an end-to-end scalable learning algorithm for large-scale subspace clustering based on the summary representation and an efficient integrated pipeline with post-processing and spectral clustering. The summary representation accelerates learning of an affinity matrix efficiently and robustly with excellent performance and the efficient integration with post-processing and spectral clustering achieves linear time complexity, making it suitable for large-scale problems. The proposed framework has been applied to various problems with different scales and shown its excellent performance and efficiency for large-scale problems.

Chapter 8

Conclusion and Future Work

From recent advances in digital technology, demands for processing power of a computing device have been highly increased. However, the advancement of processing power does not follow the geometric growth of the amount of data, called big data. What is more, the curse of dimensionality even makes an algorithm difficult to handle such massive data, making it less applicable. Fortunately, we can exploit key information from data by the blessing of dimensionality from the concept of sparsity or low-rank-ness.

One of the efficient exploitation tools, sparse representation has been widely used to select informative entries in a bunch of data. However, most of the successful algorithms are based on the convex relaxed approach using the l_1-norm, which is only efficient for convex problems and can lose its significance when conducting on inherently nonconvex problems. As a remedy of the weakness of existing problems, we have presented a new nonconvex sparsity measure for many nonconvex problems. The proposed measure embraces both l_0- and l_1-norms and possesses slowly vanishing gradients to help drawing solutions of an optimization algorithm to sparse points. Experiments on three important sparse representation problems have verified that the proposed method performs favorably against those of state-of-the-art algorithms.

Low-rank representation, another efficient exploitation tool, has been also very popular method to reduce the dimension of data safely without much losing its original information. But, the conventional algorithms are vulnerable to corruptions and algorithms handling outliers are quite slow to get a reasonable solution, making them not applicable for practical application in the presence of outliers. To address the issue of robustness and efficiency, we have first proposed an efficient algorithm based on the robust measure, the l_1-norm, and solved it using the alternating rectified gradient method, which finds a gradient to reach a stationary point quickly. Then, we have presented a regularized formulation with an orthogonality constraint to cope with overfitting and running speed of an algorithm and solved it under the augmented Lagrangian framework. It can handle a rank uncertainty issue flexibly by a rank estimation strategy for practical real-world problems. In addition, we have studied a structured matrix approximation problem which is used in a nonparametric Bayesian approach. Numerical experiments have demonstrated the robustness and efficiency of the proposed algorithms for several benchmark data sets.

The above low-rank representation methods assume that the rank of data is fixed. In order to address the rank uncertainty issue with the fixed-rank problem, we have studied the well-known elastic-net regularizer which compromises both ridge and lasso regressions and is used to analyze the rank of a matrix by regularizing singular values. We have developed a robust and stable algorithm with automatic rank estimation from the maximum rank defined by users. The strong convexity from the regularizer alleviates the instability problem by shrinking and correcting inaccurate singular values in the presence of unwanted noises.

It is extended to a joint optimization problem to handle data lying in a union of multiple subspaces based on the elastic-net regularization of singular values. Experimental results on several benchmark problems have proved the superiority of the proposed algorithm using the regularizer.

Motivated from the previous elastic-net regularizer, we have applied the regularizer to a subspace clustering task, where we regularize a coefficient matrix which reveals a subspace structure for grouping effect among highly correlated samples. Hence, we have proposed two robust group subspace clustering algorithms by extending conventional sparse and low-rank representation algorithms with explicit subspace grouping. We have shown that the proposed methods capture the similarities among data samples collected from the same subspace, theoretically and empirically. While the subspace clustering algorithms successfully applied to a number of problems, they are still not applicable for large-scale or streaming data due to their expensive computational cost. As a remedy for the high computational requirement, we have presented an end-to-end solution to reduce the complexity of all tasks in subspace clustering, by assuming the low-rank-ness of data. The proposed algorithms have been applied to well-known clustering tasks, outperforming other state-of-the-art algorithms.

For future work, more theoretical analysis of the proposed algorithms on the convergence rate and error bound will be studied. Furthermore, we will apply the concept of sparsity and low-rank-ness to other challenging applications to be explored in computer vision and robotic fields. In addition, we will extend the nonconvex sparsity measure to a 2D sparsity problem, that is, low-rank representation problem, because the ideal rank function is nonconvex and most of the low-rank matrix approximations are also nonconvex. Due to the unfavorable computational complexities of the conventional methods including our proposals for the low-rank representation, we will explore scalable approaches to reduce both time and memory complexities for a practical use. Lastly, following the recent advances in deep learning, we will apply the sparse and low-rank representation to deep learning architectures in order to represent the architectures concisely with considerably low number of parameters.

Appendices

Appendix A

Derivations of the LRA Problems

For the LRA problem, we apply SVG for modeling sparse errors, whose problem formulation, termed LRA-SVG, is constructed as follows:

$$\min_{\boldsymbol{E},\boldsymbol{M}} \|\mathcal{P}_{\Omega_{\boldsymbol{X}}}(\boldsymbol{E})\|_{\text{SVG}}^{\epsilon}, \text{ s.t. } \boldsymbol{E} = \boldsymbol{X} - \boldsymbol{M}, \quad \text{rank}(\boldsymbol{M}) \leq r. \tag{A.1}$$

The augmented Lagrangian of (A.1) is constructed as

$$\begin{aligned} \mathcal{L}(\boldsymbol{E}, \boldsymbol{M}, \boldsymbol{\Pi}) &= \|\mathcal{P}_{\Omega_{\boldsymbol{X}}}(\boldsymbol{E})\|_{\text{SVG}}^{\epsilon} \\ &+ \langle \boldsymbol{\Pi}, \boldsymbol{E} - \boldsymbol{X} + \boldsymbol{M} \rangle + \frac{\gamma}{2}\|\boldsymbol{E} - \boldsymbol{X} + \boldsymbol{M}\|_F^2, \end{aligned} \tag{A.2}$$

such that $\text{rank}(\boldsymbol{M}) \leq r$. Based on (A.2), we obtain an algorithm based on the following steps:

$$\boldsymbol{E}_+ \leftarrow \min_{\boldsymbol{E}} \|\mathcal{P}_{\Omega_{\boldsymbol{X}}}(\boldsymbol{E})\|_{\text{SVG}}^{\epsilon} + \frac{\gamma}{2}\|\boldsymbol{D} + \frac{\boldsymbol{\Pi}}{\gamma}\|_F^2, \tag{A.3}$$

$$\breve{\boldsymbol{M}} \leftarrow \min_{\boldsymbol{M}} \frac{\gamma}{2}\|\boldsymbol{D} + \frac{\boldsymbol{\Pi}}{\gamma}\|_F^2, \tag{A.4}$$

$$\boldsymbol{M}_+ \leftarrow \boldsymbol{U}_r \mathcal{S}_{\frac{1}{\gamma}}[\boldsymbol{\Sigma}_r]\boldsymbol{V}_r^T, \tag{A.5}$$

$$\boldsymbol{\Pi}_+ \leftarrow \boldsymbol{\Pi} + \gamma \boldsymbol{D}, \tag{A.6}$$

where $\boldsymbol{D} \triangleq \boldsymbol{E} - \boldsymbol{X} + \boldsymbol{M}$, $\boldsymbol{\Pi}$ denotes the Lagrange multiplier, and γ is a positive penalty parameter. For (A.5), we collect r largest singular values and their corresponding singular vectors computed by the singular value decomposition (SVD) on $\breve{\boldsymbol{M}}$ obtained from (A.4), i.e., $[\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}] = svd(\breve{\boldsymbol{M}})$. To solve for $\boldsymbol{E}$, we consider the following optimization problem for each element e_{ij} indexed by $\Omega_{\boldsymbol{X}}$:

$$\min_{e_{ij}} \frac{|e_{ij}|}{|e_{ij}| + \epsilon} + \frac{\gamma}{2}(e_{ij} - x_{ij} + m_{ij} + \frac{\pi_{ij}}{\gamma})^2 \tag{A.7}$$

where x_{ij}, m_{ij}, and π_{ij} are the $(i,j)^{th}$ elements of $\boldsymbol{X}$, $\boldsymbol{M}$, and $\boldsymbol{\Pi}$, respectively. The solution of (A.7) can be found by an efficient computation for each element separately as explained in Chapter 3. For another element e_{kl} indexed by $\overline{\Omega}_{\boldsymbol{X}}$, where $\overline{\Omega}_{\boldsymbol{X}}$ is a complementary support set of $\boldsymbol{X}$, we obtain $e_{kl} \leftarrow x_{kl} - m_{kl} - \frac{\pi_{kl}}{\gamma}$.

For the tested algorithms based on the same ADMM framework, such as LRA-L1, LRA-CapL1, and LRA-MCP, we simply switch the penalty function $\|\cdot\|_{\text{SVG}}^{\epsilon}$ in (A.1), (A.2), and (A.3) to a nonconvex penalty function and solve its corresponding optimization problem. As an example, LRA-L1 compared in Chapter 3 considers the following optimization problem when solving $\boldsymbol{E}$ in the ADMM framework:

$$\boldsymbol{E}_{+} \leftarrow \min_{\boldsymbol{E}} \|\mathcal{P}_{\Omega_{\boldsymbol{X}}}(\boldsymbol{E})\|_1 + \frac{\gamma}{2}\|\boldsymbol{D} + \frac{\boldsymbol{\Pi}}{\gamma}\|_F^2, \tag{A.8}$$

and its solution is computed as follows:

$$\boldsymbol{E}_{+} \leftarrow \mathcal{P}_{\Omega_{\boldsymbol{X}}}(\mathcal{S}_{\frac{1}{\gamma}}(\boldsymbol{Y})) + \mathcal{P}_{\overline{\Omega}_{\boldsymbol{X}}}(\boldsymbol{Y}), \tag{A.9}$$

where $\boldsymbol{Y} \triangleq \boldsymbol{X} - \boldsymbol{M} - \frac{\boldsymbol{\Pi}}{\gamma}$ and $\mathcal{S}_{\gamma}(t) = \text{sign}(t)\max(|t| - \gamma, 0)$ is the shrinkage operator [43] for a scalar variable t. Other problems based on the nonconvex penalty functions described in Chapter 3 to solve for $\boldsymbol{E}$ can be solved efficiently by the work in [33].

Appendix B

Proof of Lemma 1

The first two assumptions in Assumption 2 are similar to some of our criteria: *Symmetry* and *Monotonicity*, respectively. Thus, it is straightforward to show the symmetry of SVG. By taking a derivative of ϕ_λ for $x > 0$, $\phi'_\lambda = \frac{\lambda}{\epsilon(1+\frac{x}{a\epsilon})^{a+1}} > 0$, we can check the nondecreasing nature on the nonnegative real-line. For the third assumption, i.e., $(\frac{\phi_\lambda(x)}{x})' \leq 0$, we can verify based on the following relation for $x > 0$:

$$(\frac{\phi_\lambda(x)}{x})' \leq 0 \quad \Leftrightarrow \quad x\phi'_\lambda(x) - \phi_\lambda(x) \leq 0. \tag{B.1}$$

Let $h_\lambda(x) \triangleq x\phi'_\lambda(x) - \phi_\lambda(x)$ which should be proved as a decreasing function. If $h_\lambda(0) \leq 0$ and $h'_\lambda(x) \leq 0$, then $h_\lambda(x) \leq 0$ for $x > 0$. Since we have $h_\lambda(0) = 0 \cdot \phi'_\lambda(0) - \phi_\lambda(0) = 0$ and $h'_\lambda(x) = \phi'_\lambda(x) + x\phi''_\lambda(x) - \phi'_\lambda(x) = x\phi''_\lambda(x) < 0$ from our *Smoothness* criterion, $h_\lambda(x) \leq 0$ is satisfied for $x > 0$, and thus $(\frac{\phi_\lambda(x)}{x})' \leq 0$. For the fourth assumption, we can easily check $\lim_{x\to 0^+} \phi'_\lambda(x) = \frac{\lambda}{\epsilon}$ using the following equation described in Chapter 3,

$$\phi_{\lambda=1}(x) = 1 - \frac{1}{(1+\frac{x}{a\epsilon})^a}, \tag{B.2}$$

for $a > 0$, thus we obtain $L = \frac{1}{\epsilon}$. For the last condition, we take another derivative:

$$\rho''_\lambda(x) = \begin{cases} -\dfrac{(a+1)\lambda}{a\epsilon^2} \cdot \dfrac{1}{(1+\frac{x}{a\epsilon})^{a+2}} + \mu, & \text{if } \ x > 0, \\ -\dfrac{(a+1)\lambda}{a\epsilon^2} + \mu, & \text{if } \ x = 0, \\ -\dfrac{(a+1)\lambda}{a\epsilon^2} \cdot \dfrac{1}{(1+\frac{-x}{a\epsilon})^{a+2}} + \mu, & \text{if } \ x < 0. \end{cases} \tag{B.3}$$

Since $\phi''_\lambda(x)$ has lower bound of $-\frac{(a+1)\lambda}{a\epsilon^2}$, it is true that there exists $\mu = \frac{(a+1)\lambda}{a\epsilon^2} > 0$ satisfying the convexity of $\rho_{\lambda,\mu}(x)$.

Appendix C

Proof of Proposition 1

Since the proposed measure, SVG, is one of our representative family, we show by proving the properties in Proposition 1 for our family. We redefine the family of curves, called SVGF, as follows:

$$\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon} \triangleq y(\boldsymbol{x}) = 1 - \frac{1}{(1 + \frac{|\boldsymbol{x}|}{a\epsilon})^a}, \tag{C.1}$$

where a and ϵ are parameters of the family as defined in Chapter 3. If $a = 1$, it becomes the proposed measure.

Proposition 3. *SVGF satisfies the following properties:*

1. $\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon} \leq \|\boldsymbol{x}\|_0 \ \forall a, \epsilon$ *and* $\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon} \to \|\boldsymbol{x}\|_0$ *if* $\epsilon \to 0$.

2. $\epsilon\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon} \leq \|\boldsymbol{x}\|_1 \ \forall a, \epsilon$ *and* $\epsilon\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon} \to \|\boldsymbol{x}\|_1$ *if* $\epsilon \to \infty$.

Proof. Assume a and ϵ in $\|\boldsymbol{x}\|_{SVGF}^{a,\epsilon}$ are positive. We simply show the proposition for a scalar case, but its extension to a vector case is straightforward. It is easily checked that $y(x) = 0$ if $x = 0$ and $y(x) \leq 1$ if $x \neq 0$, thus we verify that SVGF always lower than or equal to the l_0-norm for all x regardless of ϵ. If ϵ goes to zero, $\frac{1}{(1+\frac{|x|}{a\epsilon})^a} \to 0$ when $x \neq 0$, then $y(x) \to 1$ and the asymptotic convergence to the l_0-norm holds.

Note that both $y(x)$ and the l_1-norm are symmetric around zero and nonnegative (with $y(0) = 0$). Then, $\epsilon y(x)$ is lower than or equal to the l_1-norm, since $\epsilon y'(x) = \frac{1}{(1+\frac{x}{a\epsilon})^{a+1}} \leq 1$ for all nonnegative x. This also holds for $x < 0$. Finally, in order to show that $\epsilon y(x)$ asymptotically converges to $|x|$ if $\epsilon \to \infty$, we use the following relation:

$$\lim_{\epsilon\to\infty} \epsilon y = \lim_{\beta\to 0} \frac{1}{\beta}(1 - \frac{1}{(1+\frac{\beta|x|}{a})^a}) \triangleq \lim_{\beta\to 0} \frac{f(\beta)}{g(\beta)}, \tag{C.2}$$

where $f(\beta) = 1 - \frac{1}{(1+\frac{\beta|x|}{a})^a}$ and $g(\beta) = \beta \triangleq \frac{1}{\epsilon}$. Since $\lim_{\beta\to 0} f(\beta) = \lim_{\beta\to 0} g(\beta) = 0$, $g'(\beta) = 1 \neq 0$, and $\lim_{\beta\to 0} \frac{f'(\beta)}{g'(\beta)}$ exists, we have the following results by the L'Hospital's rule:

$$\lim_{\beta\to 0} \frac{f(\beta)}{g(\beta)} = \lim_{\beta\to 0} \frac{f'(\beta)}{g'(\beta)} = \lim_{\beta\to 0} \frac{a\frac{|x|}{a}(1+\frac{\beta|x|}{a})^{-a-1}}{1} = |x|, \tag{C.3}$$

which completes the proof. $\square$

Appendix D

Proof of Theorem 1

Theorem 1. *l_1-ARG_D converges to a subspace-wise local minimum irrespective of the initial point under the three conditions.*

We will show that l_1-ARG$_D$ satisfies these conditions in order to prove its global convergence. We prove the conditions only for the case of updating X while P is orthogonal, without loss of generality, and the condition for updating P can be proved similarly.

Proposition 4. *The sequence (P_k, X_k) produced by l_1-ARG_D is contained in a compact set.*

Proof. Since l_1-ARG$_D$ is a descent algorithm, it only chooses a point that does not increase the cost function, and always satisfies the relation $||Y - P_k X_k||_1 \leq ||Y||_1$ for an appropriate choice of P_0 and X_0. Since P_k is orthogonal,

$$\begin{aligned} &||Y||_1^2 \geq ||Y - P_k X_k||_1^2 \geq ||Y - P_k X_k||_F^2 \\ &\geq (||Y||_F - ||P_k X_k||_F)^2 = (||Y||_F - ||X_k||_F)^2. \end{aligned} \tag{D.1}$$

From this, we obtain the following relation:

$$||Y||_F - ||Y||_1 \leq ||X_k||_F \leq ||Y||_F + ||Y||_1. \tag{D.2}$$

Therefore, X_k is contained in a bounded and closed set, i.e., a compact set. Similarly, we can show P_k is contained in a compact set. Therefore, (P_k, X_k) is contained in a compact set. □

Condition 2 can also be proved as follows.

Proposition 5. *$J(P_k, X_k)$ is strictly decreasing for (P_k, X_k) that is not subspace-wise local minimum.*

Proof. If (P, X) is not a subspace-wise local minimum, $||Y - P(X + \Delta X)||_1 < ||Y - PX||_1$ for some ΔX. Since $J(P, X)$ is a convex function for a fixed P, the following relation is satisfied for any constant ν, $0 \leq \nu \leq 1$:

$$\begin{aligned} &||Y - P(X + \nu\Delta X)||_1 \\ &\leq (1-\nu)||Y - PX||_1 + \nu||Y - P(X + \Delta X)||_1. \end{aligned} \tag{D.3}$$

Now we consider the following equation:

$$\begin{aligned} &f_\eta(X, 0) - f_\eta(X, \nu\Delta X) \\ &= ||Y - PX||_1 - ||Y - P(X + \nu\Delta X)||_1 - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &\quad \geq ||Y - PX||_1 - (1-\nu)||Y - PX||_1 \\ &\quad - \nu||Y - P(X + \Delta X)||_1 - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &= \nu\{||Y - PX||_1 - ||Y - P(X + \Delta X)||_1\} - \frac{\nu^2}{2\eta}||\Delta X||_F^2 \\ &= \nu a_1 - \frac{\nu^2}{2}a_2, \end{aligned} \tag{D.4}$$

where $a_1 = ||Y - PX||_1 - ||Y - P(X + \Delta X)||_1$ and $a_2 = \frac{1}{\eta}||\Delta X||_F^2$. If $0 < \nu < \frac{2a_1}{a_2}$, $f_\eta(X, 0) - f_\eta(X, \nu\Delta X)$ is larger than 0, which means that there exists $\nu\Delta X$ that satisfies $f_\eta(X,0) > f_\eta(X,\nu\Delta X) \geq f_\eta(X,\Delta X^*)$. Therefore, according to (4.44), l_1-ARG$_D$ will find a direction $\Delta X'(=\nu\Delta X)$ that satisfies

$$\begin{aligned} f_\eta(X,0) - f_\eta(X,\Delta X') &\geq \beta(f_\eta(X,0) - f_\eta(X,\Delta X^*)) > 0, \\ ||Y - PX||_1 &> ||Y - P(X+\Delta X')||_1 + \frac{1}{2\eta}||\Delta X'||_F^2, \end{aligned} \tag{D.5}$$

which is a strictly descending direction when (P_k, X_k) is not in the solution set.

□

Now, in order to prove the condition 3, we first show that ΔX^* is a continuous function w.r.t. X and η.

Proposition 6. *If* $X_k \to \overline{X}$ *and* $\eta_k \to \overline{\eta}$, *then* $\Delta X_k^* \to \Delta\overline{X}^* = \arg\min_{\Delta X} f_{\overline{\eta}}(\overline{X}, \Delta X)$.

Proof. We first state some facts in order to prove the proposition. First, the optimal sequence $\{\Delta X_k^*\}$ is obviously contained in a bounded and closed set, i.e.,

$$\begin{aligned} \frac{1}{2\eta_k}||\Delta X_k^*||_F^2 \leq f_{\eta_k}(X_k, \Delta X_k^*) &\leq f_{\eta_k}(X_k, 0) \\ &= ||Y - PX_k||_1 \leq ||Y||_1. \end{aligned} \tag{D.6}$$

(This can also be deduced from the fact that the domain of X_k is compact.) Second, ΔX_k^* satisfies the relation $f_{\eta_k}(X_k, \Delta X_k^*) \leq f_{\eta_k}(X_k, \Delta X)$ for any ΔX which is the very definition of ΔX_k^*. Third, $f_\eta(X, \Delta X)$ is a strictly convex function w.r.t. ΔX for a given (X, η) because of the term $\frac{1}{2\eta}||\Delta X||_F^2$. Hence, $f_\eta(X, \Delta X)$ has a unique optimal ΔX^*. Since $\{\Delta X_k^*\}$ is bounded, there must exist a convergent subsequence $\{\Delta X_{k_n}^*\}$, i.e., $\Delta X_{k_n}^* \to \Delta\breve{X}$. Then, for any ΔX, we can obtain the following relation:

$$\begin{aligned} f_{\overline{\eta}}(\overline{X}, \Delta\breve{X}) &= \lim_{n\to\infty} f_{\eta_{k_n}}(X_{k_n}, \Delta X_{k_n}^*) \\ &\leq \lim_{n\to\infty} f_{\eta_{k_n}}(X_{k_n}, \Delta X) = f_{\overline{\eta}}(\overline{X}, \Delta X). \end{aligned} \tag{D.7}$$

The only $\Delta\breve{X}$ that satisfies the relation is $\Delta\overline{X}^*$. Thus, any convergent subsequence of $\{\Delta X_k^*\}$ has the same limit $\Delta\overline{X}^*$. Since ΔX_k^* is bounded and all the convergent subsequences has the same limit, ΔX_k^* converges to the limit $\Delta\overline{X}^*$. □

Next, we define a function $K(X, \eta, \Delta X)$ assuming that X is not a local minimum:

$$K(X, \eta, \Delta X) \triangleq \frac{f_\eta(X, 0) - f_\eta(X, \Delta X)}{f_\eta(X, 0) - f_\eta(X, \Delta X^*)}. \tag{D.8}$$

Proposition 7. *$K(X, \eta, \Delta X)$ is continuous for non-local-minimum X.*

Proof. $K(X, \eta, \Delta X)$ is composed of $f_\eta(X, 0), f_\eta(X, \Delta X)$, and $f_\eta(X, \Delta X^*)$ with subtraction and division operations. Also $f_\eta(X, 0)$ and $f_\eta(X, \Delta X)$ are continuous functions w.r.t. $X, \Delta X$, and η ($\eta_{\min} \leq \eta \leq \eta_{\max}$), and so is $f_\eta(X, \Delta X^*)$ by Proposition 3. Moreover, $f_\eta(X, 0) > f_\eta(X, \Delta X^*)$ when X is not a local minimum. Therefore $K(X, \eta, \Delta X)$ is also continuous. □

Now finally, we prove that l_1-ARG$_D$ satisfies condition 3. Since l_1-ARG$_D$ uses an exact line-search, which is a closed mapping [83], we only need to prove that the procedure for finding a descent direction is a closed mapping at a non-local minimum. To do this, we define two point-to-set mappings G and H. $\Delta X \in G(X, \eta)$ determines the descending direction, and $\eta' \in H(\eta)$ determines η, where η' is the value of η in the next iteration. $H(\eta)$ is defined as $H(\eta) = [\eta_{\min}, \eta_{\max}]$ (η' is determined independently, regardless of η), and $G(X, \eta)$ is defined as

$$G(X, \eta) = \{\Delta X | f_\eta(X, 0) - f_\eta(X, \Delta X) \geq \beta(f_\eta(X, 0) - g_\eta(V))\}.$$

If X is not a local minimum, then this is the same as $G(X, \eta) = \{\Delta X | K(X, \eta, \Delta X) \geq \beta\}$.

Proposition 8. *Let Q be a point-to-set mapping defined as $(\Delta X, \eta') \in Q(X, \eta)$ where $\Delta X \in G(X, \eta')$ and $\eta' \in H(\eta)$. Then, Q is a closed mapping.*

Proof. Here, H is obviously a closed mapping and the domain of η is a bounded set, hence $Q(X, \eta)$, which is a composition of G and H, is a closed mapping if G is a closed mapping. Since K is a continuous function w.r.t. $(X, \eta, \Delta X)$, $K(\overline{X}, \overline{\eta}, \Delta\overline{X}) = \lim_{k\to\infty} K(X_k, \eta_k, \Delta X_k) \geq \beta$ if $X_k \to \overline{X}, \eta_k \to \overline{\eta}$, and $\Delta X_k \to \Delta\overline{X}$. Therefore, G is a closed-mapping. $\square$

Q describes the behavior of finding the descent direction in l_1-ARG$_D$. The proposed method is globally convergent by the proofs for the three conditions.

Appendix E

Proof of Theorem 2

Theorem 2. *Let* $G \triangleq (P, X, D, \widehat{D}, \Lambda_1, \Lambda_2)$ *and* $\{G^j\}_{j=1}^{\infty}$ *be generated by FactEN. Assume that* $\{G^j\}_{j=1}^{\infty}$ *is bounded and* $\lim_{j\to\infty}\{G^{j+1} - G^j\} = 0$. *Then, any accumulation point of* $\{G^j\}_{j=1}^{\infty}$ *satisfies the KKT conditions. In particular, whenever* $\{G^j\}_{j=1}^{\infty}$ *converges, it converges to a KKT point.*

Proof. First, we get the Lagrange multipliers $\Lambda_{1+}, \Lambda_{2+}$ from (5.20)

$$\begin{aligned} \Lambda_{1+} &= \Lambda_1 + \beta(D - PX) \\ \Lambda_{2+} &= \Lambda_2 + \beta(\widehat{D} - D), \end{aligned} \tag{E.1}$$

where Λ_{i+} is a next point of Λ_i in a sequence $\{\Lambda_i^j\}_{j=1}^{\infty}$. If sequences of variables $\{\Lambda_1^j\}_{j=1}^{\infty}$ and $\{\Lambda_2^j\}_{j=1}^{\infty}$ converge to a stationary point, i.e., $(\Lambda_{1+} - \Lambda_1) \to 0$ and $(\Lambda_{2+} - \Lambda_2) \to 0$, then $(D - PX) \to 0$ and $(\widehat{D} - D) \to 0$, respectively. This satisfies the first two conditions of the KKT conditions.

Second, from P_+ derived in the algorithm, we get

$$P_+ - P = (\Lambda_1 + \beta D)X^T(\lambda_1 I + \beta X X^T)^{-1} - P, \tag{E.2}$$

where I denotes an identity matrix and it can be rewritten by multiplying $(\lambda_1 I +$

βXX^T) to both sides in (E.2) as

$$\begin{aligned}(P_+ - P)&(\lambda_1 I + \beta XX^T) \\ &= (\Lambda_1 + \beta D)X^T - P(\lambda_1 I + \beta XX^T) \\ &= \Lambda_1 X^T - \lambda_1 P + \beta(D - PX)X^T.\end{aligned} \tag{E.3}$$

From the first condition, we can derive $\Lambda_1 X^T - \lambda_1 P \to 0$ when $(P_+ - P) \to 0$.

Third, using $X_+ = (\lambda_1 I + \beta P^T P)^{-1} P^T (\Lambda_1 + \beta D)$ derived from the algorithm, we can obtain the following:

$$\begin{aligned}(\lambda_1 I+&\beta P^T P)(X_+ - X) \\ &= P^T(\Lambda_1 + \beta D) - (\lambda_1 I + \beta P^T P)X \\ &= P^T \Lambda_1 - \lambda_1 X + \beta P^T (D - PX).\end{aligned} \tag{E.4}$$

If $(X_+ - X) \to 0$, then $(P^T \Lambda_1 - \lambda_1 X) \to 0$ as well.

Likewise, we can get the following equation using D_+ from the proposed algorithm,

$$\begin{aligned}(\lambda_2 + 2\beta)&(D_+ - D) \\ &= \beta(PX + \widehat{D}) - \Lambda_1 + \Lambda_2 - \lambda_2 D - 2\beta D \\ &= \beta(PX - D + \widehat{D} - D) - \Lambda_1 + \Lambda_2 - \lambda_2 D.\end{aligned} \tag{E.5}$$

Since $PX - D$ and $\widehat{D} - D$ converge to zero from the previous analysis, we obtain $\Lambda_1 - \Lambda_2 + \lambda_2 D = 0$ whenever $D_+ - D \to 0$.

Lastly, from (5.24), we obtain the following equation:

$$\widehat{D}_+ - \widehat{D} = Y - \mathcal{S}\left(Y - D + \frac{\Lambda_2}{\beta}, \beta\right) - D. \tag{E.6}$$

Since $\{G^j\}_{j=1}^{\infty}$ is bounded by our assumption, $\{X_+ X_+^T\}_{j=1}^{\infty}$ and $\{P_+^T P_+\}_{j=1}^{\infty}$ in (E.3) and (E.5) are bounded. Hence, $\lim_{j\to\infty}(G^{j+1} - G^j) = 0$ implies that both side of the above equations (E.3), (E.4), (E.5), and (E.6) tend to zero as $j \to \infty$.

Therefore, the sequence $\{G^j\}_{j=1}^{\infty}$ asymptotically satisfies the KKT condition for (5.21):

$$\begin{aligned} D - PX \to 0, \quad \widehat{D} - D \to 0, \quad \lambda_1 P - \Lambda_1 X^T \to 0, \\ \lambda_1 X - P^T \Lambda_1 \to 0, \quad \lambda_2 D + \Lambda_1 - \Lambda_2 \to 0, \\ Y - \widehat{D} - \mathcal{S}\left(Y - D + \frac{\Lambda_2}{\beta}, \frac{1}{\beta}\right) \to 0. \end{aligned} \tag{E.7}$$

This completes the proof. $\square$

Appendix E. Proof of Theorem 2

Appendix F

Proof of Theorems in Chapter 6

F.1 Proof of Theorem 3

Theorem 3. *Suppose that the data sampling is sufficient and samples are drawn from a union of k independent linear subspaces. Let us define a function f satisfying $f(Z) = f(ZP)$, for any permutation matrix P. Then, the optimal solution $Z^* \in \mathbb{R}^{n\times n}$ to the problem (6.1) is block-diagonal.*

Proof. The proof is analogous to that of Theorem 2 in [111]. Nonetheless, we give the proof for the sake of completion of Theorem 3. Assume that samples are in general position, i.e., $X = [X_1, ..., X_k] \in \mathbb{R}^{d\times n}$. Let $Z^* \in \mathbb{R}^{n\times n}$ be an optimal to the problem (6.1) or (6.14) and let $Z^B \in \mathbb{R}^{n\times n}$ be a block-diagonal matrix, whose (i, j)-th element has a value of Z^*_{ij} if x_i and x_j lie in the same subspace, otherwise 0. Let us define an off-block-diagonal matrix $Z^O = Z^* - Z^B \in \mathbb{R}^{n\times n}$.

Now, suppose that $[X]_j = [XZ^*]_j \in \mathcal{S}_l$ where $[A]_j$ is the j-th column of A. Then, we have $[XZ^B]_j \in \mathcal{S}_l$ and $[XZ^O]_j \in \oplus_{i\neq l}\mathcal{S}_i$, where $\oplus$ is the direct sum. But,

$[XZ^*]_j - [XZ^B]_j = [XZ^O]_j \in \mathcal{S}_l$. Hence, $[XZ^O]_j = 0$ because of the independent assumption among the subspaces. Thus, Z^B is a feasible solution to (6.1) and (6.14). Then, we use Lemma 3.1 in [54], which has the following relation:

$$\|Z^*\|_* = \left\| \begin{matrix} A & B \\ C & D \end{matrix} \right\|_* \geq \left\| \begin{matrix} A & 0 \\ 0 & D \end{matrix} \right\|_* = \|Z^B\|_*,$$

for any matrices B and C with compatible dimension, and this relation can also apply other functions, such as $\|Z\|_1$ and $\|Z\|_F$. Since Z^* is the optimal, i.e., $\|Z^*\|_* \leq \|Z^B\|_*$, we have $\|Z^*\|_* = \|Z^B\|_*$ meaning that Z^* is block-diagonal. Likewise, we have $\sum_i \lambda_i f_i(Z^*) = \sum_i \lambda_i f_i(Z^B)$, where f_i can be a norm in (6.1) or (6.14) and $\lambda_i > 0$. □

F.2 Proof of Theorem 4

Theorem 4. *Given a sample $\boldsymbol{x}_k \in \mathbb{R}^d$, a dataset $X \in \mathbb{R}^{d\times n}$, and parameters (λ_1, λ_2), and assume that X is normalized. Let $\boldsymbol{z}^* \in \mathbb{R}^n$ be the optimal solution to following problem:*

$$\min_{\boldsymbol{z}} \frac{1}{2}\|\boldsymbol{x}_k - X\boldsymbol{z}\|_2^2 + \lambda_1\|\boldsymbol{z}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{z}\|^2, \tag{F.1}$$

where $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_{k-1}, \boldsymbol{x}_{k+1}, ..., \boldsymbol{x}_{n+1}]$. Supposed that $z_i z_j > 0$, we have the following relation:

$$\mu(z_i^*, z_j^*) \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)}, \tag{F.2}$$

where $\mu(z_i^*, z_j^*) = \|z_i^* - z_j^*\|_2 / \|\boldsymbol{x}_k\|_2$ and $\rho = \boldsymbol{x}_i^T\boldsymbol{x}_j$ is the sample correlation.

The proof is based on Theorem 1 in [104]. Note that a similar result was reported in [111], in which the l_1-norm regularizer was absent. Nonetheless, we provide the proof for the sake of completeness. The problem considered in Theorem 2 is as follows:

$$\min_{\boldsymbol{z}} \frac{1}{2}\|\boldsymbol{x}_k - X\boldsymbol{z}\|_2^2 + \lambda_1\|\boldsymbol{z}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{z}\|^2. \tag{F.3}$$

Proof. We first take a derivative of (F.3) with respect to z_i and z_j, respectively, and replace $\boldsymbol{z}$ as $\boldsymbol{z}^*$, then we have

$$-\boldsymbol{x}_i^T(\boldsymbol{x}_k - X\boldsymbol{z}^*) + \lambda_1 \operatorname{sgn}(z_i^*) + \lambda_2 z_i^* = 0, \tag{F.4}$$

$$-\boldsymbol{x}_j^T(\boldsymbol{x}_k - X\boldsymbol{z}^*) + \lambda_1 \operatorname{sgn}(z_j^*) + \lambda_2 z_j^* = 0. \tag{F.5}$$

By subtracting (F.4) from (F.5), we have

$$z_i^* - z_j^* = \frac{1}{\lambda_2}(\boldsymbol{x}_i^T - \boldsymbol{x}_j^T)(\boldsymbol{x}_k - X\boldsymbol{z}^*) + c, \tag{F.6}$$

where $c = \alpha(\operatorname{sgn}(z_i^*) - \operatorname{sgn}(z_j^*))$ and α is a constant value. Since we assumed that $z_i z_j > 0$, it gives $\operatorname{sgn}(z_i^*) = \operatorname{sgn}(z_j^*)$. Hence, the constant c in (F.6) disappears. Since X is normalized, $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = 2(1 - \boldsymbol{x}_i^T \boldsymbol{x}_j)$. Finally, we have the following relation:

$$\begin{aligned}
\|z_i^* - z_j^*\|_2 &= \frac{1}{\lambda_2}\|\boldsymbol{x}_i^T - \boldsymbol{x}_j^T\|_2 \|\boldsymbol{x}_k - X\boldsymbol{z}^*\|_2, \\
&= \frac{1}{\lambda_2}\sqrt{2(1-\rho)} \cdot \|\boldsymbol{x}_k - X\boldsymbol{z}^*\|_2, \\
&\le \frac{1}{\lambda_2}\sqrt{2(1-\rho)} \cdot \|\boldsymbol{x}_k\|_2,
\end{aligned} \tag{F.7}$$

Therefore, we have $\|z_i^* - z_j^*\|_2 \le \frac{1}{\lambda_2}\sqrt{2(1-\rho)} \cdot \|\boldsymbol{x}_k\|_2$, where $\rho = \boldsymbol{x}_i^T \boldsymbol{x}_j$. In a case where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are negatively correlated, we can consider $-\boldsymbol{x}_j$, then $\|z_i^* - z_j^*\|_2 \le \frac{1}{\lambda_2}\sqrt{2(1+\rho)} \cdot \|\boldsymbol{x}_k\|_2$, where $\rho = -\boldsymbol{x}_i^T \boldsymbol{x}_j$. □

F.3 Proof of Theorem 5

Theorem 5. *The optimal solution of GLR has grouping effect, i.e., given a set of data samples* $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$ *and a subspace representation matrix* $Z \in \mathbb{R}^{n \times n}$*, a solution to the optimization problem of GLR using* X*, if* $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \to 0$*, then* $\|\boldsymbol{z}_i - \boldsymbol{z}_j\| \to 0$ *for all* $i \neq j$*.*

Before proving Theorem 3, we need to know the following enforced grouping effect (EGE) conditions [59]. Here, we reduce the conditions by focusing on the GLR problems.

Definition 4 (Enforced Grouping Effect conditions [59]). *The enforced grouping effect (EGE) conditions are as follows:*

(1) $f(Z) = \|Z\|_*$ *is continuous with respect to* Z.

(2) The following problem has a unique solution Z^*.

$$\min_{Z} \frac{1}{2}\|X - XZ\|_F^2 + f(Z). \tag{F.8}$$

(3) $f(Z) = f(ZP)$, *for all permutation matrix* P.

Proof. From Proposition 1 in [59], if GLR satisfies all the EGE conditions in Definition 4, the optimal solution Z^* to the problem of GLR has grouping effect. It is obvious that EGE conditions (1) and (3) are satisfied for GLR. Now, we need to show that the uniqueness of the solution of GLR, where $f(Z) = \lambda_1\|Z\|_* + \frac{\lambda_2}{2}\|Z\|_F^2$. Due to the Frobenius norm regularizer, the GLR problem is strong convex for Z [104]. If $\lambda_2 = 0$, it is reduced to the LRR problem [4]. Although the LRR problem is not strong convex, the unique optimal solution of LRR was proved in [59]. Hence, the problem of GLR has always a unique solution except the case when $\lambda_1 = \lambda_2 = 0$, which is not a subspace clustering problem. This means that GLR has the grouping effect. □

Appendix G

Proof of Theorems in Chapter 7

G.1 Proof of Theorem 6

Theorem 6. *Suppose that noiseless data samples are sufficiently collected from a union of k independent linear subspaces and basis vectors constructing the summary matrix cover the remaining samples. Let us define a function f which satisfies $f(C) = f(C\mathcal{P})$ for any permutation matrix $\mathcal{P}$. Then, the problem (7.1) based on the summary representation solves the subspace clustering problem exactly with a block-diagonal structure of $\widetilde{C}$.*

Proof. The block-diagonal structure of ILSR described in Section 7.1 for a noiseless case can be proved straight-forwardly since ILSR has an equivalent solution to the following LSR problem [111]:

$$\min_{C} \|C\|_F, \quad s.t. \;\; X = XC, \tag{G.1}$$

whose block-diagonal structure was proved in [111]. Likewise, the block-diagonal structure of the summary matrix C^S with k block matrices can be easily proved by reducing the ILSR problem to a problem with a subset S of dataset X used in ILSR. Since we assumed that remaining samples can be represented by basis vectors of the summary matrix, C^R also has a block-diagonal structure with k block matrices. Specifically, the rule of the summary representation is to collect samples having low correlation with other samples to enlarge the diversity of a summary matrix. Hence, if our basis vectors cover the true basis vectors representing a subspace, we can represent the remaining samples. Suppose we can permutate an aggregation matrix C^{Agg} which consists of C^S and C^R. Then, the aggregation matrix contains k nonzero block matrices. Since the Nyström-type reconstruction involves a multiplication of three block matrices with matching nonzero blocks, the final affinity matrix has the block-diagonal structure. □

G.2 Proof of Theorem 7

Theorem 7. *Suppose that $\widetilde{C} = \tilde{U}\tilde{U}^T \in \mathbb{R}^{n\times n}$ with a matrix $\tilde{U} \in \mathbb{R}^{n\times r}$. Then, for a matrix $\tilde{V} \in \mathbb{R}^{n\times r^2}$ satisfying $\widetilde{C}^{pp} = ([\tilde{U}\tilde{U}^T]_{ij})^2 = \widetilde{C} \odot \widetilde{C} = \tilde{V}\tilde{V}^T$, where $\odot$ is the Hadamard product, the following holds:*

$$\tilde{V} = [\tilde{U}_1 \otimes \tilde{U}_1; \tilde{U}_2 \otimes \tilde{U}_2; ...; \tilde{U}_r \otimes \tilde{U}_r], \tag{G.2}$$

where $\tilde{U}_i$ is the i-th row of $\tilde{U}$ and $\otimes$ is the Kronecker product.

Proof. Let, $M = \text{diag}(vec(I_n))$ where I_n is the $n \times n$ identity matrix and $\text{diag}(\cdot)$ and $vec(\cdot)$ are the diagonal and vectorization operators, respectively. Then, we have the following relation:

$$\begin{aligned}\widetilde{C} \odot \widetilde{C} &= \hat{M}^T((\tilde{U}\tilde{U}^T) \otimes (\tilde{U}\tilde{U}^T))\hat{M} \\ &= \hat{M}^T(\tilde{U} \otimes \tilde{U})(\tilde{U} \otimes \tilde{U})^T\hat{M} = \tilde{V}\tilde{V}^T,\end{aligned} \tag{G.3}$$

where $\tilde{V} = [(\tilde{U}_1 \otimes \tilde{U}_1)^T \ (\tilde{U}_2 \otimes \tilde{U}_2)^T \ \cdots \ (\tilde{U}_n \otimes \tilde{U}_n)^T]^T \in \mathbb{R}^{n \times r^2}$ and $\hat{M} = [M_i]_{i \in \mathcal{H}} \in \mathbb{R}^{n^2 \times n}$, where $\mathcal{H} = \{k : \sum_j M_{jk} \neq 0\}$ and M_i is an i-th column vector of M, is a matrix constructed by stacking n column vectors of M. $\square$

Appendix G. Proof of Theorems in Chapter 7

Bibliography

[1] N. Srebro, "Weighted low-rank approximations," in *Proc. of the International Conference on Machine Learning*, 2003.

[2] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Advances in Neural Information Processing Systems*, 2007.

[3] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[4] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, 2013.

[5] J. Wright, Y. Allen Y, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[6] J. Wright, Y. Ma, J. Mairal, G. sapiro, T. S. Huang, and S. Yan, "Sparse representation for computer vision and pattern recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.

[7] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[8] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, 2006.

[9] I. T. Jolliffe, *Principal Component Analysis.* John Wiley and Sons, 1986.

[10] Q. Ke and T. Kanade, "Robust l_1 norm factorization in the presence of outliers and missing data by alternative convex programming," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[11] Y. Shen, Z. Wen, and Y. Zhang, "Augmented Lagrangian alternating direction method for matrix separation based on low-rank factorization," *Optimization Methods and Software*, vol. 29, no. 2, pp. 239–263, 2014.

[12] X. Shu, F. Porikli, and N. Ahuja, "Robust orthonormal subspace learning: Efficient recovery of corrupted low-rank matrices," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[13] L. Torresani, A. Hertzmann, and C. Bregler, "Learning non-rigid 3D shape from 2D motion," in *Advances in Neural Information Processing Systems*, 2003.

[14] L. Wu, A. Ganesh, B. Shi, Y. Matsushita, T. Wang, and Y. Ma, "Robust photometric stereo via low-rank matrix completion and recovery," in *Proc. of the Asian Conference on Computer Vision*, 2010.

[15] R. Vidal, "Subspace clustering," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 52–68, 2011.

[16] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2765–2780, 2013.

[17] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, "Multi-task low-rank affinity pursuit for image segmentation," in *Proc. of the International Conference on Computer Vision*, 2011.

[18] D. Park, C. Caramanis, and S. Sanghavi, "Greedy subspace clustering," in *Advances in Neural Information Processing Systems*, 2014.

[19] E. Kim, M. Lee, and S. Oh, "Elastic-net regularization of singular values for robust subspace learning," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[20] N. Kwak, "Principal component analysis based on L_1-norm maximization," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 3, no. 9, pp. 1672–1680, 2008.

[21] E. Kim, M. Lee, C.-H. Choi, N. Kwak, and S. Oh, "Efficient l_1-norm-based low-rank matrix approximations for large-scale problems using alternating rectified gradient method," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 26, no. 2, pp. 237–251, 2015.

[22] R. Cabral, F. D. la Torre, J. P. Costeira, and A. Bernardino, "Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition," in *Proc. of the IEEE International Conference Computer Vision*, 2013.

[23] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[24] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *Proc. of the Asilomar Conference on Signals, Systems and Computers*, 1993.

[25] A. Eriksson and A. Hengel, "Efficient computation of robust weighted low-rank matrix approximations using the l_1 norm," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1681–1690, 2012.

[26] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Trans. on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[27] I. E. Frank and F. Jerome H, "A statistical view of some chemometrics regression tools," *Technometrics*, vol. 35, no. 2, pp. 109–135, 1993.

[28] J. Fan and R. Li, "Variable selection via nonconcave penalized likelihood and its oracle propoerties," *Journal of the American Statistical Association*, vol. 96, no. 456, pp. 1348–1360, 2001.

[29] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 877–905, 2008.

[30] H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed l^0 norm," *IEEE Trans. on Signal Processing*, vol. 57, no. 1, pp. 289–301, 2009.

[31] D. Wipf and S. Nagarajan, "Iterative reweighted l_1 and l_2 methods for finding sparse solutions," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 317–329, 2010.

[32] C.-H. Zhang, "Nearly unbiased variable selection under minimax concave penalty," *The Annals of Statistics*, vol. 38, no. 2, pp. 894–942, 2010.

[33] P. Gong, C. Zhang, Z. Lu, Z. Z. Huang, and J. Ye, "A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems," in *Proc. of the International Conference on Machine Learning*, 2013.

[34] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1994.

[35] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, pp. 11:1–11:37, 2011.

[36] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[37] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[38] A. Argyriou, R. Foygel, and N. Srebro, "Sparse prediction with the k-support norm," in *Advances in Neural Information Processing Systems*, 2003.

[39] D. Geman and C. Yang, "Nonlinear image recovery with half-quadratic regularization," *IEEE Trans. on Image Processing*, vol. 4, no. 7, pp. 932–946, 1995.

[40] T. Zhang, "Multi-stage convex relaxation for learning with sparse regularization," in *Advances in Neural Information Processing Systems*, 2009.

[41] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems," *Journal of Optimization*, vol. 6, no. 3, pp. 615–640, 2010.

[42] Z. Zhang, K. Zhao, and H. Zha, "Inducible regularization for low-rank matrix factorizations for collaborative filtering," *Neurocomputing*, vol. 97, pp. 52–62, 2012.

[43] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices," *Mathematical Programming*, 2010.

[44] P. S. E. Richard and N. Vayatis, "Estimation of simultaneously sparse and low rank matrices," in *Proc. of the International Conference on Machine Learning*, 2012.

[45] Y. Liu, L. C. Jiao, and F. Shang, "A fast tri-factorization method for low-rank matrix recovery and completion," *Pattern Recognition*, vol. 46, pp. 163–173, 2013.

[46] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: factorization method," *International Journal of Computer Vision*, vol. 9, no. 2, pp. 137–154, 1992.

[47] P. F. U. Gotardo and A. M. Martinez, "Non-rigid structure from motion with complementary rank-3 spaces," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[48] T. Zhou and D. Tao, "Godec: Randomized low-rank & sparse matrix decomposition in noisy case," in *Proc. of the International Conference on Machine Learning*, 2011.

[49] Y. Zheng, G. Liu, S. Sugimoto, S. Yan, and M. Okutomi, "Practical low-rank matrix approximation under robust l_1-norm," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[50] G. Liu and S. Yan, "Active subspace: Toward scalable low-rank learning," *Neural Computation*, vol. 24, pp. 3371–3394, 2012.

[51] B.-K. Bao, G. Liu, C. Xu, and S. Yan, "Inductive robust principal component analysis," *IEEE Trans. on Image Processing*, vol. 21, pp. 3794–3800, 2012.

[52] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[53] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[54] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proc. of the International Conference on Machine Learning*, 2010.

[55] R. Tron and R. Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[56] C.-Y. Lu, J. Feng, Z. Lin, and S. Yan, “Correlation adaptive subspace segmentation by trace lasso,” in *Proc. of the International Conference on Computer Vision*, 2013.

[57] P. Ji, M. Salzmann, and H. Li, “Shape interaction matrix revisied and robustified: Efficient subspace clustering with corrupted and incomplete data,” in *Proc. of the International Conference on Computer Vision*, 2015.

[58] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems*, 2001.

[59] H. Hu, Z. Lin, J. Feng, and J. Zhou, “Smooth representation clustering,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[60] X. Zhang, F. Sun, G. Liu, and Y. Ma, “Fast low-rank subspace segmentation,” *IEEE Trans. on Knowledge and Data Engineering*, vol. 26, no. 5, pp. 1293–1297, 2014.

[61] S. Xiao, W. Li, D. Xu, and D. Tao, “FaLRR: A fast low rank representation solver,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[62] A. Talwalkar, L. Mackey, Y. Mu, S.-F. Chang, and M. I. Jordan, “Distributed low-rank subspace segmentation,” in *Proc. of the International Conference on Computer Vision*, 2013.

[63] X. Peng, L. Zhang, and Z. Yi, “Scalable sparse subspace clustering,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[64] E. L. Dyer, A. C. Sankaranarayanan, and R. G. Baraniuk, "Greedy feature selection for subspace clustering," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2487–2517, 2013.

[65] C. You, D. P. Robinson, and R. Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[66] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning.* MIT Press, 2006.

[67] S. Choi, E. Kim, and S. Oh, "Real-time navigation in crowded dynamic environments using Gaussian process motion control," in *Proc. of IEEE International Conference on Robotics and Automation*, 2014.

[68] R. Mazumder, J. Friedman, and T. Hastie, "SparseNet: Coordinate descent with nonconvex penalties," *Journal of the American Statistical Association*, vol. 106, no. 495, pp. 1125–1138, 2011.

[69] C. Lu, J. Tang, S. Yan, and Z. Lin, "Generalized nonconvex nonsmooth low-rank minimization," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[70] F. H. Clarke, "Generalized gradients and applications," *Transactions of the American Mathematical Society*, vol. 205, pp. 247–262, 1975.

[71] H. Li and Z. Lin, "Accelerated proximal gradient methods for nonconvex programming," in *Advances in Neural Information Processing Systems*, 2015.

[72] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *arXiv:1511.06324*, 2015.

[73] P.-L. Loh and M. J. Wainwright, "Regularized M-estimators with nonconvexity: Statistical and algorithmic theory for local optima," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 559–616, 2015.

[74] A. M. Buchanan and A. W. Fitzgibbon, "Damped Newton algorithms for matrix factorization with missing data," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[75] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proc. of the International Conference on Machine Learning*, 2010.

[76] E. L. Dyer and A. C. Sankaranarayanan, "Greedy feature selection for subspace clustering," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2487–2517, 2013.

[77] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

[78] K.-C. Lee, J. Ho, and D. Kriegman, "Acquiring linear subspaces for face recognition under variable llighting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, 2005.

[79] C. A. R. Hoare, "Algorithm 65: find," *Communications of the ACM*, vol. 4, no. 7, pp. 321–322, July 1961.

[80] A. Rauh and G. R. Arce, "A fast weighted median algorithm based on quickselect," in *Proc. of the IEEE International Conference on Image Processing*, Sep. 2010.

[81] K.-C. Toh and S. Yun, "An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems," *Journal of Optimization*, vol. 6, no. 3, pp. 615–640, 2010.

[82] W. I. Zangwill, *Nonlinear Programming: a Unified Approach.* Prentice-Hall, 1969.

[83] D. G. Luenberger, *Linear and Nonlinear Programming.* Springer, 2010.

[84] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," in *Proc. of the IEEE Conference Automatic Face and Gesture Recognition*, Sep. 2008.

[85] E. Kim and S. Oh, "Robust orthogonal matrix factorization for efficient subspace learning," *Neurocomputing*, vol. 167, pp. 218–229, 2015.

[86] C. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[87] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *Journal of Machine Learning Research*, vol. 11, pp. 2287–2322, 2010.

[88] G. Golub and C. V. Loan, *Matrix Computations*, 3rd ed. Johns Hopkins University Press, 1996.

[89] "PETS 2009 dataset," http://www.cvg.rdg.ac.uk/PETS2009.

[90] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proc. of IEEE International Conference on Computer Vision*, 1999.

[91] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, pp. 2756–2779, 2007.

[92] C. Xhao, X. Wang, and W.-K. Cham, "Background subtraction via robust dictionary learning," *EURASIP Journal on Image and Video Processing*, pp. 1–12, 2011.

[93] E. Kim, S. Choi, and S. Oh, "Structured low-rank matrix approximation in Gaussian process regression for autonomous robot navigation," in *Proc. of IEEE International Conference on Robotics and Automation*, 2015.

[94] F. Yan and Y. Qi, "Sparse Gaussian process regression via l_1 penalization," in *Proc. of the International Conference on Machine Learning*, 2010.

[95] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*, 2001.

[96] B. Scholkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[97] E. Kim, S. Choi, and S. Oh, "A robust autoregressive Gaussian process motion model using l_1-norm based low-rank kernel matrix approximation," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[98] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Review*, vol. 52, pp. 471–501, 2010.

[99] Y. Ni, J. Sun, X. Yuan, S. Yan, and L.-F. Cheong, "Robust low-rank subspace segmentation with semidefinite guarantees," in *Proc. of the IEEE International Conference on Data Mining Workshops*, 2010, pp. 1179–1188.

[100] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a non-linear successive over-relaxation algorithm," *Rice University CAAM Technical Report TR10-07*, 2010.

[101] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems*, 2005.

[102] J. Quinonero-candela, C. E. Rasmussen, and R. Herbrich, "A unifying view of sparse approximate Gaussian process regression," *Journal of Machine Learning Research*, vol. 6, pp. 1939–1959, 2005.

[103] E. Kim, M. Lee, and S. Oh, "Robust elastic-net subspace representation," *IEEE Trans. on Image Processing*, vol. 25, no. 9, pp. 4245–4259, 2016.

[104] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society, Series B*, vol. 67, pp. 301–320, 2005.

[105] H. Li, N. Chen, and L. Li, "Error analysis for matrix elastic-net regularization algorithms," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 23, no. 5, pp. 737–748, 2012.

[106] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick, "Large-scale image classification with trace-norm regularization," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[107] P. Favaro, R. Vidal, and A. Ravichandran, "A closed form solution to robust subspace estimation and clustering," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[108] J.-F. Cai, E. J. Candes, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[109] T. Sun and C.-H. Zhang, "Calibrated elastic regularization in matrix completion," in *Advances in Neural Information Processing Systems*, 2012.

[110] S. P. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[111] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, and S. Yan, "Robust and efficient subspace segmentation via least squares regression," in *Proc. of the European Conference on Computer Vision*, 2012.

[112] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition workshop*, 2004.

[113] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[114] C. Li, L. Lin, W. Zuo, S. Yan, and J. Tang, "Sold: Sub-optimal low-rank decomposition for efficient video segmentation," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[115] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, 2011.

[116] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *Proc. of the International Conference on Computer Vision*, 2011.

[117] S. X. Yu and J. Shi, "Multiclass spectral clustering," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[118] M. Lee, J. Lee, H. Lee, and N. Kwak, "Membership representation for detecting block-diagonal structure in low-rank or sparse subspace clustering," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[119] J. Feng, Z. Lin, H. Xu, and S. Yan, "Robust subspace segmentation with block-diagonal prior," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[120] C.-G. Li and R. Vidal, "Structured sparse subspace clustering: A unified optimization framework," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[121] C. K. I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*, 2001.

[122] S. Kumar, M. Mohri, and A. Talwalkar, "On sampling-based approximate spectral decomposition," in *Proc. of the IEEE International Conference Machine Learning*, 2009.

[123] L. Zhang, M. Yang, and X. Feng, "Sparse representation or collaborative representation: which helps face recognition?," in *Proc. of the IEEE International Conference Computer Vision*, 2011.

[124] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2013.

[125] V. M. Patel and R. Vidal, "Kernel sparse subspace clustering," in *Proc. of the IEEE International Conference on Image Processing*, 2014.

[126] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, "Spectral grouping using the Nyström method," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 2, pp. 214–225, 2004.

[127] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 568–586, 2011.

[128] X. Chen and D. Cai, "Large scale spectral clustering with landmark-based representation," in *Proc. of the Association for the Advancement of Artificial Intelligence*, 2011.

초 록

저차원의 구조를 희소 (sparse) 또는 저계수 (low-rank) 표현을 기반으로 학습하는 방법은 최근 많은 주목을 받아왔으며, 다양한 분야에서 널리 사용되고 있다. 그 중 희소 표현은 이미지나 동영상과 같은 규모가 큰 데이터를 적은 수의 대표적인 샘플들의 조합으로 표현 또는 압축시키는 것을 목표로 하며, 이러한 접근의 2차원 확장이 저계수 표현법이다. 앞선 표현법들의 성공적인 적용의 이면에는 희소 및 저계수 표현을 효과적으로 학습하기 위한 많은 노력 및 연구들이 있었다. 하지만, 현실적인 문제에서 많은 데이터들을 다루거나 아웃라이어나 미싱 (missing) 데이터와 같은 원치 않는 노이즈들이 있는 상황에서 앞서 언급한 방법들은 여전히 효과적이지 못한 단점이 있다. 또한, 최근 연구들은 노이즈에 강인한 방법을 제안하기는 하지만, 많은 계산 복잡도를 요구하게 되어 현실적인 사용에 제한이 되기도 한다. 따라서 본 논문에서는 노이즈가 있는 상황에서 강인한 표현을 하면서 계산의 복잡도에 있어서도 많은 이점이 있는 데이터 표현 방법들을 제안하는 것을 목표로 한다.

우선 희소 표현에 대해서는 대부분의 알고리즘들이 오리지널 문제인 l_0-norm 기반의 문제를 풀기가 어렵기 때문에 이를 convex한 l_1-norm으로 근사하여 문제를 풀게 된다. 하지만, 시스템 자체가 nonconvex한 문제들에 대해서는 convex l_1-norm은 효과적이지 못한 선택이 될 수 있기 때문에, 이러한 l_0-norm과 l_1-norm의 장점을 모두 가질 수 있는 새로운 measure를 제안하며, 이는 gradient가 천천히 없어지는 형태를 가지기 때문에 최적화 관점에서도 매우 적절하다.

저계수 표현에 대해서는 노이즈에 강인한 학습을 위해서 l_1-norm 기반의 목적함수를 디자인 할 수 있으며, 이는 기존의 방법들에서는 효과적인 학습이 되지 않았기 때문에 빠른 학습을 위해 gradient 기반의 알고리즘을 본 논문에서 제안하였고, gradient의 방향이 최대한 빨리 최적 해에 도달할 수 있도록 학습하는 방법에 대해 연구를 진행하였다. 이러한 문제를 조금 더 빠르고 안정적으로 학습하기 위해 gradient 기반이 아니라 최근의 최적화 문제에서 많은 발전을

이룬 augmented Lagrangian 방법을 이용하여 명백한 smoothness regularizer와 orthogonality 제약과 함께 더 효율적인 학습이 가능하게 하였다. 앞선 두 방법은 노이즈에 강인하기는 하지만 적은 수의 노이즈에 한정적이며, 문제 자체의 rank를 고정시켜 풀게 되어 현실적인 문제에 적합하지 않은 단점이 있다. 이를 개선하기 위해 elastic-net 기반으로 데이터의 singular 값을 적절히 교정하고 학습하여 유연한 rank의 예측이 가능하게 하였으며, 심각한 노이즈들이 들어왔을 때에도 효과적이고 안정적인 학습을 가능하게 하여, 최신 방법들에 비해 더욱 우수한 학습 결과를 얻을 수 있었다. 추가로, 저계수 표현을 유사 행렬과 같은 구조화된 데이터로 확장 또한 rank 최소화 방법을 기반으로 성공적인 연구를 수행하였다.

마지막으로, 앞서 언급한 저계수 표현법은 데이터가 하나의 저차원 (subspace)에 있는 경우를 가정하여 문제를 해결하는데, 여러 저차원의 조합으로 구성된 데이터를 다루기 위해서 본 논문에서는 저차원 분류 문제 또한 다루게 된다. 저차원 분류에서 가장 큰 문제는 점진적인 학습이 되지 않으며 계산 복잡도 또한 매우 크다는 것이다. 이러한 알고리즘 속도의 향상을 위해 본 논문에서는 매우 적은 계산량으로도 저계수 저차원 분류가 가능한 새로운 방법을 제안한다. 유사 행렬의 점진적인 학습이 가능함과 동시에 모든 저차원 분류 과정을 선형적인 복잡도에서 처리할 수 있는 방법을 제시함으로써 기존 방법들에 비해 매우 빠른 알고리즘 처리속도를 가지며, 분류 성능 또한 경쟁력 있는 결과를 얻었다.

앞선 세 가지 큰 문제들에 대해서 벤치마크 데이터 셋들과 실제 문제들을 중심으로 제안하는 방법들의 우수성을 확인하기 위한 실험들을 진행하였으며, 많은 실험 결과들을 통해 제안하는 방법들이 다른 최근에 제안된 방법들과 비교하여 상당히 강인하고, 효과적이며, 현실적으로 적용 가능한 처리속도를 얻을 수 있음을 검증하였다.

주요어: 희소 표현, 저계수 표현, 저차원 공간 학습, 저차원 분류, 행렬 분해, 컴퓨터 비전

학 번: 2013-30226