Technische Universität München
Fakultät für Elektrotechnik und Informationstechnik
Lehrstuhl für Datenverarbeitung

Learning Image and Video Representations Based on Sparsity Priors

Xian Wei

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technische Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzende/-r: Prof. Dr. Jörg Conradt

Prüfende/-r der Dissertation:

1. Prof. Dr.-Ing. Klaus Diepold
2. Priv.-Doz. Dr. Martin Kleinsteuber

Die Dissertation wurde am 17.11.2016 bei der Technische Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 07.04.2017 angenommen.

Xian Wei. *Learning Image and Video Representations Based on Sparsity Priors.* Dissertation, Technische Universität München, Munich, Germany, 2017.

Thanks to my family.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my supervisor, Dr. Martin Kleinsteuber, for his continuous support to my Ph.D. study. His deep insights and meticulous guidance helped me through all the time in research and writing of this dissertation. Especially, I want to thank him for his financial support in the last year.

My cordial thanks also go to Prof. Dr.-Ing. Klaus Diepold for his kind support to my doctoral degree program for providing me with facilities and personnels, in particular, for his generosity to take care of the administration of my dissertation examination.

I wish to express my gratitude to my mentor, Dr. Hao Shen, for his insightful comments and encouragement, especially, for his patience and expertise in teaching me research and scientific writing.

I would like to acknowledge the financial support from China Scholarship Council (CSC) for provision of scholarship. They gave me a financial support for four years of studies and researches. I am grateful to TUM graduate school for their support to my international visits and other academic activities.

I thank my colleagues from GOL and LDV. They are Dr. Michael Zwick, Dr. Simon Hawe, Clemens Hage, Martin Kiechle, Dominik Meyer, Matthias Seibert, Alexander Sagel, Julian Wörmann, Sunil Ramgopal Tatavarty, Peter Hausamann and Sundeep Patil. I am lucky to share the happy five years with them. My thanks are also due to Ms. Ricarda Baumhoer for her assistance and advices on administration through my Ph.D. study.

I would like to express my special thanks to Simon Hawe, Clemens Hage and Martin Kiechle, for their advices and software support for my Ph.D. research. My special thanks are also due to my colleagues in the same office, Martin Knopp, Alexander Sagel and Peter Hausamann, I am happy to work with them.

Abstract

Recent development in representation learning shows that appropriate data representations are the key to the success of machine learning algorithms, since different representations can entangle different explanatory information of the data. Among the various methods of learning representations, sparse representations of data have been observed to contain rich distributed information of the data with respect to specific learning tasks, such as image classification, regression, etc. By taking advantage of such a benefit, the focus of this dissertation is on developing algorithmic framework that allows disentangling the underlying explanatory factors hidden in sparse representations of image and video data. For example, explanatory information considered in this dissertation can be an underlying linear system that explains the dynamics of texture videos, or the similarity of image data points that explores the intrinsic structure of data. Moreover, such disentangled factors have shown to conveniently solve various computer vision problems. Specifically in this dissertation, they are dynamic texture modeling and low dimensional image representations. The key concept behind this development is to construct a joint cost function, which combines the criteria for learning sparse representations and the criteria for discovering underlying factors in the learned sparse representations. Since the admissible sets of solutions to our optimization problem are restricted on appropriate matrix manifolds, geometric optimization techniques that exploit the underlying manifold structures of solutions can be employed to efficiently solve such an optimization problem. Finally, we leverage the advantage of differential geometric optimization to develop a collection of efficient algorithms on appropriate differentiable manifolds.

The key difficulty for solving the proposed joint learning problem is the differentiability of sparse representation with respect to a given dictionary. For addressing such a challenge, we consider the sparse coding problem by minimizing a quadratic reconstruction error with appropriate convex sparsity priors, such as elastic net prior and Kullback-Leibler divergence prior. In this way, sparse representation can be shown to be a locally differentiable function with respect to a dictionary, and hence a generic form of the directional derivative of sparse representation with respect to the given dictionary is developed. The ability to compute such a derivative leads to various further learning mechanisms in sparse representations that disentangle different underlying explanatory factors. By leveraging such an algorithmic benefit and geometric optimization techniques, in what follows, we construct joint learning cost functions to study two aforementioned challenging computer vision problems, dynamic textures and image dimensionality reduction.

Modeling Dynamic Textures (DT) is a long standing active research topic in the computer vision community. Study and analysis of DT attracts both theoretical and practical research efforts, such as building a stable DT modeling system, video segmentation, video recognition and video synthesis. However, the continuous change in the shape and appearance of a dynamic texture makes the application of traditional computer vision algorithms very

challenging. Thus, finding an appropriate spatio-temporal generative representation model to explore the evolution of the dynamic textured scenes is the key to many DT studies. One classical technique is to model the dynamical course of DTs as a Markov random process. Following the Markov random process, one typical model is developed and widely applied to the practice, namely, linear dynamical system (LDS). LDS assumes that each observation is correlated to an underlying latent variable, or "state", and the dynamic process of these consecutive states can be captured by a parameter transition operator. In this dissertation, we follow the framework of classical LDS, and present to treat the sparse coefficients over a learned dictionary as the underlying "states". In this way, the dynamical process of dynamic textures exhibits a transition course of corresponding sparse events. Next, our goal is to find a suitable and robust linear transition matrix that captures the dynamics between two adjacent frames of sparse representations in time series. Under several reasonable assumptions, we read this transition as a linear transformation matrix with the constraint of stability. Under this way, a DT sequence is represented by an appropriate sparse transition matrix together with a dictionary, shortly called DT parameters. Such learned DT parameters can be used for various DT applications, such as DT synthesis, recognition and denoising.

The second computer vision problem studied in this dissertation is finding an appropriate low dimensional representations of raw images. It is known that natural images are often very high dimensional, statistically non-Gaussian, and show abundant varying texture patterns. Hence, they are difficult to be explicitly parameterized by a common probabilistic model. Therefore, some machine learning techniques, such as linear smooth regression, may not be directly used to construct the prediction model for such raw images. Finding appropriate low dimensional representations of image data is an efficient way to promote the further prediction models learning. In this dissertation, we present a unified algorithmic framework for learning low dimensional representations of images for the three classic machine learning scenarios of unsupervised, supervised and semi-supervised learning. The core concept of our development is to combine two popular data representation criteria, namely sparsity and trace quotient. The former is known to be a convenient tool to identify underlying factors, and the latter is known for disentangling underlying discriminative factors. We construct a generic cost function for learning jointly a sparsifying dictionary and a dimensionality reduction transformation. The proposed cost function covers a wide range of classic low dimensional representation methods, such as Principal Component Analysis, Local Linear Embedding, Laplacian Eigenmap, Linear Discriminant Analysis (LDA), Semi-supervised LDA, and more. Experimental evaluations on image classification, clustering, 2/3 D visualization, and object categorization demonstrate the strong competitive performance in comparison with state-of-the-art algorithms.

Contents

List of Symbols

$\mathbb{R}$	The set of real numbers.
$\mathbb{Z}$	The set of integers.
$\mathbb{R}^{m\times n}$	The set of real $m \times n$ matrices.
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Matrices, written as capital boldface letters.
$\mathbf{a}, \mathbf{b}, \mathbf{c}$	Column vectors, written as lowercase boldface letters.
A, b, c	Scalars, written as lowercase or capital letters.
$\mathbf{a}_i$	$i-$th column of matrix $\mathbf{A}$.
a_i	$i-$th element of vector $\mathbf{a}$.
$\mathbf{A}_{ij}$	Refer in particular, $i-$th element in the $j-$th column of matrix $\mathbf{A}$.
$\mathbf{e}_i$	The i-th standard basis vector of $\mathbb{R}^m$.
$\mathbf{E}_{ij}$	A matrix whose i^{th} entry in the j^{th} column is equal to one, and all others are zero.
Gr	Grassmann manifold.
$\mathcal{S}^{(m-1)}$	The $(m-1)$ dimensional unit sphere.
$\mathcal{S}(m,k)$	Product of k unit spheres $\mathcal{S}^{(m-1)}$.
$O(m)$	The orthogonal group.
$\mathfrak{so}(m)$	Lie algebra of real skew-symmetric $m \times m$ matrices.
$\boldsymbol{\mathcal{M}}$	Differentiable manifold.
$T_{\mathbf{x}}\boldsymbol{\mathcal{M}}$	The tangent space of manifold $\boldsymbol{\mathcal{M}}$ at point $\mathbf{x} \in \boldsymbol{\mathcal{M}}$.
$\mathbf{I}_m$	$m \times m$ identity matrix.
$\mathbf{H}_m$	$\mathbf{I}_m - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$ the centering matrix in $\mathbb{R}^m$.
$\mathbf{A}^\top$	Transpose of matrix $\mathbf{A}$.
$\mathrm{tr}(\mathbf{A})$	Trace of matrix $\mathbf{A}$.
$\mathrm{rk}(\mathbf{A})$	Rank of matrix $\mathbf{A}$.

$\det(\mathbf{A})$ Determinant of matrix $\mathbf{A}$.

$\operatorname{diag}(\cdot)$ Vector containing the diagonal entries of a square matrix.

$\operatorname{ddiag}(\mathbf{A})$ Diagonal matrix with the same diagonal entries as $\mathbf{A}$.

$\operatorname{sgn}(\cdot)$ Sign function.

$\log(\cdot)$ Natural logarithm.

$\mathcal{D}f(\mathbf{x})$ Derivative of map f at $\mathbf{x}$.

$\mathbf{D} \in \mathbb{R}^{m \times k}$ Dictionary matrix.

$\mathbf{U} \in \mathbb{R}^{k \times l}$ Projection matrix.

$\mathbf{x} \in \mathbb{R}^m$ Signal vector in $\mathbb{R}^m$.

$\boldsymbol{\phi} \in \mathbb{R}^k$ Sparse coefficient vector in $\mathbb{R}^k$.

$\mathbf{y} \in \mathbb{R}^l$ Vector of low-dimensional representation in $\mathbb{R}^l$.

$\boldsymbol{\epsilon} \in \mathbb{R}^m$ Noise vector in $\mathbb{R}^m$.

n Number of observations.

m The dimension of signal vector.

l The dimension of low-dimensional representation vector.

k Number of dictionary atoms.

$\|\cdot\|_F$ Frobenius norm.

$\|\cdot\|_2$ Euclidean norm.

$\|\cdot\|_1$ ℓ_1-norm of vectors and matrices.

$\|\cdot\|_p$ ℓ_p-pseudo-norm of vectors and matrices.

$\|\cdot\|_0$ ℓ_0-pseudo-norm of vectors and matrices.

$\|\cdot\|_\infty$ Infinity norm.

List of Figures

List of Tables

Chapter 1

Introduction

Machine learning algorithms attempt to discover the structure (e.g., patterns) in data and make accurate predictions for previously unseen data. From a probabilistic perspective, that often means discovering statistical dependencies between random variables. More generally, that means discovering where probability mass concentrates in the joint distribution of all the observations. However, it is known that the natural signals often have complex statistical structure with unknown distribution. Typical examples like natural images, they are photographed under changeable environment by cameras with various internal settings, thus, they are often difficult to be explicitly parameterized by a common probabilistic model. Another example is raw image sequences or videos, which show the continuous changes in the shape and appearance of dynamic scenes associated with varying illumination conditions, viewpoint of moving cameras or complex backgrounds. It is also not easy to infer the dynamic courses or extract visual information from such spatial and temporal changes occurring in an image sequence. Therefore, most machine learning techniques may not be directly used to construct the prediction model for these raw images or videos. Recent development in representation learning shows that finding appropriate representations of data plays a critical role as a preprocessing procedure for the success of modern machine learning algorithms, cf. [5]. It aims to disentangle suitable underlying information or factors of the data to facilitate the learning task of interest. For instance, the class information can be disentangled after the use of multiple layers of nonlinear transformations [6] or a set of kernel functions [7]. For that reason, many researchers focus on building preprocessing pipeline to support effective machine learning algorithms, i.e., finding appropriate representations of "raw" inputs to improve the performance of supervised or unsupervised tasks, such as classification and 2/3D visualization. The aim of this dissertation is to investigate effective data representation learning approaches to disentangle various explanatory or discriminative information in image data for solving computer vision problems.

1.1 The Role of Image Representation in Computer Vision

In the literature, image representation learning refers to the problem of extracting useful features from raw images or image sequences to feed a specific machine learning predictor, such as a linear classifier. For the reason that different representations can disentangle different explanatory factors of variation behind the data, the choice of representation has an enormous effect on the performance of such machine learning predictors. A *good* ab-

stract representation should carry explanatory factors that either describe the underlying internal structures of the raw data, or explain the target for a specific supervised task. Ultimately, such a *good* representation is expected to make further task learning easier. For the supervised learning, the factors are directly described by given targets. For example, in classification problem, the factors could be specified by observed class labels, which lead to training an explicit learner to separate these observed factors from the others. More generally, for the learning problem with large unlabeled data, the underlying internal structures of data can be described by a set of causal factors that generate the observed data. For instance, they can be the posterior distribution of the underlying explanatory factors of variation hidden in observed data, or the geometrical relationships between a data point and its adjacent points. Knowing specific underlying structures of data allows composing models of the considered signals, to disentangle the information of interest in representation space.

However, identifying and disentangling the underlying causal/explanatory factors that described the underlying structures of data needs prior knowledge. Such knowledge priors are conveniently built into the representation learner. In other words, the practical algorithms for learning underlying causal/explanatory factors are associated with some explicit or implicit knowledge priors. The latter are expected to provide clues or hints about the former. These general knowledge priors are not necessary task specific but help the representation learner discover, identify and disentangle the underlying factors hidden in data. As introduced in [8, 5, 6, 9, 10], examples of such general-purpose priors include *smoothness*, *linearity*, *depth of explanatory factors*, *manifold*, *linear subspace*, *natural clustering*, *semi-supervised learning*, *temporal and spatial coherence*, *sparsity*, *factor dependencies*, etc. Specifically, the performance of representation learning is strongly dependent on the choice and organization of various such general purpose priors.

This thesis focuses on developing representation learning methods for solving computer vision problems. Knowing that the knowledge about specific raw inputs, such as the images or the videos, can also be used to help design representations. Therefore, many research efforts are attracted on choosing appropriate aforementioned purpose priors as ways to help the representation learner discover some of the underlying priori unknown factors of specific raw inputs, cf. [5, 9]. By incorporating appropriate purpose priors into algorithms for solving specific problem in computer vision, in what follows, we review some typical examples of such techniques. These examples include low dimensional representation learning, sparse coding and deep learning.

In the past several decades, the increasingly larger volume of data challenges many computer vision algorithms. To overcome such a difficulty, methods on learning low dimensional representations are proposed to avoid the so called “curse of dimensionality”. It refers to various phenomena that arise when analyzing and organizing data in high-dimensional spaces, e.g., the natural images or videos (often with hundreds or thousands of dimensions). Note that, the "curse of dimensionality" is not a problem of high-dimensional data, it arises when the machine learning algorithm does not scale well to high-dimensional data, typically due to needing an amount of time or memory that is exponential in the number of dimensions of the data. On the other hand, the high dimensional data can contain high degree of irrele-

vant and redundant information which may greatly degrade the performance of some specific prediction models. Fig. 1.1 shows that the performance of a classifier decreases when the dimensionality of the problem becomes too large.

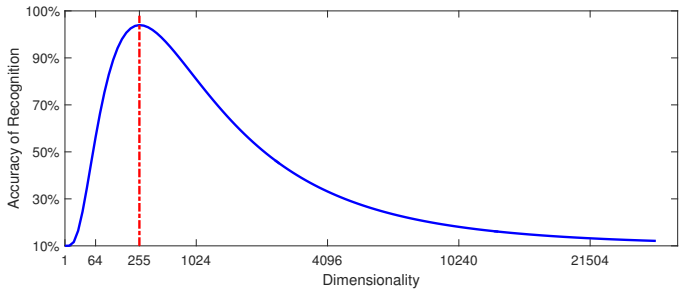


Figure 1.1: Using marginal fisher criterion to classify a large scale image dataset. As the dimensionality increases, the classifier's performance increases until the optimal number (255 dimension) of features is reached. Further increasing the dimensionality without decreasing the number of training samples results in a decrease in classifier performance.

When facing the curse of dimensionality, a good solution can be found by developing the suitable functions to map the data into a lower-dimensional form without significant information loss. The understanding of such mapping often resorts to the biological and geometric interpretation. Massive high-dimensional vision information (e.g., images and videos) about the natural world is captured every second by generic sensors (e.g., eyes or cameras). This information is often interrelated and largely redundant in two main aspects: First, it often contains multiple correlated versions of the same physical world and each version is usually sampled by widely distributed generic sensors. Compared to such large volume of recorded data sets, the relevant information about the underlying processes that cause our observations is generally of much reduced dimensionality. The extraction of this relevant information by identifying the cause factors within observed signals provides a good way to understand vision data. Secondly, each pixel of image corresponds to a potentially independent light intensity measurement, these intensities show strong spatial correlations in natural images [11]. Because of such structures, the variations of possible natural images can be described by only a few underlying factors, which includes various levels of brightness, resolutions, sharpness, camera orientations, etc. Typical example like human action videos, such variations could be arbitrary illuminations, poses, viewpoints, background clutters, and occlusion. It could conclude that the intrinsic dimension of natural images is much lower than the ambient dimension, thus natural images are concentrated around low-dimensional form, such as "manifold", cf. [12, 13, 5]. The objective of low dimensional representations is to reduce such irrelevance and redundancy of the high-dimensional image data in order to be able to store or transmit data in an efficient form.

Under the general purpose priors that the high dimensional images lie on a low-dimensional *smooth manifold* or subspaces, various methods have been proposed to find a suitable low dimensional space that high dimensional ambient space embedded in. They seek to convey underlying structure of interest (e.g., global/local geometry of data) from original high-

dimensional space to low dimensional ones. The related learning process relies on the decomposition of data relationship matrix over an *undercomplete* bases set, called *undercomplete* dictionary. The different choices of bases in such a dictionary can be interpreted as underlying explanatory factors that can describe the underlying data structures of interest. Typical data structures include covariance, dynamical structure, correlation between data sets, input-output relationships, and margin between data classes, cf. [14, 15]. For example, as a popular feature extraction algorithm, Principal Components Analysis (PCA) [16] is to find the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. These directions are orthogonal and their collection form an orthogonal dictionary. As a set of underlying factors, these orthogonal directions are expected to disentangle the uncorrelated structure of the data, which is measured by the proportion of variance explained by principal eigenvectors of the data covariance matrix. Changes in the directions of the principal components are perfectly captured by the PCA, whereas changes in the orthogonal directions are completely lost. Thus, a set of uncorrelated features are efficiently disentangled by cutting the low-variance directions out. Similarly, another example is known as Independent Component Analysis (ICA) [17], which finds the independent factors hidden in a mixture of several unknown sources. More examples include Locally Linear Embedding (LLE) [13], Spectral Clustering (SC) [18], Non-Negative Matrix Factorization (NMF) [19], etc. The details about these learning algorithms will be presented in Chapter 4.

Similar to representation learners based on the prior of *manifold*, another popular way is to represent the raw images or patches by appealing to the prior of *sparsity*. Formally, it often reads a natural signal $\mathbf{x} \in \mathbb{R}^m$ (such as a m-pixel image) actually reside in a union of much lower dimensional subspace of dimension s, with $s \ll m$. For example, one wants to approximate a given image signal as a linear combination of as few as possible basis functions $\{\mathbf{d}_j \in \mathbb{R}^m\}$. Each basis function is called an atom and their collection is called a dictionary $\mathbf{D} = [\mathbf{d}_1, \cdots, \mathbf{d}_k] \in \mathbb{R}^{m \times k}$, such as wavelets of various sorts [20]. The dictionary is overcomplete $k > m$ when it spans the signal space and its atoms are linearly dependent. In that case, every signal $\mathbf{x} \in \mathbb{R}^m$ can be represented as a linear combination of a wavelet basis $\mathbf{D}$ as follows:

$$\mathbf{x} = \mathbf{D}\boldsymbol{\phi} = \sum_{i=1}^{k} \varphi_i \mathbf{d}_i, \tag{1.1}$$

where $\boldsymbol{\phi} = [\varphi_1, \cdots, \varphi_k]^\top \in \mathbb{R}^k$ represents the wavelet and scaling function coefficients. For most natural signals $\mathbf{x} \in \mathbb{R}^m$, most components of the vector $\boldsymbol{\phi}$ have negligible amplitude, i.e., it contains s coefficients that are large in magnitude while all other $k-s$ coefficients are very small or exactly zero in the ideal case, see Fig. 1.2. The s large coefficients carry all important information about the image and the corresponding wavelets span the subspace the image resides in. Geometrically, the set of s-sparse signals in the basis $\mathbf{D}$ consists of the union of all possible s-dimensional subspaces in $\mathbb{R}^m$ spanned by s-basis vectors from $\mathbf{D}$. Therefore, if $\boldsymbol{\phi}^\star$ represents the weights $\boldsymbol{\phi}$ with the smallest coefficients set to zero, the signal $\mathbf{x}$ is reconstructed by $\widetilde{\mathbf{x}} = \mathbf{D}\boldsymbol{\phi}^\star$. The relative reconstruction error $\|\mathbf{x}-\widetilde{\mathbf{x}}\|_2$ is often negligibly

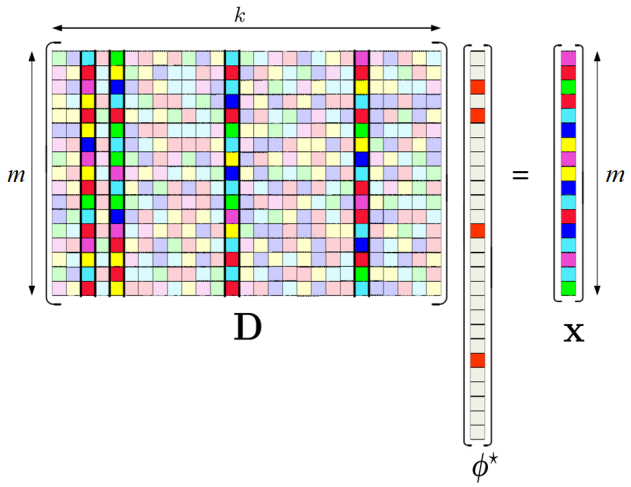


Figure 1.2: Sparse representation.

small for $s \ll k$. This property has led to the development of state-of-the-art compression algorithms based on wavelet-based transform coding, cf. [21, 22, 23].

Knowing that the dictionary is overcomplete ($k > m$), and hence $\boldsymbol{\phi}^\star$ is not unique. Now, the question becomes how to find a sparest $\boldsymbol{\phi}^\star$ that well reconstruct the input $\mathbf{x}$. This is where the sparsity constraint comes into play. To achieve efficient and sparse representations, ones generally relax the requirement for finding the exact representation. We look for a sparse linear expansion with an approximation error $\boldsymbol{\epsilon}$, i.e.,

$$\mathbf{x} = \mathbf{D}\boldsymbol{\phi}^\star + \boldsymbol{\epsilon} \tag{1.2}$$

with $\boldsymbol{\phi}^\star$ being a sparse vector in $\mathbb{R}^k$. The objective is now to find a sparse vector $\boldsymbol{\phi}^\star$ that contains a small number of significant coefficients, while the rest of the coefficients are close or equal to zero. Under some mild conditions on the dictionary $\mathbf{D}$, the sparse representation of $\mathbf{x}$ is computed by solving a constrained optimization problem of the form

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\boldsymbol{\phi}\|_0, \;\; \text{s.t.} \;\; \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2 \leq \varepsilon, \tag{1.3}$$

where $\varepsilon \in \mathbb{R}^+$ is reconstruction error. Therein, $\|\cdot\|_0$ denotes the ℓ_0-norm, and $\|\boldsymbol{\phi}\|_0$ counts the number of nonzero terms in $\boldsymbol{\phi}$. Unfortunately, this optimization problem $\|\cdot\|_0$ is an integer-valued, discontinuous and nonconvex function. The only known searching method for the exact solution is an intractable combinatorial search, which is well known to be NP-hard, cf. [24]. In order to overcome such difficulty, many relaxed approximation algorithms were proposed to find a suboptimal solution for the sparse vector $\boldsymbol{\phi}$, and the details will be described in Chapter 2.1.

Many literatures [21, 22, 23] have shown that some natural signals like image patches could be represented as sparse coefficients over some fixed dictionary, such as wavelet bases.

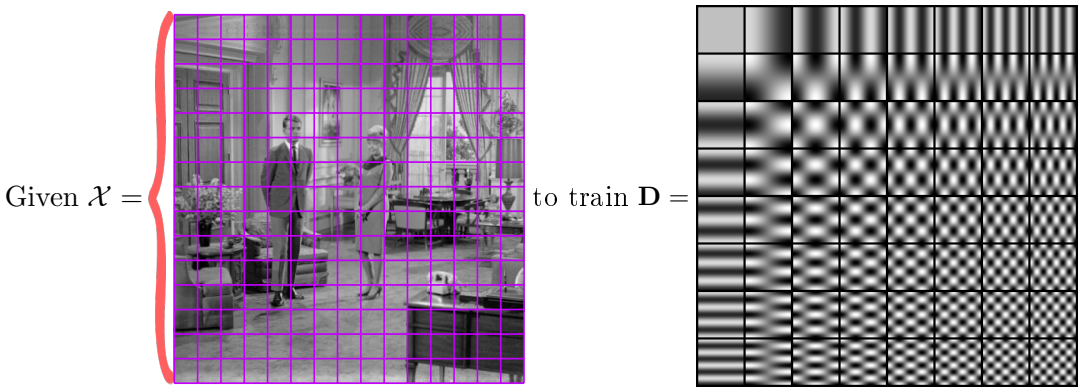


Figure 1.3: Learning dictionary for image patches.

However, for broader application areas, it is not easy to get a fixed base like wavelets with the capability of sparse representation. Therefore, jointly learning $\mathbf{D}$ and $\boldsymbol{\phi}$ is well developed, known as the dictionary learning (DL) problem [4, 25, 26]. As shown in Fig. 1.3, given a collection of training image patches $\mathcal{X}$, the goal of DL problem is to train a dictionary that allows each patch to be represented most accurately with a coefficient vector $\boldsymbol{\phi}$ that is as sparse as possible. The technical details will be introduced in Chapter 2.2. The sparse factorization of Eq. (1.1) is the backbone of many successful signal reconstruction, denoising and image classification [4, 25, 3].

Aforementioned research on design of data representation often imply that the input images have already been preprocessed, such as the well cropped images/patches set with uniform scales, good alignment and illumination. Then, some linear/nonlinear transformation model (e.g., PCA or sparse representation) is constructed directly on such features. When it comes time to achieve good results on tiny image processing problems, such as faces and handwritten digits, but it may fail to deal with the more general practical real-world computer vision problems. In summary, it may suffer from the following aspects: i) Most raw photographic images have very high dimension (millions of dimensions) under various changes, such as rotations, illuminations, centering offsets, etc. ii) The raw input images may contain multiple objects in a variety of scales, locations and viewpoints with complex backgrounds (e.g., background clutter and occlusion, or scenes in both outdoor and indoor). In addition, another challenge is that the raw image pixel intensity values may not provide enough unambiguous information to directly generate semantic-level concepts (e.g., the label of object).

With the aim of adapting more general computer vision problems, such as large scale object categorization, it needs to learn higher-level features on "raw" image input. One popular way is first to detect various local image features, such as Scale-invariant feature transform (SIFT) or Histogram of Oriented Gradients (HOG), cf. [27, 28, 29, 30]. These extracted features are often invariant to the scales, rotation, small image perturbation, even illumination. We

then quantize them into discrete "visual words" over a codebook or dictionary. Finally, it computes a spatial pyramid pooling (SPP) vector of acquired "visual words", cf. [27, 28, 29]. The objective of the model is to represent an image as a bag of visual words or features, called BoW or BoF. Fig. 1.4 shows a general pipeline for a visual recognition system based

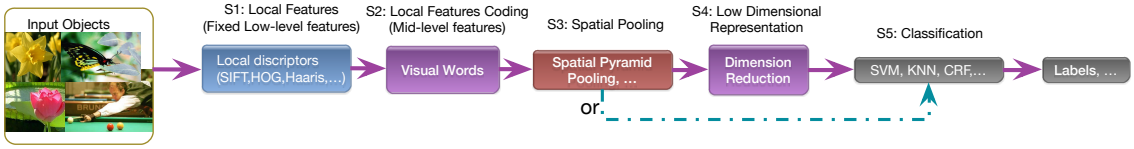


Figure 1.4: An example of general visual recognition pipeline using the bag-of-features model.

on the use of local feature extraction, features coding and SPP technique. This pipeline generates a single fixed-length vector that describes the entire image, which is convenient to learn a uniform task-related prediction model for each dataset. This example illuminates the whole process of classical visual recognition pipeline.

It is easy to see that the pipeline depicted in Fig. 1.4 relies on various hand-crafted local feature descriptors. Hand-crafted feature extraction provide a preliminary bridge between raw image pixels and semantic-level concepts. However, such a feature engineering is often labor-intensive and inflexible to adapt a wide variety of machine learning tasks. In order to address this challenge, many algorithms are emerged to exploit the *depth* of representation learning, called deep learning, i.e., learning multiple levels of representation. Deep learning algorithms admit the prior of the *depth* of explanatory factors, and seek to exploit the unknown structure in the input distribution in order to discover more abstract features in the higher levels of the representation. Fig. 1.5 shows a classical method known as the convolutional neural network (LeNet5 network), cf. [1]. It focused on tackling the vision problem through a fully-supervised multilayer network with convolution operators in each layer mapping their inputs to produce a new representation via a bank of filters. Such a highly adapted hierarchical local connectivity has the potential to encode structure suitable for modeling raw images [31]. Recent developments show that the convolutional deep network [32] can be used to classify large-scale image datasets with millions of training data, convincingly winning the ImageNet Large Scale Visual Recognition Challenge. However, such deep learning networks often requires huge amounts of training images/patches with prohibitive training time on the specialized computing hardwares, such as the multi-threaded computing on the GPU, cf. [32, 33].

1.2 Sparse Representation in Image Processing

Among the various ways of learning representations, sparse representation over a redundant dictionary is one widely adopted approach of representing the data, which has been verified as an efficient and useful tool to promote the tasks of image processing. For example, it has led to a great success in signal reconstruction, denoising and image super-resolution,

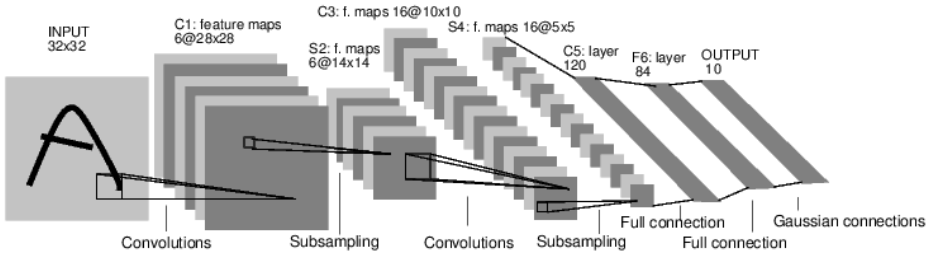


Figure 1.5: The LeNet5 network [1] was trained to recognize hand written characters in order to automatically process bank checks.

cf. [4, 25, 34]. Works in [28, 35, 36, 37] show that the locality could lead to the sparsity and hence the aforementioned manifold-based dimensionality reduction (DR) algorithms can be modeled as local sparse coding problems. Sparse representation is also an efficient technique to acquire the visual words of local descriptors in the classical visual recognition pipeline, which is depicted in Fig. 1.4. The results in [28, 37] show that such a local descriptors coding technique exactly improve the final object categorization accuracy. Recent literatures in [38, 39] valid that when the representations are learned in a deep neural network architecture that encourages sparsity, improved performance is obtained on image classification and object detection. It also has been empirically observed that the structure in sparse domain could make the hidden patterns more prominent and easier to be captured, and sparse coefficients are often interpreted as the extracted features to promote the tasks of machine learning, such as image classification in [3, 40, 41, 42]. Motivated by such progress, the focus of this dissertation is on the investigation of sparse coding methods and their potential applications in image and video processing. Before presenting our main contributions of this dissertation, in this section, we review some popular applications of sparse coding on solving computer vision problems. One intuitive illustration is shown in Fig. 1.6.

Many computer vision applications can be modeled as resolving a linear inverse problem. Prominent examples are image denoising [4], inpainting [25] or super-resolution [43], which refer to recover a high-quality image from its low-quality version, such as the image suffers from additive Gaussian noise, occlusion, missing or damaged portions, or very low resolution. Another one intriguing example is well known as Compressive Sensing (CS) [22, 23], which involves reconstructing an unknown image as accurately as possible from the measurements in far fewer dimensional space than what is usually considered necessary. Let us denote such down-sampled or corrupted measurements by $\mathbf{y} \in \mathbb{R}^l$, i.e.,

$$\mathbf{y} = \boldsymbol{\mathcal{A}}\mathbf{x} + \boldsymbol{\epsilon}, \tag{1.4}$$

where the vector $\boldsymbol{\epsilon} \in \mathbb{R}^l$ models sampling errors and noise, and $\boldsymbol{\mathcal{A}}$ is the measurement system modeling the sampling process. In this formulation, given a measurement system matrix $\boldsymbol{\mathcal{A}}$, determining $\mathbf{x}$ from the measurements $\mathbf{y}$ is the famous linear inverse problem. Certainly, when $\boldsymbol{\epsilon} = 0$ and $l \geq m$, the reconstructed signal $\mathbf{x}^\star \in \mathbb{R}^m$ simply can be computed via $\mathbf{x}^\star = \boldsymbol{\mathcal{A}}^\dagger \mathbf{y}$. However, in the presence of noise term $\boldsymbol{\epsilon} \neq 0$ or when the system is under-

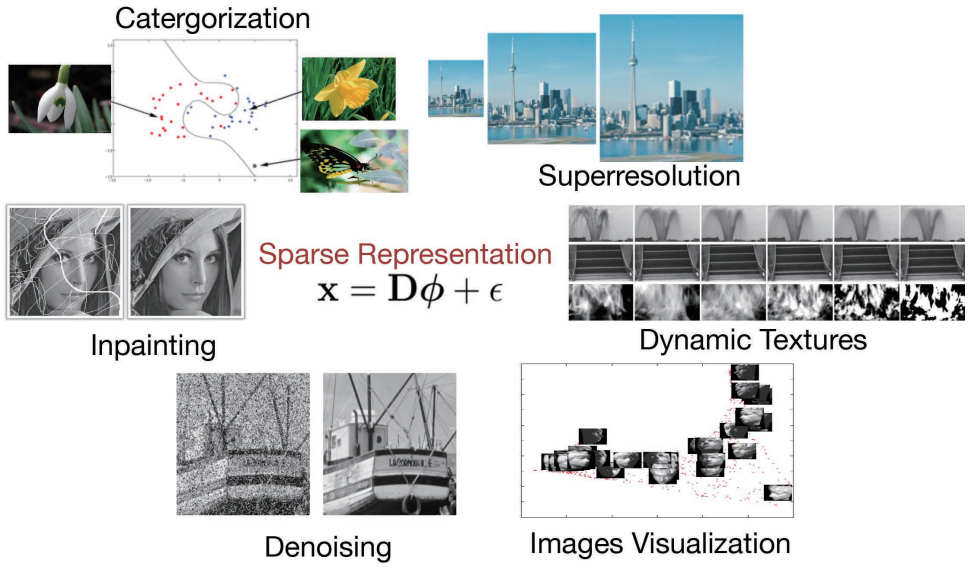


Figure 1.6: Incorporating sparse coding into computer vision problems, such as image denoising, inpainting, objects or scenes categorization, etc.

determined (i.e., $l < m$), there exist infinitely many solutions to the inverse problem of Eq. (1.4). However, if $\mathbf{x}$ is known to be sparse or could be sparsely represented in a given basis set $\mathbf{D}$, then under additional mild conditions on $\boldsymbol{\mathcal{A}}$ [23, 44], the reduced measurements $\mathbf{y}$ determine $\mathbf{x}$ uniquely as long as l is large enough. Formally, the sparse representation $\boldsymbol{\phi}$ of an original signal $\mathbf{x}$ is formulated in the following constrained ℓ_0-minimization

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\boldsymbol{\phi}\|_0, \;\; \text{s.t.} \; \|\mathbf{y} - \boldsymbol{\mathcal{A}}\mathbf{D}\boldsymbol{\phi}\|_2 \leq \varepsilon \tag{1.5}$$

with $\varepsilon \in \mathbb{R}^+$. Here, $\boldsymbol{\mathcal{A}}$ is called a sampling (measurement) matrix which could be chosen as a random matrix in CS, or $\boldsymbol{\mathcal{A}}$ is a square identity matrix in classical denoising or inpainting problem [4, 25]. $\mathbf{D} \in R^{m \times k}$ is the basis (such as wavelet basis [22, 23]), also called dictionary, in which all putative signals $\mathbf{x}$ are supposed to be sparse under the sparse factorization Eq. (1.3).

Using a wide adaption function $g : \mathbb{R}^k \to \mathbb{R}$ to measure the sparsity, the problem (1.5) can be recast as the following general minimization problem

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \sum_{i=1}^{k} g(\varphi_i), \;\; \text{s.t.} \; \|\mathbf{y} - \boldsymbol{\mathcal{A}}\mathbf{D}\boldsymbol{\phi}\|_2 \leq \varepsilon, \quad \mathbf{D} \in \mathfrak{D}(m,k), \tag{1.6}$$

where $\boldsymbol{\phi}^\star = [\varphi_1^\star, \ldots, \varphi_k^\star]^\top \in \mathbb{R}^k$, and $\mathfrak{D}(m,k)$ is some predefined admissible set of solutions for the dictionary.

1.3 Contributions and the Dissertation Outline

From a perspective of representation learning, it is a logical conclusion that sparse representations contain rich distributed information of the data with respect to certain learning tasks. This motivates us to construct effective learning mechanisms for further disentangling underlying explanatory information hidden in sparse representations. Such disentangled information is expected to make it easier to build machine learning predictors, such as linear classifiers. To be more specific, by exploring the differentiability of solutions of some sparse coding methods, the main objective of this dissertation is to investigate how it learns representative features in sparse coefficients of images data. Such learned representative features can improve the performance of some specific computational visual tasks, such as image classification, dimensionality reduction, 2/3D visualization and timing image sequence modeling.

The main contributions of this dissertation are summarized as follows.

(I) With the aim of disentangling useful information in sparse representation space for solving specific computer vision problems, we propose a two-layer representation learning framework. The related optimization problem can be treated as minimizing/maximizing a generic cost function, which involves a sparsifying dictionary and a further representation learning instrument in sparse representation. In this dissertation, such a further representation learning instrument is built according to specific computer vision problems, such as dynamic textures (DT) and image dimensionality reduction. Since the admissible sets of solutions to the constructed cost function are restricted on appropriate matrix manifolds, solving the related optimization problem requires that i) the differentiability of sparse representation with respect to a given dictionary, and ii) an efficient geometric gradient optimization algorithm for learning the model parameters.

To address the first requirement, this dissertation regards the sparse representation of data as a locally differentiable function with respect to a given dictionary. By adopting appropriate convex sparsity priors in the sparse coding formulation, we develop a generic form of the directional derivative of sparse representation with respect to the specific dictionary. Such a directional derivative can be adapted to a variety of convex sparsity priors, such as elastic net prior and Kullback-Leibler (KL)-divergence prior. Secondly, to give a suitable solution to the optimization problem of the constructed cost function, we develop a collection of efficient algorithms on appropriate differentiable manifolds.

(II) In benefit of learning image representations, we present to explore the evolution of the dynamic textured scenes in representation space. Concretely, we follow the framework of classical linear dynamical system (LDS), which assumes that each observation is correlated to an underlying "state", and the dynamic process of these consecutive states can be captured by a transition matrix. However, the LDS is sensitive to input variations due to various noise. Especially, it is vulnerable to non-Gaussian noise, such as missing data or occlusion of the dynamic scenes. To tackle these challenges, by treating the sparse coefficients over a learned dictionary as the underlying "states", we propose to learn a transition matrix between two adjacent frames of sparse events in time series. We construct a combined ℓ_2 regression, which involves jointly learn a sparsifying dictionary and a linear transformation

with several constraints. Such a learning scheme has been used for synthesizing dynamic textures, denoising and classifying a query DT sequence. In addition, numerical experiments show that an appropriately sparse transition matrix can significantly improve the results of DT recognition.

(III) We then consider one general problem of constructing effective low dimensional representation learning approaches to disentangle sparse coefficients for solving discriminative features learning problems. Our main construction is to apply trace quotient criterion on sparse coefficients for triple supervised, unsupervised and semi-supervised learning tasks. With the aim of leveraging the DR-driven sparse representation and signal reconstruction, we also developed several differentiable regularizations to promote the learning dictionary for having both reconstructive and discriminative power. Then, we construct a differentiable cost function, namely *SparLow*, to jointly learn a sparsifying dictionary and a corresponding DR transformation matrix.

This dissertation is organized as follows.

In Chapter 2, we provide a survey of state-of-the-art algorithms on sparse representation with respect to pre-defined dictionaries and dictionary learning with the goal of data reconstruction. By leveraging the local differentiability of solutions of some sparse coding methods, Chapter 3 introduces a generic cost function that learn representations in sparse coefficients for solving various computer vision problems. We then give a brief introduction of sparse regression with convex sparsity priors. A generic form of the directional derivative of sparse representation with respect to a given dictionary is also developed in this chapter. Since the admissible sets of solutions to such a cost function are restricted on matrix manifolds, some basic concepts of optimization on matrix manifolds are finally reviewed in Chapter 3. By taking advantage of the local smoothness of the elastic net solutions, Chapter 4 models the dynamic scene in sparse representation space. In this way, the dynamical process of dynamic textures exhibits a transition course of corresponding sparse events. By exploring the differentiability of more general sparse coding methods based on convex sparsity priors, a framework, called *SparLow*, is constructed in Chapter 5, for learning dictionary and orthogonal DR transformation with unsupervised, supervised and semi-supervised learning. The proposed approach has been adapted to a wide variety of image processing tasks, such as 2/3D data visualization, face/digit/cartoon recognition, and object categorization. Chapter 6 comes a conclusion of this dissertation. Some suggestions for future work are presented as well.

Chapter 2

Sparse Coding and the Related Optimization Algorithms

In this chapter, we first review basic knowledge of sparsity, sparse representation with respect to pre-defined dictionary. Then, we recall some technical details of dictionary learning methods.

2.1 Sparse Representation and the State-of-the-art Optimization Algorithms

Sparse Representation is established based on one basic concept of *sparsity*, which is the signal structure behind many data analysis algorithms that employ sparse factorization of Eq. (1.2). It is the most prevalent signal structure used beyond in compressive sensing, include signal denoising, deconvolution, restoration and inpainting.

Earlier research assume that the dictionary is predefined and solving the sparse factorization of Eq. (1.2) generally depends on how to choose the measurement of sparsity. Ideally, sparsity is measured by ℓ_0 norm, and learning spare representation corresponds to solving the problem (1.3). Formally, if $\boldsymbol{\phi} \in \mathbb{R}^m$ satisfies sparsity condition

$$\|\boldsymbol{\phi}\|_0 \leq s, \text{ for } s \in \mathbb{Z}^+, \tag{2.1}$$

we call the vector $\boldsymbol{\phi}$ is s-sparse. Here, $\|\boldsymbol{\phi}\|_0$ denotes the number of nonzero terms in $\boldsymbol{\phi}$. However, as introduced in Chapter 1.1, such a sparsity measurement function is nonconvex, highly non-robust against small perturbations of the zero elements, besides, finding sparsest solution to the corresponding sparse learning problem (1.3) is in general NP-hard [24].

To overcome this difficulty, many algorithms turn to find a suboptimal yet sparse enough representation, and one popular extension is to consider instead the ℓ_0 norm with ℓ_p, $0 < p \leq 1$ (the smaller we choose p, the more we are putting a premium on sparsity). The $p = 1$ special case, known as Lasso problem (least absolute shrinkage and selection operator) [45, 4], has become particularly popular since in this case the relaxation leads to a convex problem. Under such kind of sparsity measurement, the sparse representation of a signal $\mathbf{x}$ over a fixed dictionary $\mathbf{D} \in \mathbb{R}^{m \times k}$ can be found by solving the optimization

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\boldsymbol{\phi}\|_p, \text{ s.t. } \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2 \leq \varepsilon, \tag{2.2}$$

where $0 \leq p \leq 1$, and $\varepsilon \in \mathbb{R}^+$ is the residual term.

It is known that the linear system (1.1) is under-determined, or ill-conditioned when $m < k$. There are a lot of algorithms, which have been developed for learning sparsity under such an ill-conditioned linear system. In what follows, we review several state-of-the-art optimization algorithms that are important and popular for resolving the problem (2.2), under various sparsity measurement functions.

2.1.1 Convex Relaxation Algorithms

Convex relaxation is a well-known class of algorithms for sparse learning. A standard approach, as aforementioned, is based on the convex relaxation of ℓ_0-norm, namely, ℓ_1-norm, and the sparsest representation is the solution of either

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\boldsymbol{\phi}\|_1, \;\; \text{s.t. } \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2 \leq \varepsilon, \tag{2.3}$$

or

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2, \; \text{s.t. } \|\boldsymbol{\phi}\|_1 \leq s \tag{2.4}$$

with $\varepsilon \in \mathbb{R}^+$ and $s \in \mathbb{Z}^+$. Note that, problem (2.3) is a quadratically constrained linear program (QCLP), whereas (2.4) is a quadratic program (QP).

Problem of the form (2.3) or (2.4) is closely related to the following convex unconstrained optimization problem

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \frac{1}{2}\|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2 + \lambda\|\boldsymbol{\phi}\|_1, \tag{2.5}$$

called ℓ_1-regularized least squares problem (RLS), where $\mathbf{x} \in \mathbf{R}^m$ and $\mathbf{D} \in \mathbb{R}^{m \times k}$, $\lambda \in \mathbb{R}^+$ weighs the sparsity and reconstruction error. From a Bayesian perspective, the form (2.5) can be seen as a maximum a posteriori (MAP) criterion for estimating $\boldsymbol{\phi} = [\varphi_1, \ldots, \varphi_k]^\top \in \mathbb{R}^k$ from observations $\mathbf{x} = \mathbf{D}\boldsymbol{\phi} + \boldsymbol{\epsilon}$. The prior distribution for the elements of coefficient vector $\boldsymbol{\phi}$ is assumed to be Laplace in $\mathbb{R}$, which is a priori independent from $\boldsymbol{\epsilon}$ and could be defined as

$$p(\boldsymbol{\phi}) = \prod_{i=1}^{k} p(\varphi_i), \; p(\varphi_i) = \lambda \cdot \exp\{-\lambda|\varphi_i| + \mathbf{K}\}. \tag{2.6}$$

Problem of the form (2.5) has been used for more than four decades in several signal processing problems where sparseness is sought. Convex analysis can be used to show that a solution of (2.3) is either $\boldsymbol{\phi} = \mathbf{0}$ or else is a minimizer of (2.5) with $\lambda \in \mathbb{R}^+$. Similarly, a solution of (2.4) for any $t \geq 0$ is also a minimizer of (2.5) with $\lambda \in \mathbb{R}^+$. More details we refer the interested readers to [46].

Unlike the ℓ_0-norm which enumerates the nonzero coordinates, the presence of the ℓ_1 term encourages small components of $\boldsymbol{\phi}$ to become exactly zero, thus promoting sparse solutions, cf. [45, 47]. It is known that minimizing ℓ_1-norm is a convex optimization problem that conveniently reduces to a linear program known as basis pursuit (BP) [47], whose computational complexity is about $O(k^3)$. Such a solution implies that the ℓ_1-norm is computationally more tractable than the ℓ_0-norm, which requires an exhaustive enumeration

of all C_k^s possible locations of the nonzero entries in $\boldsymbol{\phi}$. In addition, literatures in [48, 49] showed exact equivalence between the two programs ℓ_1 norm and ℓ_0 norm.

Based on ℓ_1-minimization, many optimization algorithms and codes have been proposed to solve the QCLP (Eq. (2.4)), the QP (Eq. (2.4)), and the unconstrained RLS formulation Eq. (2.5). For example, the Lasso regression has the form of Eq. (2.4), while the BP [47] has the form of Eq. (2.3) with $\varepsilon = 0$, i.e., a linear program (LP)

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\boldsymbol{\phi}\|_1, \;\; \text{s.t. } \mathbf{x} = \mathbf{D}\boldsymbol{\phi}, \tag{2.7}$$

and basis pursuit denoising (BPDN) has the form of Eq. (2.3) with $\varepsilon > 0$, cf. [50, 47].

It is known that the formulations Eq. (2.3), Eq. (2.4), and Eq. (2.7) are classical linear programming problems, therefore, it can be solved with general simplex methods or interior point methods, cf. [51, 52, 53]. E.J.candès and Tao propose to solve the LP problem (2.7) using a generic path-following primal-dual method [54]. Similar to LP, E.candès and J. Rombergas show that the second-order cone programs (SOCP) [55] are also computationally tractable on solving ℓ_1 minimization problem. Besides such basic model selection algorithms, homotopy algorithms [56, 57] are applied to find the full path of solutions to the quadratic programming formulation of Eq. (2.5). The name Homotopy refers to the fact that this objective function is undergoing a homotopy from the ℓ^2 constraint to the ℓ_1 objective as λ decreases, i.e., it solves the problem Eq. (2.5) for essentially all values of λ. As another one typical homotopy method, the least angle regression (LARS) procedure described in [50] can be adopted to solve the Lasso problem (2.4).

However, such kind of solvers reach a best solution by traversing the interior of the feasible region and may not scale well when the dimension of involved problem is big. To accelerate the convergence, a technique based on Barzilai-Borwein (BB) steps is used. The typical approaches are known as fixed-point continuation (FPC) method [58], and its improvement version, called fixed-point continuation and active set approach (FPC-AS)[59, 60]. FPC is based on two powerful algorithmic ideas: continuation and operator-splitting, to solve the general problem (2.5). They established a Q-linear rate of convergence of the method without assuming strict convexity nor uniqueness of solution. FPC can be simply explained as the decomposition of a maximal monotone operator $T := \nabla(|\boldsymbol{\phi}|_1 + \frac{1}{2}\|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2)$ into the sum of two maximal monotone operators $T_1 := \frac{1}{\lambda}\partial|\boldsymbol{\phi}|_1$ and $T_2 := \partial(\|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2)$, and then naturally consider the fixed point iterations as: $\boldsymbol{\phi}^{k+1} = (\boldsymbol{I} + \tau T_1)(\boldsymbol{I} - \tau T_2)\boldsymbol{\phi}^k$ with $\tau \in \mathbb{R}^+$ being an adjustable parameter.

Similar to FPC algorithms, a gradient projection (GP) algorithm, called *gradient projection for sparse reconstruction* (GPSR), is proposed to find sparse solutions to the problem (2.5), cf. [61]. In GPSR, the search path from each iteration is obtained by projecting the negative-gradient direction onto the feasible set. Different to the second part T_2 of the FPC operator T, GPSR reformulate Eq. (2.5) as a bound-constrained quadratic program, and then apply projected gradient steps, optionally using a variant of Barzilai-Borwein steps (GPSR-BB) in order to accelerate convergence. Another one gradient methods, called spectral projected gradient for ℓ_1 minimization (SPGL1) [62, 63], utilize the pareto root-finding

approach to perform the ℓ_1 regularization optimization problem (2.5). At each iteration, a spectral gradient-projection (SPG) method approximately minimizes a least-squares problem with an explicit ℓ_1-norm constraint. The algorithm is suitable for problems that are large scale and for those that are in the complex domain. Experimental study shows that it exactly improve the performance of original BP, BPDN and Lasso for large scale data, and the related SPGL1 software package is online available at `http://www.cs.ubc.ca/labs/scl/spgl1/`. In order to speed up the gradient approach, a block coordinate gradient descent method is proposed in [64] to solve the ℓ_1 regularized optimization problem (2.5). The Q-linear convergence rate can be obtained when the coordinate block is chosen by a Gauss-Southwell-type rule. Very recently, M. Afonso et al. propose for solving the problem Eq. (2.5) based on a variable splitting to obtain an equivalent constrained optimization formulation, which is then addressed with an augmented Lagrangian method, called Split Augmented Lagrangian Shrinkage Algorithm (SALSA) [65].

2.1.2 Iterative Shrinkage/Thresholding Algorithms

Iterative shrinkage/thresholding (IST) algorithms, also tailored for objective functions with the form of Eq. (2.5), were independently proposed by several authors in different frameworks. Initially, IST was presented as an Expectation–maximization (EM) algorithm, in the context of image deconvolution problems, cf. [66]. IST can also be derived in a majorization-minimization (MM) framework [67, 68, 69]. Convergence of IST algorithms was shown in [70, 67]. In the original IST [67, 68], the formulation of Eq. (2.5) can be written as an iterative equation

$$\boldsymbol{\phi}^{(t+1)} = (1-\beta)\boldsymbol{\phi}^{(t)} + \beta\mathcal{P}\left(\boldsymbol{\phi}^{(t)} + \mathbf{D}^{\top}(\mathbf{x} - \mathbf{D}\boldsymbol{\phi}^{(t)})\right), \tag{2.8}$$

where $0 < \beta < 1$ is a tuning parameter, and $\mathcal{P}$ is a projection or transform function, e.g., $\mathcal{P}$ is the wavelet transform [68]. Under the updating form (2.8), each iteration of the IST algorithm only involves sums of residuals. Benefit from its simple form, various IST methods were proposed as iterative shrinkage/thresholding methods to resolve the problem (2.5), cf. [67, 71, 72]. Recently, a novel two-step IST method (TwIST) was presented by J.Dias and M.Figueiredo [71], achieving a significantly faster convergence rate than original IST.

2.1.3 Greedy Pursuit Algorithms

Iterative greedy pursuit is another well-known class of algorithms for learning sparse representation, cf. [73]. Rather than minimizing an global objective function, these methods are iterative select columns of $\mathbf{D}$ according to their correlation with the measurements $\mathbf{x}$ determined by an appropriate inner product, and then construct a sparse solution to this given problem by iteratively building up an approximation. The earliest ones include the matching pursuit (MP)[74] and orthogonal matching pursuit (OMP)[75]. MP works by iteratively choosing the dictionary element that has the highest inner product with the current

residual, thus it reduces the reconstruction error at most. Considering it maybe emerge the suboptimal solutions from each iteration in MP, OMP follows the atomic selection criteria in the MP algorithm, and ensures the optimality of each iteration just by recursively orthogonalized atomic collection of selected subsets. OMP includes an extra orthogonalization step, which is known to reduce the number of iterations.

On this basis, various methods have been developed and improved including the Regularized Orthogonal Matching Pursuit (ROMP)[76], Compressive Sampling Matching Pursuit (CoSaMP)[77], Stagewise Orthogonal Matching Pursuit (StOMP)[78], etc. Zhang [79] recently proposed a combination algorithm that is based on the forward greedy algorithm but takes backward steps adaptively whenever beneficial.

Low computational cost is one of the main arguments in favor of greedy schemes like OMP. Thus, they are easy to invent, easy to implement and most of the time quite efficient. Many general optimization problems can be solved correctly by greedy approaches. However, such methods are not designed to solve any of the optimization problems on learning sparse, e.g., minimizing an global objective functions like Eq. (2.5).

2.1.4 Non-convex Optimization Algorithms

We have reviewed some literatures that using convex relaxed sparsity measures, i.e., ℓ_1-norm, or turning to greedy optimization. Although these methods show good performance in practical applications, they inherently require a degree of over-sampling above the theoretical minimum sampling rate to guarantee that exact reconstruction can be achieved. In this section, we consider the case of replacing the ℓ_1-norm by a ℓ_p norm with $0 \leq p < 1$, which is non-convex but a sharper measure on sparsity.

Among the existing methods, the IST method is quite universal, robust and much easier to be implemented by engineers. Similar to IST for resolving convex optimization problem (2.5), IST has also been used to directly solve ℓ_0 regularized optimization problem

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2 + \lambda\|\boldsymbol{\phi}\|_0,\ \lambda \in \mathbb{R}^+ \tag{2.9}$$

by Kingsbury, N.G in [80]. More recently, T. Blumensath and M. Davies [72] proposed two nonconvex iterative hard thresholding algorithms(IHT) that are directly minimizing the ℓ_0 regularized problem to find non-zero coefficients of $\boldsymbol{\phi}$, i.e.,

$$\boldsymbol{\phi}^\star := \arg\min_{\boldsymbol{\phi}} \|\mathbf{x} - \mathbf{D}\boldsymbol{\phi}\|_2^2, \quad \text{s.t.} \quad \|\boldsymbol{\phi}\|_0 \leq s,\ s \in \mathbb{Z}^+. \tag{2.10}$$

But they are very sensitive to initialization, for example initializing the coefficients with zero, the algorithms were often found to perform worse than Matching Pursuit. L.C. Patrick and R.W. Valèrie [81] presented a general iterative thresholding derived for signal recovery from forward-backward splitting. I. Daubechies et al. [82] gave a rigorous convergence proof for iterative soft thresholding. T. Blumensath, M. Davies [83, 72] gave a mathematical analysis for IHT and then extend the application from general sparse approximation to Compressive Sensing.

Different to aforementioned non-smooth IST or IHT algorithms for solving ℓ_0 minimization problem, Sℓ_0 (Smoothed ℓ_0) [84, 85] is a smooth approximation algorithm for finding the sparsest solutions of an undetermined system of linear Eq. (1.1). It approximates the ℓ_0 norm of a vector $\boldsymbol{\phi}$ by a smooth exponential function $F_\sigma(\boldsymbol{\phi})$, where σ determines the quality of approximation: The larger σ, the smoother $F_\sigma(\cdot)$ but worse approximation to the ℓ_0 norm; and the smaller σ, the better approximation to the ℓ_0 norm but the less smooth $F_\sigma(\cdot)$. Sℓ_0 was firstly proposed by H. Mohimani et al[84], and then was extended to deal with complex-valued signals [86]. More recently, H.Mohimani et al[85] study the convergence properties of Sℓ_0, and show that under a certain sparsity constraint in terms of Asymmetric Restricted Isometry Property (ARIP), the convergence of Sℓ_0 to the sparsest solution is guaranteed, as well as with same order complexity as Matching Pursuit (MP).

We have discussed some ℓ_0-minimization algorithms using iterative thresholding, in what follows, another non-convex relaxation optimization problem, ℓ_p minimization with $0 < p < 1$, will be introduced.

We say $\mathbf{x}$ can be recovered by ℓ_p-minimization if and only if it is the unique solution to the problem (2.2). The related optimization problem can be viewed as the focal underdetermined system solver (FOCUSS), cf. [87, 88], which is very similar to convex relaxations optimization methods, i.e., using the ℓ_p-norm with as a replacement for the ℓ_1-norm. Here, for $p < 1$, the similarity to the true sparsity measure is better but the overall problem becomes nonconvex, giving rise to local minima that may mislead in the search for solutions. Work in [44, 89] demonstrate that the much simpler task of finding a local minimizer can produce exact reconstruction of sparse signals with many fewer measurements than when $p = 1$. However, such methods often result in the high computational cost and, in some cases, compromised convergence.

FOCUSS, proposed in [87, 88], develop a nonparametric algorithm designed to address the shortcomings of such techniques. Namely, the algorithm provides a relatively inexpensive way to accurately reconstruct sparse signals. FOCUSS algorithm consists of two parts: It starts by finding a low resolution estimate of the sparse signal, and then, this solution is pruned to a sparse signal representation. The pruning process is implemented using a generalized Affine Scaling Transformation (AST), which scales the entries of the current solution by those of the solutions of previous iterations. Considering the underdetermined matrix equation form of Eq. (1.1), a straightforward solution is computed as

$$\boldsymbol{\phi} = \mathbf{D}^\dagger \mathbf{x} \tag{2.11}$$

where $\mathbf{D}^\dagger = \mathbf{D}^T(\mathbf{D}\mathbf{D}^T)^{-1}$ denotes the Moore–Penrose inverse. This solution has the advantage of low computational cost, but it does not provide sparse solutions. Compared to the search methods with maximum sparsity constraint, the FOCUSS algorithm utilize a weighted minimum norm to promote sparsity, i.e., ℓ_p diversity ($p < 1$), given by

$$\|\boldsymbol{\phi}\|_p^p = \operatorname{sgn}(p) \sum_{i=1}^{k} |\varphi_i|^p, \quad p < 1.$$

The FOCUSS algorithm gives an exact solution to problem (1.1). The basic form of the FOCUSS algorithm is (at t^{th} iteration)

$$\begin{aligned} \mathbf{W}_{t+1} &= (\text{diag}(\boldsymbol{\phi}_{t-1}[i])) \\ \mathbf{q}_{t+1} &= (\mathbf{D}\mathbf{W}_{t+1})^{\dagger}\mathbf{x} \\ \boldsymbol{\phi}_{t+1} &= \mathbf{W}_{t+1}\mathbf{q}_{t+1}. \end{aligned} \tag{2.12}$$

While considering the general model of Eq. (1.1) with noise, a variation of the FOCUSS algorithm that allows noise has been discussed in [90]. The iterations are given by

$$\begin{aligned} \mathbf{W}_{t+1} &= \text{diag}\big((|\boldsymbol{\phi}_{t-1}[i]|^{1-\frac{p}{2}})\big) \\ \mathbf{q}_{t+1} &= \arg\min_{\mathbf{q}} \|\mathbf{D}\mathbf{W}_{t+1}\mathbf{q} - \mathbf{x}\| + \lambda\|\mathbf{q}\|^2 \\ \boldsymbol{\phi}_{t+1} &= \mathbf{W}_{t+1}\mathbf{q}_{t+1}, \end{aligned} \tag{2.13}$$

where Eq. (2.13) is a regularization optimization problem. The two terms in Eq. (2.13) are a function of the parameter λ, and the regularization problem is a compromise between sparsity and error in the representation.

R. Chartrand [89, 91], Chartrand and Yin [92], Saab and Yilmaz [93] have proposed several nonconvex ℓ_p-norm optimization methods for reconstructing a sparse signal, which showed that the ℓ_p-minimization could be more effective for some special cases under weaker Restricted Isometry Constant (RIP) condition. Recently, R. Chartrand and his cooperators [94] extend the results of Candès, Romberg and Tao [22] to the $p < 1$ case and proved that ℓ_p minimization with certain values of $p < 1$ provides better theoretical guarantees in terms of stability and robustness to noise level, than ℓ_1 minimization does.

2.2 Dictionary Learning

The aforementioned methods for learning sparse representation approximate the observations with the linear combination of a few column vectors (or atoms) from a fixed dictionary. The performance of these algorithms in terms of the approximation quality and the sparsity of coefficient vector depends not only on the signal itself, but also on the choice of dictionary. In the simplest case, the dictionary is orthogonal, and the representation coefficients can be computed as inner products of the signal and the atoms in dictionary. For example, the orthogonal transforms, such as discrete cosine transform (DCT) and discrete wavelet transform (DWT), provide a unique representation for a given signal, and have been widely employed in signal processing due to their mathematical simplicity.

On the contrary, redundant or overcomplete dictionaries (referred to as redundant systems) do not have a unique representation for a given signal. Expanding a signal under a redundant system raises an ill posed problem, but also provides extra freedoms of selecting an optimized solution, and have more power on signal expressiveness. Typically, overcomplete bases are

constructed by merging a set of complete bases (e.g., Fourier, wavelet, and Gabor), or by adding basis functions to a complete basis (e.g., adding frequencies to a Fourier basis).

Although overcomplete bases can be more fexible in terms of how the signal is represented, there is no guarantee that hand-selected basis vectors will be well matched to the structure in the natural data, e.g., images. Ideally, one would like the basis itself to be adapted to the data, so that for signal class of interest, each basis function captures a maximal amount of structure. Such a type of dictionaries deliver increased flexibility and the ability to adapt to specific signal data. Therefore, recently, many research efforts are attracted on how to jointly learn dictionary $\mathbf{D}$ and sparse solution $\boldsymbol{\phi}$ from the input data, cf. [4, 25, 26].

Compared to the pre-determined dictionaries, the main advantage of learning based dictionaries is that they fit the input images or signals and can significantly improve the sparsity and thus the results of signal processing. Roughly speaking, the research in dictionary learning (DL) has followed two main directions that correspond to two categories of algorithms: i) data-driven dictionary learning methods, i.e., the methods for learning dictionaries with goal of data expressiveness, and ii) target-driven dictionary learning methods, i.e., the methods for learning dictionaries with a particular signal processing target, such as classification. In what follows, we first present the main principles of representative algorithms in data-driven dictionary learning category. Some popular methods on target-driven dictionary learning will be introduced in Chapter 3.

Given a set of data samples $\mathbf{x}_i \in \mathbb{R}^m$, the aim of data-driven dictionary learning is to find a collection of atoms $\mathbf{d}_j \in \mathbb{R}^m$ such that each data sample can be approximated by a linear combination of only a few of the atoms $\{\mathbf{d}_j\}$. The collection of atoms (often as columns in a matrix) is called a dictionary $\mathbf{D} \in \mathbb{R}^{m\times k}$. The focus of this section consider atoms in the dictionary are not required to be orthogonal, but overcomplete.

Let $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$ be the matrix containing the n independent training samples arranged as its columns, the task of dictionary learning focuses on finding the best dictionary to sparsely represent the elements of $\mathbf{X}$. Formally, let $\boldsymbol{\Phi} := [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n] \in \mathbb{R}^{k\times n}$ contain the corresponding n sparse transform coefficient vectors, a common approach to the classical dictionary learning techniques [95, 96, 97, 98, 4, 99, 100], is the optimization problem

$$\begin{aligned} &\min_{\mathbf{D},\boldsymbol{\Phi}} \ell_{\mathbf{X}}(\mathbf{D},\boldsymbol{\Phi}) \\ \ell_{\mathbf{X}}(\mathbf{D},\boldsymbol{\Phi}) &:= \frac{1}{2}\|\mathbf{X}-\mathbf{D}\boldsymbol{\Phi}\|^2 + \sum_{i=1}^{n} g(\boldsymbol{\phi}_i), \quad \text{s.t.} \quad \mathbf{D} \in \mathfrak{D}(m,k), \end{aligned} \tag{2.14}$$

where $\boldsymbol{\phi} = [\varphi_1, \cdots, \varphi_k]^\top \in \mathbb{R}^k$, the first term penalizes the reconstruction error of sparse representation, $g : \mathbb{R}^k \to \mathbb{R}$ is a function that promotes the sparse structure of $\boldsymbol{\phi}$, and $\mathfrak{D}(m,k)$ is some predefined admissible set of solutions for the dictionary.

Generally, $\mathfrak{D}(m,k)$ are defined as the convex set of matrices that satisfying several constraints. In the following, we give two definitions that are frequently used in many DL methods:

- In the first case, with the aim of preventing the ℓ_2-norm of $\{\mathbf{d}_i\}$ from being arbitrarily large, which would lead to arbitrarily small value of φ_i, $\mathfrak{D}(m,k)$ is defined as a convex set of matrices satisfying

$$\mathcal{D}(m,k) := \left\{ \mathbf{D} \in \mathbb{R}^{m\times k} \middle| \, \mathrm{rank}(\mathbf{D}) = \omega, \|\mathbf{d}_i\|_2 \leq 1 \right\} \tag{2.15}$$

  with $\omega := \min(m,k)$. Such a definition has been widely favored by lots of gradient based methods, such as efficient ℓ_1 [99] and online DL (ODL) [101].

- In the second case, it restricts each column $\mathbf{d}_i \in \mathbb{R}^m$ of $\mathbf{D}$ to have unit norm, i.e.,

$$\mathcal{S}(m,k) := \left\{ \mathbf{D} \in \mathbb{R}^{m\times k} \middle| \, \mathrm{rank}(\mathbf{D}) = \omega, \|\mathbf{d}_i\|_2 = 1 \right\}, \tag{2.16}$$

  which is a product manifold of $(m-1)$-dimensional unit spheres, known as Oblique manifold [102]. Such a definition could sharply avoid the problem of scale ambiguity, and has been widely adopted, such as K-SVD [4] and separable DL (SDL) [25].

Earlier work in [95] assumed the prior probability distribution over the elements of sparse vector $\boldsymbol{\phi}$ was Cauchy or Gaussian and took the Kullback-Leibler (KL) divergence to promote the sparsity of $\boldsymbol{\phi}$. Differently, work in [4, 99] assumed the elements of $\boldsymbol{\phi}$ admitted the Laplace distribution, and ℓ_1-norm was used to measure the sparsity. Methods in [96, 87, 98, 25] utilized ℓ_p-norm $(0 < p \leq 1)$ to measure the sparsity. The iterative shrinkage algorithms, such as Iterative least square based DL (ILS-DL)[100] and Recursive Least Squares based DL (RLS-DL) [103], used the approximation of ℓ_0-norm to measure sparsity.

With the sparsity measurement at hand, a practical optimization strategy, not necessarily leading to a global optimum, can be found by splitting the problem into two parts which are alternately solved within an iterative loop. The two parts are often described as

1. Keeping fixed $\mathbf{D}$, update $\boldsymbol{\Phi}$;
2. Keeping fixed $\boldsymbol{\Phi}$, update $\mathbf{D}$.

Such a learning scheme is shared by lots of DL methods, such as the generalized Lloyd algorithm (GLA) [104], K-SVD [4], ILS-DL [100], and efficient ℓ_1 [99].

In the following, we present some representatives of classical data-driven DL methods.

2.2.1 Method of Optimal Directions and its Extensions

Engan et. al. [96, 105] present an appealing dictionary training algorithm, namely, method of optimal directions (MOD). MOD follows more closely the K-Means outline, with a sparse coding stage that uses either OMP or FOCUSS followed by an update of the dictionary. The main contribution of the MOD method is its simple and efficient way of updating the dictionary.

Assuming that $\boldsymbol{\Phi}$ is fixed, we can seek an update to $\mathbf{D}$ such that the

$$R(\mathbf{D}) := \|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\|_F^2$$

is minimal. Taking the derivative of $R(\mathbf{D})$ with respect to $\mathbf{D}$, we obtain the relation $(\mathbf{X} - \mathbf{D}\boldsymbol{\Phi})\boldsymbol{\Phi}^\top = 0$. This results in a simple update for $\mathbf{D}$ in $(t+1)^{\text{th}}$ iteration:

$$\mathbf{D}^{(t+1)} = \mathbf{X}\boldsymbol{\Phi}^{(t)\top}(\boldsymbol{\Phi}^{(t)}\boldsymbol{\Phi}^{(t)\top})^{-1}. \tag{2.17}$$

Finally, we normalize $\mathbf{D}$, i.e., scale each column vector (atom) of $\mathbf{D}$ to unit norm.

It is known that MOD was aimed at learning block oriented dictionaries with the application of signal compression. Later work extended MOD to the design of overlapping dictionaries [106, 107]. The essence of MOD and its extensions is summarized in [100], where the least square approach of the different variants is clearly presented. Such approaches are included in the family of algorithms, namely, iterative least squares dictionary learning algorithm (ILS-DLA). Work in [103] follow the ILS-DLA, and go one step further to develop the algorithm into a recursive least squares (RLS) algorithm, called RLS-DLA.

MOD and its extensions have proved to be efficient for representing low-dimensional input data while requiring only a few iterations to converge. However, for high-dimensional data, the inversion operation in Eq. (2.17) often leads to a very high computational cost. On the other hand, computing the pseudoinverse is in many cases intractable.

2.2.2 Clustering Based Methods

A slightly different family of dictionary learning techniques to MOD, is based on vector quantization (VQ) achieved by K-means clustering. Such kind of algorithm optimizes a dictionary given a set of image patches by first grouping patterns such that their distance to a given atom is minimal, and then by updating the atom such that the overall distance in the group of patterns is minimal.

Such kind of methods is typically known as K-SVD [4], in which a generalized K-Means clustering process is proposed. The related objective function is constructed as

$$\min_{\mathbf{D},\boldsymbol{\Phi}} \|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\|_F^2, \quad \text{s.t.} \quad |\boldsymbol{\phi}_i|_0 \leq s, \ \forall i. \tag{2.18}$$

Followed the learning scheme of MOD, i.e., alternatively update $\mathbf{D}$ and $\boldsymbol{\Phi}$. In the first step, update $\boldsymbol{\Phi}$ via the OMP algorithm while $\mathbf{D}$ is fixed; In the second step, a Singular Value Decomposition (SVD) of the error matrix $(\mathbf{X} - \mathbf{D}\boldsymbol{\Phi})(\mathbf{X} - \mathbf{D}\boldsymbol{\Phi})^\top$ is used to update $\mathbf{D}$. This approach is an approximation of the ℓ_0-norm solution.

K-SVD is considered to be standard for dictionary learning and many extensions have been proposed in a variety of applications, cf. [108, 109]. However, it shares weaknesses with MOD being efficient only for signals with relatively low dimensionality and having the possibility for a solution to be stuck at local minima.

This shortcoming has inspired the development of other dictionary learning methods, such as gradient based approaches.

2.2.3 Lagrange Dual Method

Different to afore-described iterative shrinkage algorithms that are appealing to non-convex sparsity measures, the methods in [99] adopted the convex sparse learning formation (2.5), and solved the problem

$$\min_{\mathbf{D}\in\mathcal{D}(m,k),\{\boldsymbol{\phi}_i\in\mathbb{R}^k\}} \frac{1}{2}\sum_{i=1}^{n}\|\mathbf{x}_i-\mathbf{D}\boldsymbol{\phi}_i\|_2^2+\lambda\sum_{i=1}^{n}\|\boldsymbol{\phi}_i\|_1. \tag{2.19}$$

This problem can be written more concisely in matrix form, i.e.,

$$\min_{\mathbf{D}\in\mathcal{D}(m,k),\boldsymbol{\Phi}} \frac{1}{2}\|\mathbf{X}-\mathbf{D}\boldsymbol{\Phi}\|_F^2+\lambda\sum_{i=1}^{n}\|\boldsymbol{\phi}_i\|_1. \tag{2.20}$$

The basic idea in [99] is to alternatively minimize Eq. (2.20) over $\boldsymbol{\phi}$ for a given dictionary $\mathbf{D}$, and then over $\mathbf{D}$ for a given $\boldsymbol{\phi}$, leading to a local minimum of the overall objective function. The technical details can be stepped as following: i) While keeping the bases fixed, update $\boldsymbol{\phi}_i$ via solving (2.5) using the feature-sign search algorithm; ii) Given fixed coefficients $\boldsymbol{\Phi}_i$, update $\mathbf{D}$ solving a Lagrange dual problem. Consider the following Lagrangian:

$$\mathcal{L}=\frac{1}{2}\|\mathbf{X}-\mathbf{D}\boldsymbol{\Phi}\|_F^2+\sum_{i=1}^{k}\lambda_i\left(\sum_{j=1}^{k}\mathbf{d}_{ij}^2-c\right),$$

where c is a constraint on the norm of the atoms and λ_i are the so-called dual variables forming the diagonal matrix $\boldsymbol{\Lambda}$. We can then provide an analytical expression for the Lagrange dual after minimization over

$$\min_{\mathbf{D}}\mathcal{L}(\mathbf{D},\boldsymbol{\Lambda}) \text{ with } \mathcal{L}(\mathbf{D},\boldsymbol{\Lambda}):=\|\mathbf{X}\mathbf{X}^\top-\mathbf{X}\boldsymbol{\Phi}^\top(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top+\boldsymbol{\Lambda})^{-1}(\mathbf{X}\boldsymbol{\Phi}^\top)^\top-c\boldsymbol{\Lambda}\|_F^2.$$

Hence the optimal bases $\mathbf{D}$ has a closed expression

$$\mathbf{D}^\top=(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top+\boldsymbol{\Lambda})^{-1}(\mathbf{X}\boldsymbol{\Phi}^\top)^\top,$$

where Λ contains all k Lagrangian dual variables. Note that, this problem is also known as basis pursuit [47], or the Lasso [45].

Such a learning scheme leads to lots of task-specified dictionary learning methods, such as super-resolution [43, 34] and object categorization [27, 28].

Solving this problem is less computational hard because the amount of dual variables k is a lot of times much less than the amount of variables in the primal problem. However, like MOD or K-SVD, due to the high complexity of the inversion operation, computing the pseudoinverse in high dimensional cases is in many cases intractable.

2.2.4 Learning Dictionary Based on Stochastic Gradient Descent Algorithms

Most aforementioned algorithms for dictionary learning, cf. [95, 96, 97, 4, 99], are iterative batch procedures, accessing the whole training set at each iteration in order to minimize a cost function under some constraints. Thus, they cannot efficiently deal with very large training sets (e.g., millions of patches) [110]. To address these issues, different to efficient ℓ_1 DL, which use the first-order gradient descent method to update the dictionary, authors in [111, 112] propose to use the classical projected first-order stochastic gradient descent algorithm. It consists of a sequence of updates of

$$\mathbf{D}^{(t+1)} = \prod_{\mathcal{D}}[\mathbf{D}^{(t)} - \delta_{t+1}\nabla_{\mathbf{D}}\ell_{\mathbf{x}^{(t+1)}}(\mathbf{D}^{(t)}, \boldsymbol{\phi}^{(t)})] \tag{2.21}$$

where $\ell_{\mathbf{x}^{(t+1)}}(\mathbf{D}^{(t)}, \boldsymbol{\phi}^{(t)})$ is the loss function for DL, defined in Eq. (2.14), $\mathbf{D}^{(t+1)}$ is the estimate of the optimal dictionary at iteration $(t+1)$, δ_{t+1} is the gradient step, $\Pi_{\mathcal{D}}$ is the orthogonal projector onto $\mathcal{D}(m,k)$, and the vectors $\mathbf{x}^{(t+1)}$ are i.i.d. samples of the (unknown) distribution $p(\mathbf{x})$.

Similarly, the authors in [101, 113] go further and exploit the specific structure of sparse coding in the design of an optimization procedure tuned to large training problem, i.e., an online approach that processes the signals, one at a time, or in mini-batches. The technical details can be stepped as

- Fixed $\mathbf{D}$, compute sparse coefficients $\{\boldsymbol{\phi}^{(t)}\}$ using LARS/Lasso [50];
- Update the media matrices: $\mathbf{A}^{(t+1)} \leftarrow \mathbf{A}^{(t)} + \boldsymbol{\phi}^{(t)}\boldsymbol{\phi}^{(t)\top}$, $\mathbf{B}^{(t+1)} \leftarrow \mathbf{B}^{(t)} + \mathbf{x}^{(t)}\boldsymbol{\phi}^{(t)\top}$;
- Update $\mathbf{D}^{(t+1)} = \arg\min_{\mathbf{D}\in\mathcal{D}(m,k)} \frac{1}{(t)}\left(\frac{1}{2}\operatorname{Tr}(\mathbf{D}^\top\mathbf{D}\mathbf{A}^{(t+1)}) - \operatorname{Tr}(\mathbf{D}^\top\mathbf{B}^{(t+1)})\right)$.

Chapter 3

A Two-layer Representation Learning Framework

Already introduced in previous chapters, many computer vision problems, such as image denoising, inpainting and super-resolution, can be solved relying on the fact that an image/-patch admits a sparse representation over a given dictionary. We also recalled some popular methods in data-driven dictionary learning, i.e., the methods for learning dictionaries with the goal of data reconstruction. Form a perspective of representation learning, this chapter starts with a review of learning or constructing dictionary for specific computer vision task, such as classification. This chapter then gives a brief introduction of the main optimization problem of the dissertation, i.e., disentangle sparse coefficients for learning further representation of interest. Solving such an optimization problem requires that i) the differentiability of sparse representation with respect to a given dictionary, and ii) an efficient geometric gradient optimization algorithm for learning the parameters. By regarding the sparse representation as a locally differentiable function with respect to a specific dictionary, a generic form of the directional derivative of sparse representation with respect to the given dictionary is developed. Since the admissible sets of solutions to our main optimization problem are restricted on suitable matrix manifolds, we then recall some basic concepts of optimization on matrix manifolds.

3.1 Introduction

As introduced in Chapter 1, recent development in representation learning shows that appropriate data representations are a key to the success of many machine learning algorithms, cf. [5]. Namely, different representations of the data can disentangle different explanatory information with respect to the specific applications. Disentangling appropriate explanatory information or factors that describe the specific underlying structure is expected to facilitate the learning problem of interest. For example, to explore the evolution of the dynamic textured scenes, finding an appropriate spatio-temporal generative representation model plays a critical role on many successful video processing applications, cf. [114, 115]. For another example, finding suitable low dimensional image representations has demonstrated its prominent capability and convenience in images visualization, segmentation and classification, cf. [116, 117, 118].

The key challenge to image representations is attributed to the difficulty of disentangling

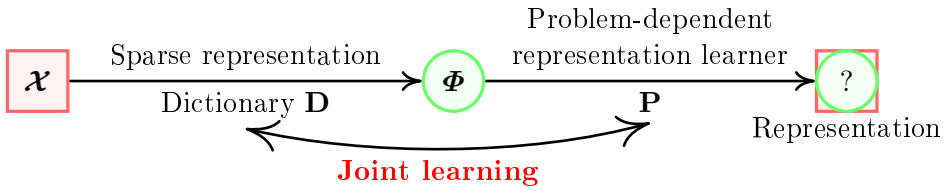
appropriate underlying factors that can exactly capture the useful information hidden in images or videos. This more or less relies on how the learning of representative features discover the underlying internal structure of observed data.

To address such a challenge, a common strategy is to adopt a mechanism of layer-wise disentangling to unwrap abstract factors of more fundamental features. In previous chapters, we have discussed that sparse representation is a powerful, fundamental instrument to leverage the underlying sparse structure of data for representing images [4, 3, 119, 120]. Given a collection of atoms, known as a dictionary, an image of interest is modeled as a linear combination of only a few atoms. The dictionary can be interpreted as underlying explanatory factors that are responsible for describing internal structures of the data. Such a model has led to a great success in many signal processing tasks, such as compressed sensing, signal reconstruction, denoising, inpainting and image super-resolution, as introduced in Chapter 1.2.

On the other hand, sparse representations of image data can also be interpreted as some disentangled features with respect to a predefined specific dictionary, i.e., explanatory information within the images to facilitate the learning task of interest, such as face recognition [3, 121], visual tracking [122] and object categorization [123, 27, 124, 42]. One popular approach is to deploy sparse representations directly as inputs to classifiers or other predictors. Often, performance of such an approach relies significantly on the construction of an appropriate sparsifying dictionary, which can be obtained by either randomly picking up some/all data points from a given set of training data [3, 121, 125, 119], or imposing specific structural constraints, such as group sparsity [42]. Nevertheless, it is known that these strategies can be numerically expensive, and is often not guaranteed with an optimal dictionary.

In order to cope with the aforementioned difficulty, a strategy of jointly learning both a dictionary and structured sparse representation is popularly adopted. For example, works in [28, 36, 37] propose to jointly learn a dictionary and structured sparse representation that promotes the locality of data. With the aim of acquiring a discriminative SR set, many approaches are developed focus on learning dictionary for maximizing the separation of sparse coefficients. The typical examples include, but not limited to, learning multiple class-specific (sub)-dictionaries [40, 126, 127], or learning one compact dictionary by imposing specific structural constraints, such as optimal Fisher discrimination criterion [41] and maximal mutual information [128]. A *similar* approach of adopting sparse representations in a classical expected risk minimization formulation leads to the so-called *task-driven dictionary learning* methods [26, 129, 130]. These approaches often involve jointly learning a sparsifying dictionary and a problem specific parameter. Well known loss functions are least squares loss function [108, 26, 109], logistic loss function [129, 26], and square hinge loss [130].

From a perspective of representation learning, aforementioned works can be considered as a single layer representation disentanglement to directly feed a specific predictor, such as a linear classifier. Based on such a great success, it is a logical conclusion that sparse representations can entangle the rich explanatory information of image data with respect to certain learning tasks. For example, the structure in sparse domain has been empirically observed that could make the hidden discriminative patterns more prominent and easier to



‡: $\boldsymbol{\mathcal{X}}$ can be descriptive features, such as SIFT

Figure 3.1: The proposed two-layer representation learning framework.

be captured [3, 121, 42]. This motivates the researchers to construct further representation learning mechanisms that allow to disentangle the underlying explanatory factors hidden in sparse representations of image data. For example, work in [125] demonstrates that applying a PCA directly on sparse representations of data is capable of enhancing performance in 3D visualization and clustering. Similar result in [131] also shows that low dimensional representations of sparse coefficients, obtained by a linear projection preserving pairwise inner products, can facilitate the task of classification. Finally, applying spectral clustering framework to sparse representations of data leads to the so-called *Sparse Subspace Clustering* method [119, 132], which enhances the performance in motion segmentation and face clustering. However, such representation leaning approaches are presented as a separated two-layer encoding scheme, i.e., the further disentangling instrument is separated from the layer of sparse coding. This may make it fails to incorporate such a two-layer representation learning paradigm into a deeper learning structure, i.e., multiple levels of representation, that could enable prominent disentanglibility of underlying factors that explains discrepancy underneath the observed image data [9, 5].

Therefore, it necessitates a generic joint learning paradigm that allows to construct further effective learning mechanisms to jointly disentangle a sparsifying dictionary and underlying factors hidden in image sparse representations. Such disentangled explanatory information or factors are expected to conveniently solve various computer vision problems. For example, explanatory information considered in this dissertation can be an underlying linear system that explains the dynamics of texture videos, or the similarity of image data points that explores the intrinsic structure of data. To achieve this goal, we construct a generic cost function for jointly learning a sparsifying dictionary and a problem-dependent representation learner. In this dissertation, such a problem-dependent representation learner is built according to specific computer vision problems, such as dynamic textures modeling and low dimensional image representation learning. The related learning framework is demonstrated in Fig. 3.1.

As shown in Fig. 3.1, the atoms in dictionary $\mathbf{D}$ can be interpreted as a set of fundamental underlying explanatory factors that are responsible for describing internal structures of the data. The further explanatory factors in matrix $\mathbf{P}$ are in charge of disentangling task-related information that is hidden in image sparse representations. The combination of $\mathbf{D}$ and $\mathbf{P}$ can be understood as layer-wise factors that explains variations behind the input data. At first,

by selecting a subset of atoms in $\mathbf{D}$, the sparse representation divides a raw input vector into the directions according to selected atoms. Such a layer of representation learning is expected to discover a few factors (the atoms in $\mathbf{D}$) that generate the raw input vector. These factors could capture several pieces of most fundamental information that underlie each input vector. Secondly, since $\mathbf{P}$ contains a set of factors that explains variation in sparse representation that are informative to the tasks of interest, the second layer in Fig. 3.1 transforms the sparse representation into a feature space that are spanned by only a few column vectors in $\mathbf{P}$. The representations in transformed space are expected to make the task-related information more prominent or task-driven predictors (e.g., classifiers) easier to be built. As an example, consider images of faces, and several factors: person identity, illumination and pose. The most variations of possible faces in pixel space can be explained by the aforementioned three factors. The atoms in $\mathbf{D}$ and $\mathbf{P}$ are responsible for identifying these factors and hence describing the variations of faces. The final representations disentangled by $\mathbf{D}$ and $\mathbf{P}$ are abstract features that are dominated by such factors. For instance, two persons of the same sex, age, and hair type will be distinguishable only by looking at their disentangled features.

In order to drive the whole learning mechanism (depicted in Fig. 3.1) forward on solving the final specific computer vision problem, the differentiability and convexity of the cost function with respect to the parameters play a crucial role. For addressing such a challenge, we consider the sparse coding problem by minimizing a quadratic reconstruction error with appropriate convex sparsity priors, such as elastic net prior and Kullback-Leibler (KL) divergence prior. In this way, sparse representation can be shown to be a locally differentiable function with respect to a given dictionary, and hence a generic form of the directional derivative of sparse representation with respect to the given dictionary is developed. On the other hand, we consider the set of solutions of our whole joint optimization problem is restricted to suitable differentiable manifolds. By leveraging such algorithmic benefits, we introduce an efficient geometric gradient optimization algorithm on the underlying Riemannian manifold.

3.2 The Main Optimization Problem

Already introduced in Chapter 3.1, the focus of this dissertation is on developing algorithmic framework that allows to disentangle the underlying explanatory factors hidden in sparse representations of image/video data. The developed algorithmic framework is depicted in Fig. 3.1. By regarding the whole learning process as an optimization problem that involves a dictionary $\mathbf{D}$ and a problem-dependent representation learner $\mathbf{P}$, in the following, we construct a generic cost function for jointly learning $\mathbf{D}$ and $\mathbf{P}$.

One key challenge for aforementioned joint learning process is relying on a set of implicit variables, i.e., the sparse coefficients $\boldsymbol{\Phi} := \{\boldsymbol{\phi}_i\}$ in Fig. 3.1, which bridges the given observations $\boldsymbol{\mathcal{X}} := \{\mathbf{x}_i\}$ and the final representations. Here, we treat each sparse vector $\boldsymbol{\phi}_i$ as an implicit function with respect to a dictionary $\mathbf{D}$. Formally, once a dictionary is given, according to Eq. (2.14), finding the sparse representation of a signal $\mathbf{x}_i \in \mathbb{R}^m$ is computed

as

$$\boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D}) := \underset{\boldsymbol{\phi}\in\mathbb{R}^k}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{x}_i - \mathbf{D}\boldsymbol{\phi}\|_2^2 + g(\boldsymbol{\phi}). \tag{3.1}$$

Here, the first term penalizes the reconstruction error of sparse representation, and the second term is a function that promotes the sparse structure of $\boldsymbol{\phi}_i$. Commonly, the function g is designed to be $g\colon \mathbb{R}^k \to [0, +\infty)$ such that $g(0) = 0$. There are many choices of function g in the literature, such as ℓ_0 norm and its variations, which have been introduced in Chapter 2.1.

It is reasonable and essentially critical to assure the sparse representations to be unique. The uniqueness can be achieved by choosing an appropriate convex function g. In such a way, by using a fixed dictionary $\mathbf{D}$, the solution $\boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D})$ as given in Eq. (3.1) can be considered as a function in $\mathbf{D}$, i.e., $\boldsymbol{\phi}^*_{\mathbf{D}}\colon \mathbb{R}^m \to \mathbb{R}^k$. In Section 3.3, we discuss that under the condition of uniqueness of the sparse solution to the problem in Eq. (3.1), the sparse representation $\boldsymbol{\phi}^*_{\mathbf{D}}$ is locally differentiable with respect to the dictionary $\mathbf{D}$.

Furthermore, let $\mathbf{X} := [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$ be a given collection of n training data points in $\mathbb{R}^m$. For each $\mathbf{x}_i$, the sparse codes $\boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D})$ is computed by Eq. (3.1). Hence, we construct a generic algorithmic framework to directly disentangle sparse coefficients for learning further representation of interest. Such an algorithmic framework can be formulated as a minimization problem

$$\begin{aligned} &\underset{\mathbf{D}\in\mathfrak{D},\mathbf{P}\in\mathfrak{P}}{\arg\min}\, \mathbb{E}_{\mathbf{x}} J(\mathbf{D},\mathbf{P}) \\ &J(\mathbf{D},\mathbf{P}) := f(\mathbf{P}, \boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D})) + \mu\gamma(\mathbf{D},\mathbf{P}), \end{aligned} \tag{3.2}$$

where $\mathfrak{D}$ and $\mathfrak{P}$ are some predefined admissible sets of solutions for the parameters $\mathbf{P}$ and $\mathbf{D}$, respectively. $f(\mathbf{P}, \boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D}))$ is a function to measure the loss of problem-dependent representation learner in sparse representation. Therein, $\gamma(\mathbf{D},\mathbf{P})$ denotes the differentiable regularization term on $\mathbf{D}$ and $\mathbf{P}$.

In this dissertation, we restrict admissible sets of solutions $(\mathbf{D},\mathbf{P})$ on suitable matrix manifolds $\mathcal{M} := \mathfrak{D} \times \mathfrak{P}$. For example, we restrict each column $\mathbf{d}_i \in \mathbb{R}^m$ of $\mathbf{D}$ to have unit norm, i.e., $\mathbf{D} \in \mathcal{S}(m,k)$. $\mathcal{S}(m,k)$ is defined as Oblique manifold as shown in Eq.(2.16). For another example, $\mathfrak{P}$ could define a set of orthogonal transformations, i.e., $\mathbf{P} \in St(l,m)$. Here, $St(l,m)$ denotes the Stiefel manifold

$$St(l,m) := \left\{ \mathbf{V} \in \mathbb{R}^{m\times l} | \mathbf{V}^\top\mathbf{V} = \mathbf{I}_l \right\}. \tag{3.3}$$

When the set of solutions of an optimization problem defined in a smooth manifold, geometric optimization techniques that exploit the underlying manifold structures of parameters can be employed to efficiently solve such an optimization problem. In Section 3.4, we will shortly review the general concepts of optimization on matrix manifolds.

In this dissertation, we confine ourselves to adopting the popular geometric gradient methods for solving the optimization problem (3.2). In order to develop a solvable geometric gradient algorithm to minimize the cost function J, the differentiability of J with respect

to $\mathbf{D}$ and $\mathbf{P}$ plays a crucial role. Given a set of sparse codes $\{\boldsymbol{\phi}^*_{\mathbf{x}_i}(\mathbf{D})\}$, f is a convex loss function that measures the loss of learning problem with respect to a problem-dependent model parameter $\mathbf{P} \in \mathfrak{P}$. In this dissertation, such a family of loss functions are expected to be globally differentiable with respect to $\mathbf{P}$.

Therefore, the key difficulty for solving the optimization problem (3.2) is the differentiability of cost function J with respect to a given dictionary $\mathbf{D}$. Hence, such a difficulty could be reduced to exploit the differentiability of sparse representation with respect to $\mathbf{D}$, which is implicitly included in the loss function f. By adopting appropriate convex sparsity priors, such as elastic net prior and KL divergence prior, sparse representation can be shown to be a locally differentiable function with respect to $\mathbf{D}$. More discussions will be delivered in Section 3.3.

3.3 Local Differentiability of Sparse Representation with Convex Sparsity Priors

We have discussed that the key requirement for developing a geometric algorithm to minimize the cost function J is the differentiability of sparse representation with respect to a given dictionary. The authors are aware of existing results on the matter of differentiability, such as [26, 130]. Unfortunately, these results are often difficult for further contributing to a sophisticated algorithm, such as a geometric conjugate gradient algorithm for minimizing the cost function J. In this section, we want to investigate the (local) differentiability of the sparse representation in the dictionary from the perspective of global analysis. Specifically, by considering the sparse coefficients as a element wise function to each atom $\{\mathbf{d}_i\}$, we discuss that the local differentiability of sparse representation with respect to a given dictionary is available. Research on such a problem relies on the suitable choice of the convex measures of $g(\cdot)$. In this section, we also present a number of popular convex sparsity measures for estimating $\boldsymbol{\phi}^*$.

3.3.1 Local Differentiability of Sparse Representation

Let $\boldsymbol{\phi}^* = [\varphi^*_1, \ldots, \varphi^*_k]^\top \in \mathbb{R}^k$ be a solution of the sparse representation of $\mathbf{x}$ for a given dictionary $\mathbf{D}$. Then we denote the set of indexes of non-zero entries of $\boldsymbol{\phi}^*$, known as the *support* of $\boldsymbol{\phi}^*$, by

$$\Lambda(\mathbf{x}, \mathbf{D}) := \{i \in \{1, \ldots, k\} | \varphi^*_i \neq 0\}, \tag{3.4}$$

and by $r := |\Lambda(\mathbf{x}, \mathbf{D})|$ the cardinality of $\Lambda(\mathbf{x}, \mathbf{D})$. By the fact of $g(\cdot)$ being convex, the sparse representation $\boldsymbol{\phi}^*$ is a global minimal of the cost function J. Moreover, it is intuitive and reasonable to assume that the representation $\boldsymbol{\phi}^*$ is *unique*.

We denote further by $\boldsymbol{\phi}^*_\Lambda = \{\varphi^*_j\}, j \in \Lambda$ and $\mathbf{D}_\Lambda \in \mathbb{R}^{m \times r}$ being the subset of $\mathbf{D}$, in which the index of atoms (columns) fall into the support Λ. By eliminating all inactive atoms, the vector $\boldsymbol{\phi}^*_\Lambda$ is the solution of the following restricted cost function

$$\min_{\boldsymbol{\phi}_\Lambda \in \mathbb{R}^r} \vartheta_\Lambda(\boldsymbol{\phi}_\Lambda) := \tfrac{1}{2}\|\mathbf{x} - \mathbf{D}_\Lambda \boldsymbol{\phi}_\Lambda\|_2^2 + g(\boldsymbol{\phi}_\Lambda). \tag{3.5}$$

Proposition 1. *Assume the sparse solution $\boldsymbol{\phi}_\Lambda^*$ to the problem (3.5) is unique. Then, $\boldsymbol{\phi}_\Lambda^*$ is locally differentiable at $\mathbf{D}_\Lambda$ and the derivative of $\boldsymbol{\phi}_\Lambda^*$ in direction $\boldsymbol{\mathcal{H}}$ has a close form expression as*

$$\mathrm{D}\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}} = \big(K(\mathbf{D}_\Lambda)\big)^{-1} \cdot \Big(\boldsymbol{\mathcal{H}}^\top \mathbf{x} - \big(\boldsymbol{\mathcal{H}}^\top \mathbf{D}_\Lambda + \mathbf{D}_\Lambda^\top \boldsymbol{\mathcal{H}}\big)\boldsymbol{\phi}_\Lambda\Big), \tag{3.6}$$

where $K(\mathbf{D}_\Lambda) := \mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda + \mathsf{H}g(\boldsymbol{\phi}_\Lambda)$ is the Hessian matrix of ϑ_Λ with $\mathsf{H}g(\boldsymbol{\phi}_\Lambda)$ being the Hessian matrix of g.

Proof 1 (Proof of **Proposition** 1). *The first derivative of the function $\vartheta_\Lambda(\boldsymbol{\phi}_\Lambda)$ with respect to $\boldsymbol{\phi}_\Lambda$ in direction $\boldsymbol{\xi} \in \mathbb{R}^r$*

$$\mathrm{D}\,\vartheta_\Lambda(\boldsymbol{\phi}_\Lambda)\boldsymbol{\xi} = -(\mathbf{x} - \mathbf{D}_\Lambda \boldsymbol{\phi}_\Lambda)^\top \mathbf{D}_\Lambda \boldsymbol{\xi} + \left(\nabla_g(\boldsymbol{\phi}_\Lambda)\right)^\top \boldsymbol{\xi}, \tag{3.7}$$

where $\nabla g(\boldsymbol{\phi}_\Lambda) \in \mathbb{R}^r$ denotes the Euclidean gradient of g at $\boldsymbol{\phi}_\Lambda$. Then by setting the first derivative of J to zero, we get the critical point condition for the unique sparse representation as

$$\nabla g(\boldsymbol{\phi}_\Lambda) = \mathbf{D}_\Lambda^\top (\mathbf{x} - \mathbf{D}_\Lambda \boldsymbol{\phi}_\Lambda). \tag{3.8}$$

Then, the critical point condition serves simply as an implicit function in $\mathbf{D}_\Lambda$, i.e., $\boldsymbol{\phi}_\Lambda \colon \mathcal{S}(m,k) \to \mathbb{R}^r$. Now, we take the derivative on the both sides of Eq. (3.8) with respect to $\mathbf{D}_\Lambda$ in direction $\boldsymbol{\mathcal{H}} \in T_{\mathbf{D}_\Lambda}\mathcal{S}(m,k)$ as

$$\begin{aligned} \mathrm{D}(\nabla_g(\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)))\boldsymbol{\mathcal{H}} = {} & \boldsymbol{\mathcal{H}}^\top(\mathbf{x} - \mathbf{D}_\Lambda \boldsymbol{\phi}_\Lambda) - \\ & - \mathbf{D}_\Lambda^\top \boldsymbol{\mathcal{H}} \boldsymbol{\phi}_\Lambda - \mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda \,\mathrm{D}\,\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}} \end{aligned} \tag{3.9}$$

Let's take a closer look at the left hand side of the equation, i.e., by the chain rule, we have

$$\mathrm{D}(\nabla_g(\boldsymbol{\phi}(\mathbf{D})))\boldsymbol{\mathcal{H}} = \mathsf{H}g(\boldsymbol{\phi}_\Lambda) \cdot \mathrm{D}\,\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}, \tag{3.10}$$

where $\mathsf{H}g(\boldsymbol{\phi}_\Lambda) \colon \mathbb{R}^r \times \mathbb{R}^r \to \mathbb{R}$ is the Hessian matrix of function g as a bilinear form. Substituting Eq. (3.10) into Eq. (3.9) leads to a linear equation in $\mathrm{D}\,\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}$ as

$$K(\mathbf{D}_\Lambda) \cdot \mathrm{D}\,\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{H}}^\top \mathbf{x} - \big(\boldsymbol{\mathcal{H}}^\top \mathbf{D}_\Lambda + \mathbf{D}_\Lambda^\top \boldsymbol{\mathcal{H}}\big)\boldsymbol{\phi}_\Lambda. \tag{3.11}$$

where $K(\mathbf{D}_\Lambda) := \mathbf{D}_\Lambda^\top \mathbf{D}_\Lambda + \mathsf{H}g(\boldsymbol{\phi}_\Lambda)$ is the Hessian matrix that is positive definite by assuming that $\boldsymbol{\phi}_\Lambda^*$ is unique. Thus, the derivative of $\boldsymbol{\phi}_\Lambda$ has a close form expression as Eq. (3.6).

The ability to compute $\mathrm{D}\boldsymbol{\phi}_\Lambda(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}$ leads to computing the directional derivative of J at $\mathbf{D}_\Lambda$. Some smooth solvers, like stochastic gradient descent (SGD) algorithm or conjugate gradient (CG) algorithm can be used to solve the minimization problem (3.2). We have discussed that the function g is required to be convex. In the following subsection, we will introduce some popular convex sparsity priors. More convex sparsity priors are described in [133, 134, 135].

3.3.2 Convex Sparsity Priors

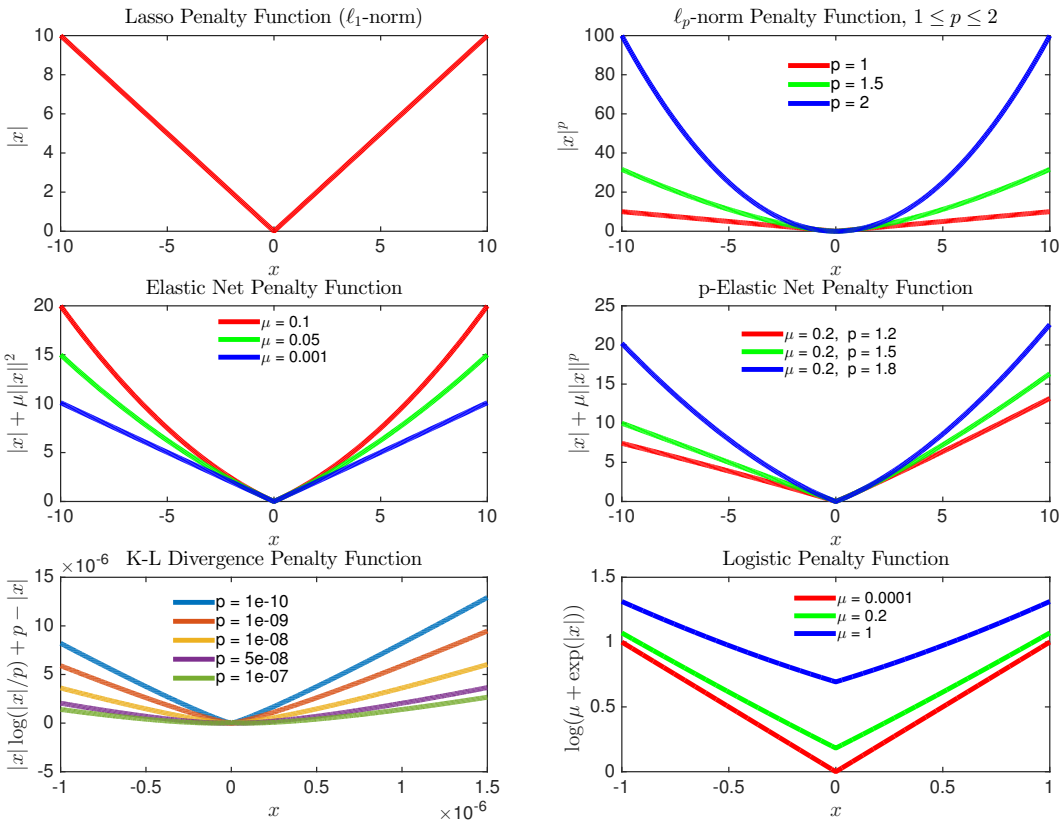


Figure 3.2: Convex sparsifying functions. This figure shows several commonly applied convex sparsifying functions with different parameter settings. For the KL divergence, we assume p and x carry the same sign.

Roughly speaking, the widely used convex sparsity regularizers can be divided into two categories.

The first class of regularizers is the usual ℓ_1-regularization term (also called lasso above) and its various extensions, namely, the family of ℓ_1-sparsity regularizers in this dissertation. Let us define such the family of ℓ_1-sparsity regularizers by component-wise addition

$$g(\boldsymbol{\phi}) = \sum_{i=1}^{k} \psi(|\varphi_i|) \tag{3.12}$$

with $\boldsymbol{\phi} = [\varphi_1, \ldots, \varphi_k] \in \mathbb{R}^k$, where $\psi(\varphi_i)$ is convex and satisfies

$$0 < a \leq \psi(z; \delta)' \leq b \tag{3.13}$$

for all $z \geq 0$, cf. [135]. Therein, $\psi(z; \delta)'$ denotes the directional derivative of ψ at z in direction δ, and a, b are two positive constants. The condition (3.13) is widely adopted that the

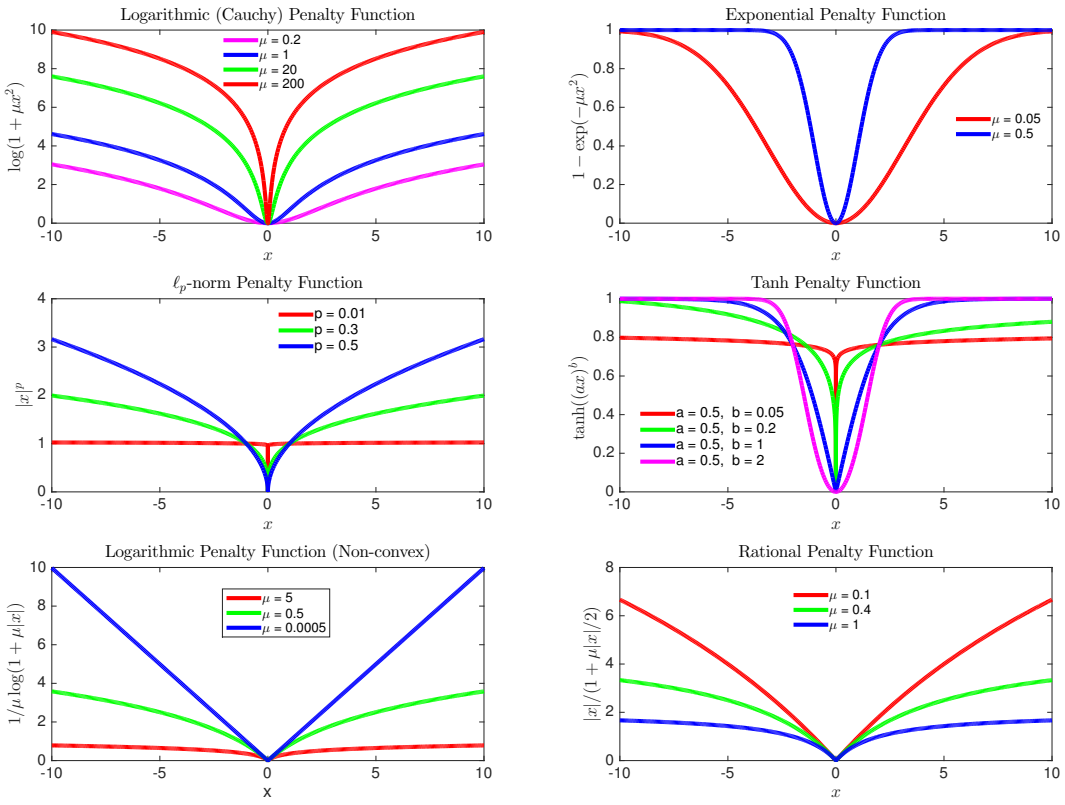


Figure 3.3: Non-convex sparsifying functions. This figure shows several commonly applied non-convex sparsifying functions with different parameter settings.

priors, i.e., the sparsity measure $\psi(\cdot)$ in Eq. (3.13), produce sparse solutions that are smooth on nonzero points with respect to small changes in $\mathbf{D}$ and $\mathbf{x}$. Several commonly applied ℓ_1-sparsity regularizers with different parameter settings are depicted in Fig. 3.2, while their definitions are recalled in Table 3.1. As for comparison, we also show the performance of several non-convex sparsifying functions, depicted in Fig. 3.3.

Table 3.1: Some commonly used ℓ_1-sparsity regularizers

The family of ℓ_1-sparsity regularizers	$\psi(\lvert\varphi_i\rvert)$
LARS/Lasso [50]	$\lambda\lvert\varphi_i\rvert$
Elastic Net [136]	$\lambda_1\lvert\varphi_i\rvert + \frac{\lambda_2}{2}\lvert\varphi_i\rvert^2, \lambda_1, \lambda_2 \in \mathbb{R}^+$
p-Elastic Net	$\lambda_1\lvert\varphi_i\rvert + \frac{\lambda_2}{2}\lvert\varphi_i\rvert^p, \lambda_1, \lambda_2 \in \mathbb{R}^+, p > 1$
Kullback-Leibler (KL)-divergence [95]	$\lambda\left(\lvert\varphi_i\rvert \log\frac{\lvert\varphi_i\rvert}{p_i} + p_i - \lvert\varphi_i\rvert\right), \lambda, p_i \in \mathbb{R}^+$
The logistic regularizer	$\lambda_1 \log\left(\lambda_2 + \exp(\lvert\varphi_i\rvert)\right), \lambda_1 \in \mathbb{R}^+, 0 < \lambda_2 < 1$

The family of ℓ_1-sparsity regularizers are effective for learning sparse, but the resulting optimization is challenging due to the non-differentiability at zero of sparse coefficient. To

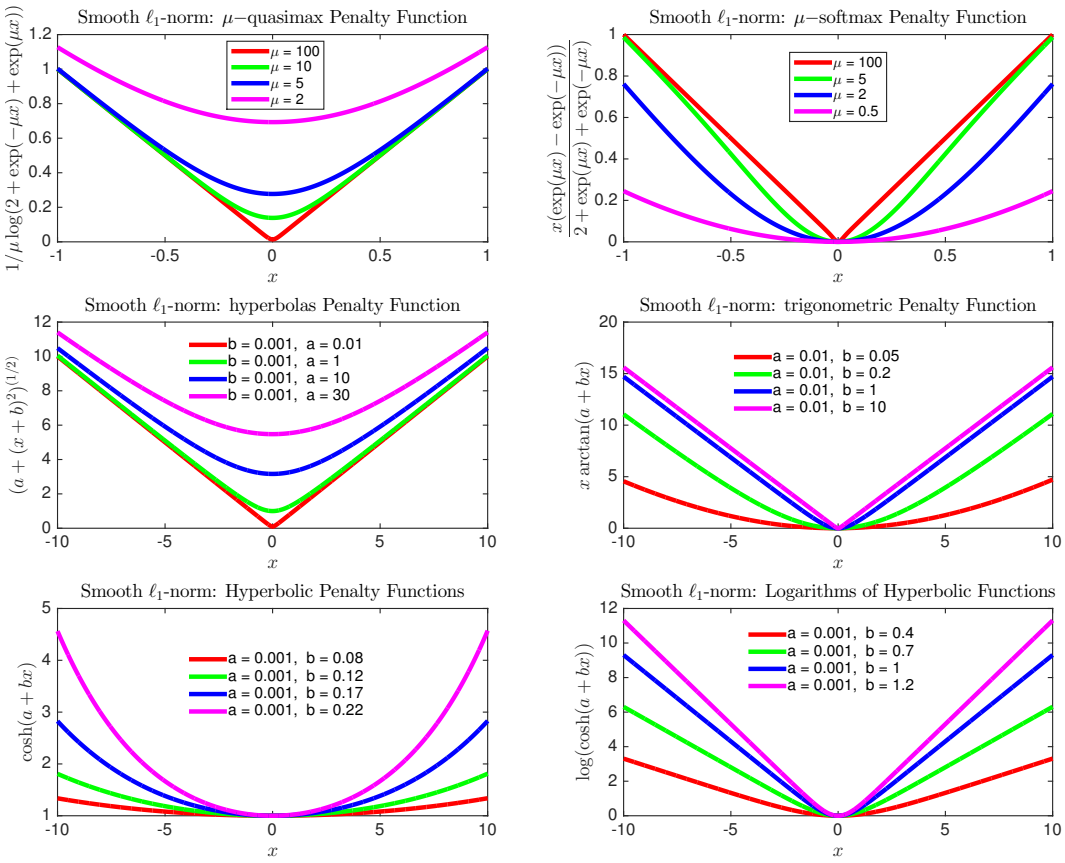


Figure 3.4: Smooth approximation of ℓ_1-norm.

fix such a drawback, the second set of convex sparsifying functions are proposed as a smooth function, i.e.,

$$g(\boldsymbol{\phi}) = \sum_{i=1}^{k} \psi(\varphi_i) \tag{3.14}$$

where $\psi(\cdot)$ is smooth for all $\{\varphi_i\}$, typically known as the smooth approximation to the ℓ_1-norm function. Fig. 3.4 plots some examples of such kind of functions and the corresponding definitions are listed in Table 3.2.

Base on such two categories of convex sparsity priors, it is easy to compute $\mathrm{D}\boldsymbol{\phi}_{\Lambda}(\mathbf{D}_{\Lambda})\boldsymbol{\mathcal{H}}$, based on the Proposition 1. The ability to compute such a directional derivative is one foundation for solving the minimization problem (3.2). In the following subsections, we introduce two convex sparsity priors and the corresponding conditions to guarantee the uniqueness of the sparse solution to the problem (3.1). These two convex sparsity priors have been applied by work presented in Chapter 3 and Chapter 4.

Table 3.2: Some commonly used sparsity regularizers that are smooth and convex.

Smooth and convex sparsity regularizers	$\psi(\varphi_i)$
Smoothing ℓ_1 (SL1) [137, 134]	$\frac{1}{\alpha}\log\left(2+\exp(-\alpha\varphi_i)+\exp(\alpha\varphi_i)\right)$
Smoothing ℓ_1 (SL1) [138]	$\frac{\varphi_i(\exp(\alpha\varphi_i)-\exp(-\alpha\varphi_i))}{2+\exp(\alpha\varphi_i)+\exp(-\alpha\varphi_i)}$
Trigonometric functions	$\varphi_i \arctan(\alpha+\beta\varphi_i)$
Logarithms of hyperbolic functions	$\log(\cosh(\alpha+\beta\varphi_i))$
Hyperbolas	$\sqrt{\alpha+(\varphi_i+\beta)^2}$

3.3.3 Lasso and Elastic Net

Lasso/Elastic Net is a popular way to replace ℓ_0-norm by its ℓ_1-norm convex relaxation. The Lasso problem has been introduced in Section 2.1. Elastic net is proposed as an extension of LARS/Lasso, but it considers the structure of sparse coefficients, cf. [136]. In Elastic net, one reads the prior distribution for the elements of each coefficient vector $\boldsymbol{\phi}$ as a mixture of Laplace and Gaussian in $\mathbb{R}$, i.e.,

$$p(\boldsymbol{\phi}) = \prod_{i=1}^{k} p(\varphi_i),\ p(\varphi_i) = C\cdot\exp\{-\lambda_1|\varphi_i| - \tfrac{\lambda_2}{2}\|\varphi_i\|^2\} \tag{3.15}$$

with $C, \lambda_1, \lambda_2 \in \mathbb{R}^+$. Note that, while $\lambda_2 = 0$, the elastic net problem (3.15) becomes the classical Lasso, cf. [45]. Let us denote by $\boldsymbol{\phi} \sim \mathcal{LG}(\mu, C, \lambda_1, \lambda_2)$ with $\mu = 0$ is location parameter.

Therefore, the MAP estimate of the coefficient over $p(\mathbf{x}_i|\mathbf{D})$, assuming a uniform prior on the dictionary, is the solution to the following optimization problem,

$$\boldsymbol{\phi}^* := \underset{\boldsymbol{\phi}\in\mathbb{R}^k}{\operatorname{argmin}} \tfrac{1}{2}\|\mathbf{x}-\mathbf{D}\boldsymbol{\phi}\|_2^2 + g(\boldsymbol{\phi}),\ \text{ with }\ g(\boldsymbol{\phi}) = \lambda_1\|\boldsymbol{\phi}\|_1 + \tfrac{\lambda_2}{2}\|\boldsymbol{\phi}\|_2^2, \tag{3.16}$$

where λ_1 and λ_2 are regularization parameters, which play an important role in ensuring stability and uniqueness of the solutions.

Let us define the set of indices of the non-zero entries of the solution $\boldsymbol{\phi}^* = [\varphi_1^*, \ldots, \varphi_k^*]^\top \in \mathbb{R}^k$ by $\Lambda := \{i \in \{1,\ldots,k\} | \varphi_i^* \neq 0\}$ and $r := |\Lambda|$. We compute

$$\begin{aligned} \mathrm{D}\, g(\boldsymbol{\phi}_\Lambda)\mathbf{h}_\Lambda &= \lambda_1 \mathbf{s}_\Lambda \mathbf{h}_\Lambda^\top + \lambda_2 \boldsymbol{\phi}_\Lambda \mathbf{h}_\Lambda^\top, \\ \mathrm{D}\left(\nabla_g(\boldsymbol{\phi}(\mathbf{D}_\Lambda))\right)\boldsymbol{\mathcal{H}}_\Lambda &= \lambda_2 \mathbf{I}_r\, \mathrm{D}\,\boldsymbol{\phi}(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}_\Lambda, \\ \mathsf{H}_g(\mathbf{D}_\Lambda) &= \lambda_2 \mathbf{I}_r, \end{aligned} \tag{3.17}$$

where $\mathbf{I}_r$ is the $r \times r$ identity matrix and $\mathbf{s}_\Lambda \in \{\pm 1\}^r$ carries the signs of $\boldsymbol{\phi}_\Lambda^*$.

Suppose $\mathbf{D} \in \mathfrak{D}$, let us denote $\mathbf{K} := \mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda + \lambda_2\mathbf{I}_r$ and $\mathbf{u} := \mathbf{D}_\Lambda^\top\mathbf{x} - \lambda_1\mathbf{s}_\Lambda$. Using Eq. (3.17) to substitute the $\mathsf{H}_g(\mathbf{D}_\Lambda)$ in Eq. (3.6), we have the first derivative of $\boldsymbol{\phi}_\mathbf{x}^*(\mathbf{D}_\Lambda)$ of Eq. (3.16) with respect to $\mathbf{D}_\Lambda$ in the direction $\boldsymbol{\mathcal{H}}_\Lambda$ is

$$\mathrm{D}\,\boldsymbol{\phi}_\mathbf{x}^*(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}_\Lambda = \mathbf{K}^{-1}\left(\boldsymbol{\mathcal{H}}_\Lambda^\top\mathbf{x} - (\mathbf{D}_\Lambda^\top\boldsymbol{\mathcal{H}}_\Lambda + \boldsymbol{\mathcal{H}}_\Lambda^\top\mathbf{D}_\Lambda)\boldsymbol{\phi}_\Lambda^*\right) \tag{3.18}$$

Such a prominent property leads to the framework of supervised task-driven dictionary learning, which is specifically dedicated to supervised learning problems, cf. [130, 26, 34].

Real world data and a simulation study in [136] show that the elastic net often outperforms the Lasso, while enjoying a similar sparsity of representation. In addition, the elastic net encourages a grouping effect, where strongly correlated predictors tend to be in (out) the model together. An efficient algorithm called LARS-EN is proposed for computing elastic net regularization paths efficiently, much like the LARS algorithm does for the Lasso.

3.3.4 Kullback-Leibler Divergence

To favor the sparsity and differentiability, one regularization function that produces this form of pseudo-sparsity is the Kullback-Leibler (KL)-divergence [95], which has been demonstrated its effectiveness on a wide variety of applications, e.g., the sparse autoencoder [139, 140] or deep neural networks [141].

Authors in [141] present that KL-divergence regularization is suited to sparse coding, and produce sparse solutions that integrate into a larger learning architecture, e.g., back-propagation neural networks. With the aim of approximating ℓ_p-norm ($p \leq 1$) to promote the sparsity, the KL-divergence $KL(\boldsymbol{\phi} \parallel p)$ is presented as a regularization function to replace ℓ_p-norm sparsity. A standard form of KL, served as the penalty $g(\boldsymbol{\phi})$ in Eq. (2.14), could be defined as

$$KL(\boldsymbol{\phi} \parallel p) = \lambda \sum_{i=1}^{k} \left[|\varphi_i| \log \frac{|\varphi_i|}{p_i} - |\varphi_i| + p_i \right] \tag{3.19}$$

and hence the corresponding loss function for learning sparse with given $\mathbf{D}$ is defined by

$$\boldsymbol{\phi}^*_{\mathbf{x}}(\mathbf{D}) = \underset{\phi \in \mathbb{R}^k}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{x} - f(\mathbf{D}\boldsymbol{\phi})\|^2 + KL(\boldsymbol{\phi} \parallel p). \tag{3.20}$$

The derivative of the gradient of the KL-divergence prior with respect to $\mathbf{D}_\Lambda$ is the product of a Hessian matrix and $\mathrm{D}\,\boldsymbol{\phi}(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}_\Lambda$, i.e.,

$$\mathrm{D}\left(\nabla_g(\boldsymbol{\phi}(\mathbf{D}_\Lambda))\right)\boldsymbol{\mathcal{H}}_\Lambda = \lambda \mathsf{H}_g(\mathbf{D}_\Lambda)\,\mathrm{D}\,\boldsymbol{\phi}(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}_\Lambda, \tag{3.21}$$

with $\mathsf{H}_g(\mathbf{D}_\Lambda) = \operatorname{diag}(\frac{\lambda}{\boldsymbol{\phi}_\Lambda})$ being a diagonal matrix whose diagonal entries are $\frac{\lambda}{\boldsymbol{\phi}_j}, j \in \Lambda$. By taking Eq. (3.21) in Eq. (3.6), we have

$$\mathrm{D}\,\boldsymbol{\phi}^*_{\mathbf{x}}(\mathbf{D}_\Lambda) \cdot \boldsymbol{\mathcal{H}}_\Lambda = -\big(\mathbf{D}_\Lambda{}^\top \mathbf{D}_\Lambda + \operatorname{diag}(\frac{\lambda}{\boldsymbol{\phi}_\Lambda})\big)^{-1} \left(\mathbf{D}_\Lambda^\top \boldsymbol{\mathcal{H}}_\Lambda \boldsymbol{\phi}_\Lambda + \boldsymbol{\mathcal{H}}_\Lambda(\mathbf{D}_\Lambda \boldsymbol{\phi}_\Lambda - \mathbf{x}) \right). \tag{3.22}$$

Moreover, the results in [141] shows that the learned $\boldsymbol{\phi}$ using KL-regularization were more useful for prediction than those produced with ℓ_1 regularization (the Lasso in [99]).

3.4 Resolving the Main Problem using Geometric Optimization

Recalling that the solutions of the constrained optimization problem (3.2) are defined on a product manifold $\mathcal{M} := \mathfrak{D} \times \mathfrak{P}$, in this section, we explain how such a differentiable cost function can be optimized by so-called geometric optimization method.

In the past decades, optimization on matrix manifolds has drawn more attention since it can reduce the dimension of optimization problems compared against solving the problems in their ambient Euclidean space. It also provides a good way for solving the optimization problems with matrix constraints, e.g., matrices on a unit sphere or Stiefel manifold. Many traditional optimization methods such as the steepest decent method, conjugate gradient method and Newton method have been extended to Riemannian manifolds or smooth manifolds, cf. [142, 143, 102]. Recently, authors in [25, 144, 145, 146] proposed to efficiently solve the sparse coding problem using the geometric conjugate gradient method with line search along geodesics on Riemannian manifolds. In this dissertation, we also focus on the gradient methods on smooth manifolds for solving data representation problems. In this section, we shortly recall some basic definitions and facts of differential geometry. For a detailed overview on optimization on matrix manifolds, we refer the interested reader to [102, 25].

3.4.1 Geometric Optimization

The essence of optimization problems is to find the maximum or minimum of a cost function. For example, considering a general unconstrained optimization problem, if its cost function J is defined in the Euclidean space $\mathbb{R}^n$, one can use conventional methods such as the steepest descent method, conjugate gradient method or Newton method to minimize this function, cf. [147]. However, many optimization problems that occurred in computer vision tasks, often consist of maximizing or minimizing a real function subject to fixed outside conditions or constraints, known as constrained optimization problem. In such a case, finding a closed form for the cost function being extremized is often difficult. One widely adopted strategy is to transform the constrained problem into an unconstrained form using the method of Lagrange multipliers or using a barrier penalty function [147]. This kind of approaches are efficient and allow one to solve the constrained optimization problem by taking advantage of the aforementioned conventional optimization techniques in Euclidean space. However, they merely treat constrained problem as a 'black box' and solves it using algebraic manipulation.

Instead of solving the optimization problem in their ambient Euclidean space, methods of solving minimization problem on smooth manifolds is developed and allow to exploit the underlying manifold structure of solutions, cf. [102]. By extending the concepts of vector addition in Euclidean space to the exponential map and parallel translation, minimization along lines to minimization along geodesics, partial differentiation to covariant differentiation, many conventional optimization techniques in Euclidean space can have their counterparts on smooth manifolds, which have been well studied in [148, 142, 149, 102]. These methods include steepest descent [148, 142, 149], conjugate gradient [150, 25], trust-region method [102, 151] and Newton's method [152, 102]. By appealing to these geometric gradient

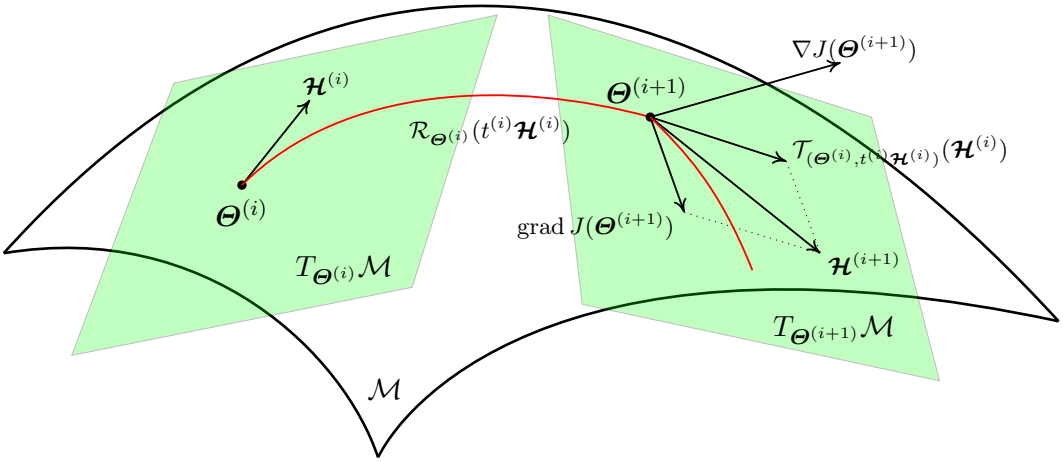


Figure 3.5: This figure shows two points $\boldsymbol{\Theta}^{(i)}$ and $\boldsymbol{\Theta}^{(i+1)}$ on a manifold $\mathcal{M}$ together with some required concepts on $\mathcal{M}$. Tangent space (green areas): $T_{\boldsymbol{\Theta}}\mathcal{M}$; The search direction (tangent vector) at $\boldsymbol{\Theta}$: $\boldsymbol{\mathcal{H}} \in T_{\boldsymbol{\Theta}}\mathcal{M}$; The Euclidean gradient $\nabla J(\boldsymbol{\Theta})$ and its projection onto the tangent space at $\boldsymbol{\Theta}$, called Riemannian gradient: grad $J(\boldsymbol{\Theta}) \in T_{\boldsymbol{\Theta}}\mathcal{M}$; *Retraction*: $\mathcal{R}_{\boldsymbol{\Theta}} \colon T_{\boldsymbol{\Theta}}\mathcal{M} \to \mathcal{M}$; Vector transport: $\mathcal{T}_{\boldsymbol{\Theta},t\boldsymbol{\mathcal{H}}} \colon T_{\boldsymbol{\Theta}}\mathcal{M} \to T_{\mathcal{R}_{\boldsymbol{\Theta}}(t\boldsymbol{\mathcal{H}})}\mathcal{M}$.

methods, this dissertation focuses on solving the optimization problem (3.2) formulated on a product manifold $\mathcal{M} := \mathfrak{D} \times \mathfrak{P}$, i.e., searching or selecting the admissible solutions on the sets that admitting the geometry structure of $\mathcal{M}$. This gives rise to some general concepts of Riemannian manifold, such as a Riemannian gradient, geodesic, Riemannian exponential mapping and Parallel transport.

Geometric Gradient Methods on Matrix Manifolds Let $\mathcal{M}$ be a smooth Riemannian submanifold of some Euclidean space and let $J \colon \mathcal{M} \to \mathbb{R}$ be the differentiable cost function. We consider the problem of finding

$$\arg\min_{\boldsymbol{\Theta} \in \mathcal{M}} J(\boldsymbol{\Theta}), \tag{3.23}$$

where $\mathcal{M} := \mathfrak{D} \times \mathfrak{P}$ in the optimization problem (3.2).

To compute the *Riemannian gradient* of J at every point $\boldsymbol{\Theta} \in \mathcal{M}$, the concept of tangent space $T_{\boldsymbol{\Theta}}\mathcal{M}$ should be defined. $T_{\boldsymbol{\Theta}}\mathcal{M}$ is a real vector space containing all possible directions that tangentially pass through $\boldsymbol{\Theta}$. An element $\boldsymbol{\Xi} \in T_{\boldsymbol{\Theta}}\mathcal{M}$ is called a tangent vector at $\boldsymbol{\Theta}$.

In order to characterize which direction of motion from $\boldsymbol{\Theta}$ produces the steepest increase in J, we further need a notion of length that applies to tangent vectors. This is done by endowing every tangent space $T_{\boldsymbol{\Theta}}\mathcal{M}$ with an inner product, denoted by $\langle \cdot, \cdot \rangle$. Such an inner product $\langle \cdot, \cdot \rangle$ induces a norm, denoted by $\| \cdot \|$, i.e., $\|\boldsymbol{\Xi}\| = \sqrt{\langle \boldsymbol{\Xi}, \boldsymbol{\Xi} \rangle}$ for any $\boldsymbol{\Xi} \in T_{\boldsymbol{\Theta}}\mathcal{M}$.

The *Riemannian gradient* of J at $\boldsymbol{\Theta}$ is an element of the tangent space $T_{\boldsymbol{\Theta}}\mathcal{M}$ that points

in the direction of steepest ascent of the cost function on the manifold. For the case where J is globally defined on the entire surrounding Euclidean space, the *Riemannian gradient* $\operatorname{grad} J(\boldsymbol{\Theta})$ is simply the orthogonal projection of the (standard) gradient $\nabla J(\boldsymbol{\Theta})$ onto the tangent space $T_{\boldsymbol{\Theta}}\mathcal{M}$, which reads as

$$\operatorname{grad} J(\boldsymbol{\Theta}) = \Pi_{T_{\boldsymbol{\Theta}}\mathcal{M}}(\nabla J(\boldsymbol{\Theta})). \tag{3.24}$$

A *geodesic* is a smooth curve $\Gamma(\boldsymbol{\Theta}, \boldsymbol{\Xi}, t)$ emanating from $\boldsymbol{\Theta}$ in the direction of $\boldsymbol{\Xi} \in T_{\boldsymbol{\Theta}}\mathcal{M}$ which locally describes the shortest path between two points on $\mathcal{M}$. Intuitively, it can be interpreted as the equivalence of a straight line in the manifold setting. The corresponding Riemannian exponential mapping, which maps a point from the tangent space to the manifold, is defined as

$$\exp_{\boldsymbol{\Theta}} \colon T_{\boldsymbol{\Theta}}\mathcal{M} \to \mathcal{M}, \quad \boldsymbol{\Xi} \mapsto \Gamma(\boldsymbol{\Theta}, \boldsymbol{\Xi}, 1). \tag{3.25}$$

In general, such an exponential map is only locally defined, that is, it only takes a small neighborhood of the origin at $T_{\boldsymbol{\Theta}}\mathcal{M}$, to a neighborhood of $\boldsymbol{\Theta}$ in the manifold.

As Riemannian exponential mappings of Eq. (3.25) are costly to compute in general, to deal with the cases of large scale datasets, we adopt an alternative approach, based on the concept of *Retraction* and its corresponding *Vector transport*. Some required concepts are depicted in Fig. 3.5 to alleviate the understanding.

A *Retraction* at $\boldsymbol{\Theta} \in \mathcal{M}$ is a smooth mapping from $T_{\boldsymbol{\Theta}}\mathcal{M}$ to $\mathcal{M}$ denoted by

$$\mathcal{R}_{\boldsymbol{\Theta}} \colon T_{\boldsymbol{\Theta}}\mathcal{M} \to \mathcal{M}$$

with a local rigidity condition [102], as $\mathcal{R}_{\boldsymbol{\Theta}}(\mathbf{0}) = \boldsymbol{\Theta}$ and $\mathrm{D}\,\mathcal{R}_{\boldsymbol{\Theta}}(\mathbf{0}) = \mathrm{id}_{T_{\boldsymbol{\Theta}}\mathcal{M}}$. Therein, $\mathrm{id}_{T_{\boldsymbol{\Theta}}\mathcal{M}}$ denotes the identity mapping on $T_{\boldsymbol{\Theta}}\mathcal{M}$. As shown in Fig. 3.5, for a tangent vector $\boldsymbol{\mathcal{H}}^{(i)} \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$, the curve

$$\Gamma_{\boldsymbol{\mathcal{H}}^{(i)}} \colon t^{(i)} \mapsto \mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})$$

satisfies $\mathrm{D}\,\Gamma_{\boldsymbol{\mathcal{H}}^{(i)}}(\mathbf{0}) = \boldsymbol{\mathcal{H}}^{(i)}$.

The *Vector transport* on $\mathcal{M}$ specifies how to transport a tangent vector $\boldsymbol{\Xi}$ from a point $\boldsymbol{\Theta}^{(i)} \in \mathcal{M}$ to a point $\boldsymbol{\Theta}^{(i+1)} \in \mathcal{M}$ along the curve $\mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})$, denoted by $\mathcal{T}_{(\boldsymbol{\Theta}^{(i)}, t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})}(\boldsymbol{\Xi})$.

The geometric optimization method reviewed in this section is based on iterating the following line search scheme. Given a current optimal point $\boldsymbol{\Theta}^{(i)}$ and a search direction $\boldsymbol{\mathcal{H}}^{(i)} \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$ at the i^{th} iteration, the step size $t^{(i)}$ which leads to sufficient decrease of J can be determined by finding the minimizer of

$$t^{(i)} = \underset{t \geq 0}{\arg\min}\, J(\mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t\boldsymbol{\mathcal{H}}^{(i)})). \tag{3.26}$$

Once $t^{(i)}$ has been determined, the new iterate is computed by

$$\boldsymbol{\Theta}^{(i+1)} = \mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t^{(i)}\boldsymbol{\mathcal{H}}^{(i)}). \tag{3.27}$$

Now, one straightforward approach to minimize J is to alternate Equations (3.24), (3.26), and (3.27) using

$$\boldsymbol{\mathcal{H}}^{(i)} = -\mathbf{G}^{(i)}$$

with the short hand notation $\mathbf{G}^{(i)} := \operatorname{grad} J(\boldsymbol{\Theta}^{(i)})$, which corresponds to the steepest descent on a Riemannian manifold. By leveraging the simplicity such a geometric steepest descent, in Chapter 4, it is used to solve the optimization problem for modeling dynamic textures.

However, as in standard optimization, steepest descent only has a linear rate of convergence. Therefore, another solver, called conjugate gradient (CG) method is often employed on a manifold, as it offers a superlinear rate of convergence, while still being applicable to large scale optimization problems with low computational complexity, e.g., in sparse recovery [144]. We refer to [102, 144] for further technical details for these computations. In Chapter 4, a geometric CG algorithm on $\mathcal{M}$ is presented.

In the CG method, the updated search direction $\boldsymbol{\mathcal{H}}^{(i+1)} \in T_{\boldsymbol{\Theta}^{(i+1)}}\mathcal{M}$ is a linear combination of the gradient $\mathbf{G}^{(i+1)} \in T_{\boldsymbol{\Theta}^{(i+1)}}\mathcal{M}$ and the previous search direction $\boldsymbol{\mathcal{H}}^{(i)} \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$. Since addition of vectors from different tangent spaces is not defined, we need to map $\boldsymbol{\mathcal{H}}^{(i)}$ from $T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$ to $T_{\boldsymbol{\Theta}^{(i+1)}}\mathcal{M}$. This is done by the so-called *Vector transport* $\mathcal{T}_{(\boldsymbol{\Theta}^{(i)}, t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})}(\boldsymbol{\Xi}^{(i)})$, which transports a tangent vector $\boldsymbol{\Xi}^{(i)} \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$ along the *curve* $\mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})$ to the tangent space $T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$. Then, the new CG search direction is computed by

$$\boldsymbol{\mathcal{H}}^{(i+1)} = -\mathbf{G}^{(i+1)} + \beta^{(i)} \mathcal{T}_{(\boldsymbol{\Theta}^{(i)}, t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})}(\boldsymbol{\mathcal{H}}^{(i)}) \tag{3.28}$$

with the direction parameter $\beta^{(i)}$, which is proposed in [153], as

$$\beta^{(i)} = \frac{\left\langle \mathbf{G}^{(i+1)}, \mathbf{G}^{(i+1)} - \mathbf{G}^{(i)} \right\rangle}{\left\langle \boldsymbol{\mathcal{H}}^{(i)}, \mathbf{G}^{(i)} \right\rangle}. \tag{3.29}$$

Note that the concrete formulations of above-mentioned concepts, i.e., *Riemannian gradient*, *Retraction* and *Vector transport*, are established according to the specific definition of each smooth manifold $\mathcal{M}$. Throughout the dissertation, $\mathcal{M} := \mathfrak{D} \times \mathfrak{P}$ with $\mathfrak{D}$ being a product of k unit spheres, i.e., $\mathcal{S}(m, k)$, defined in Eq. (2.16). The admissible set for $\mathfrak{P}$ is chosen as a common matrix set in $\mathbb{R}^{k \times k}$ in Chapter 4 or the set of all m-dimensional rank-l orthogonal projectors, called Grassmann manifold [154], i.e., $Gr(l, m)$, defined by

$$Gr(l, m) := \left\{ \mathbf{V}\mathbf{V}^\top | \mathbf{V} \in St(l, m) \right\}. \tag{3.30}$$

In the following two subsections, we quickly review some facts about the product of k unit spheres and the Grassmann manifold.

3.4.2 Geometry of the Product of k Unit Spheres

Given the tangent space of $\mathcal{S}(m, k)$ at $\mathbf{D}$ as

$$T_{\mathbf{D}}\mathcal{S}(m, k) := \{\boldsymbol{\mathcal{X}} \in \mathbb{R}^{m \times k} | \operatorname{ddiag}(\boldsymbol{\mathcal{X}}^\top \mathbf{D}) = 0\}, \tag{3.31}$$

the orthogonal projection of a matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times k}$ onto the tangent space $T_{\mathbf{D}}\mathcal{S}(m,k)$ with respect to the inner product $\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \operatorname{tr}(\boldsymbol{\mathcal{X}}^\top \boldsymbol{\mathcal{Y}})$ is given by

$$\mathit{\Pi}_{\mathbf{D}}(\boldsymbol{\Sigma}) := \boldsymbol{\Sigma} - \mathbf{D}\,\operatorname{ddiag}(\mathbf{D}^\top \boldsymbol{\Sigma}). \tag{3.32}$$

Therein, $\operatorname{ddiag}(\mathbf{Z})$ is the diagonal matrix whose entries on the diagonal are those of $\mathbf{Z}$.

Recalling that the $\mathcal{S}(m,k)$ is a Riemannian submanifold of a product of k unit spheres S^{m-1}, and let $\mathbf{d} \in S^{m-1}$ be a point on a sphere. Given a tangent vector $\mathbf{h} \in T_{\mathbf{d}}S^{m-1}$ at $\mathbf{d}$ and a fixed step size t_0, a *Retraction* is given by $\mathcal{R}_{\mathbf{d}} := \gamma(\mathbf{d}, \mathbf{h}, t = t_0)$ with a curve

$$\gamma(\mathbf{d}, \mathbf{h}, t) \colon = \tfrac{\mathbf{d}+t\mathbf{h}}{\|\mathbf{d}+t\mathbf{h}\|_2}.$$

Then the vector transportation along the direction $\mathbf{h} \in T_{\mathbf{d}}S^{m-1}$ at $\mathbf{d}$ for transporting $\boldsymbol{\xi} \in T_{\mathbf{d}}S^{m-1}$ is defined as

$$\tau(\boldsymbol{\xi}, \mathbf{d}, \mathbf{h}, t) \colon = \tfrac{1}{\|\mathbf{d}+t\mathbf{h}\|_2} \left(\mathbf{I}_n + \tfrac{(\mathbf{d}+t\mathbf{h})(\mathbf{d}+t\mathbf{h})^\top}{\|\mathbf{d}+t\mathbf{h}\|_2^2} \right) \boldsymbol{\xi}.$$

Using this, the *Retraction* through $\mathbf{D} \in \mathcal{S}(m,k)$ in the direction of $\boldsymbol{\mathcal{H}}_{\mathbf{D}} \in T_{\mathbf{D}}\mathcal{S}(m,k)$ is simply the combination of the *Retraction* for each column of $\mathbf{D}$ in the direction of the corresponding column of $\boldsymbol{\mathcal{H}}_{\mathbf{D}}$, i.e. $\mathcal{R}_{\mathbf{D}} := \mathit{\Gamma}_{\mathcal{S}}(\mathbf{D}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}, t_0)$ with $\boldsymbol{\mathcal{H}}_{\mathbf{D}} \colon = [\mathbf{h}_1, \cdots, \mathbf{h}_k]$ and a curve being

$$\mathit{\Gamma}_{\mathcal{S}}(\mathbf{D}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}, t) = [\gamma(\mathbf{d}_1, \mathbf{h}_1, t), \cdots, \gamma(\mathbf{d}_k, \mathbf{h}_k, t)]. \tag{3.33}$$

Accordingly, the vector transport of $\boldsymbol{\mathit{\Xi}}_{\mathbf{D}} \in T_{\mathbf{D}}\mathcal{S}$ with $\boldsymbol{\mathit{\Xi}}_{\mathbf{D}} \colon = [\boldsymbol{\xi}_1, \cdots, \boldsymbol{\xi}_k]$ along the curve $\mathit{\Gamma}_{\mathcal{S}}(\mathbf{D}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}, t)$ is given by

$$\mathcal{T}_{\mathcal{S}}(\boldsymbol{\mathit{\Xi}}_{\mathbf{D}}, \mathbf{D}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}, t) = [\tau(\boldsymbol{\xi}_1, \mathbf{d}_1, \mathbf{h}_1, t), \cdots, \tau(\boldsymbol{\xi}_k, \mathbf{d}_k, \mathbf{h}_k, t)]. \tag{3.34}$$

3.4.3 Geometry of Grassmann Manifold

Let us denote the set of all $k \times k$ skew-symmetric matrices by

$$\mathfrak{so}(k) := \left\{ \boldsymbol{Q} \in \mathbb{R}^{k\times k} \;\middle|\; \boldsymbol{\Omega} = -\boldsymbol{\Omega}^\top \right\}. \tag{3.35}$$

The tangent space of $Gr(l,k)$ at $\mathbf{P} \in Gr(l,k)$ is given by

$$T_{\mathbf{P}}Gr(l,k) := \{[\mathbf{P}, \boldsymbol{\Omega}] \mid \boldsymbol{\Omega} \in \mathfrak{so}(k)\} \tag{3.36}$$

with matrix commutator $[\mathbf{A}, \mathbf{B}] := \mathbf{AB} - \mathbf{BA}$. Let $\mathbf{P} \in Gr(l,k)$ and let $\boldsymbol{\mathit{\Xi}} \in T_{\mathbf{P}}Gr(l,k)$ be a tangent vector. We consider the Euclidean Riemannian metric on $Gr(l,k)$ induced by the embedding space of symmetric matrices, which is defined by the Frobenius inner product, i.e., $\langle \boldsymbol{\mathit{\Xi}}_1, \boldsymbol{\mathit{\Xi}}_2 \rangle := \operatorname{tr}(\boldsymbol{\mathit{\Xi}}_1^\top \boldsymbol{\mathit{\Xi}}_2)$ for all $\boldsymbol{\mathit{\Xi}}_1, \boldsymbol{\mathit{\Xi}}_2 \in T_{\mathbf{P}}Gr(l,k)$. The orthogonal projection of an arbitrary point $\boldsymbol{\Sigma} \in \mathbb{R}^{k\times k}$ onto the tangent space at $\mathbf{P}$ is

$$\mathit{\Pi}_{\mathbf{P}} \colon \mathbb{R}^{k\times k} \to T_{\mathbf{P}}Gr(l,k), \quad \boldsymbol{\Sigma} \mapsto [\mathbf{P}, [\mathbf{P}, \boldsymbol{\Sigma}_s]] \tag{3.37}$$

with $\boldsymbol{\Sigma}_s = \frac{1}{2}(\boldsymbol{\Sigma} + \boldsymbol{\Sigma}^\top)$ being the symmetric part of $\boldsymbol{\Sigma}$.

The unique geodesic $\gamma_{\mathbf{P},\boldsymbol{\Xi}}$ through $\mathbf{P}$ in direction $\boldsymbol{\Xi} \in T_{\mathbf{P}}Gr(l,k)$ is given by

$$\gamma_{\mathbf{P},\boldsymbol{\Xi}} \colon \mathbb{R} \to Gr(l,k), \quad \gamma_{\mathbf{P},\boldsymbol{\Xi}}(t) := \mathrm{e}^{t[\boldsymbol{\Xi},\mathbf{P}]}\,\mathbf{P}\,\mathrm{e}^{-t[\boldsymbol{\Xi},\mathbf{P}]}. \tag{3.38}$$

The parallel transport of $\boldsymbol{\eta} \in T_{\mathbf{P}}Gr(l,k)$ with respect to the Levi-Civita connection along the geodesic $\gamma_{\mathbf{P},\boldsymbol{\Xi}}(t)$ is given by

$$\boldsymbol{\eta}(t) = e^{t[\boldsymbol{\Xi},\mathbf{P}]}\boldsymbol{\eta}e^{-t[\boldsymbol{\Xi},\mathbf{P}]}. \tag{3.39}$$

Let

$$O(m) := \left\{ \mathbf{Q} \in \mathbb{R}^{k\times k} \;\middle|\; \mathbf{Q}^\top\mathbf{Q} = \mathbf{I}_k \right\}, \tag{3.40}$$

be the Lie group of all $k \times k$ orthogonal matrices. As $Gr(l,k)$ is a homogeneous space of $O(k)$, one can represent any point $\mathbf{P} \in Gr(l,k)$ by

$$\mathbf{P} = \mathbf{Q}\begin{bmatrix}\mathbf{I}_l & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathbf{Q}^\top \tag{3.41}$$

for suitable $\mathbf{Q} \in O(k)$, and can accordingly represent

$$[\boldsymbol{\Xi},\mathbf{P}] = \mathbf{Q}\begin{bmatrix}\mathbf{0} & -\mathbf{Z}^\top\\ \mathbf{Z} & \mathbf{0}\end{bmatrix}\mathbf{Q}^\top, \tag{3.42}$$

where $\mathbf{Z} \in \mathbb{R}^{(k-l)\times k}$, cf. [155].

As discussed before, geodesic and parallel transport are often more computationally demanding. In this work, by appealing to the general concepts of *Retraction* and its corresponding *Vector transport*, we present an approximation of *Riemannian exponential mappings* based on the *QR*-decomposition, cf. [156]. For this purpose, firstly we introduce a lemma from [156] without giving a proof.

Lemma 1. *The map*

$$\Upsilon_{\boldsymbol{\Omega}} : \mathbb{R} \to \mathcal{U}(k), \Upsilon_{\boldsymbol{\Omega}}(t) := (\mathbf{I}_k + t\boldsymbol{\Omega})_{\mathbf{Q}} \tag{3.43}$$

is smooth for all $\boldsymbol{\Omega} \in \mathfrak{so}(k)$.

Therein, $\mathcal{U}(k)$ denotes the set of unitary matrices in $\mathbb{R}^{k\times k}$, and $(\cdot)_Q$ is the unique QR decomposition of an invertible matrix, i.e. all diagonal entries of the upper triangular part of QR are positive.

Let $\boldsymbol{\Xi} \in T_{\mathbf{P}}Gr(l,k)$ and $\Upsilon_{[\boldsymbol{\Xi},\mathbf{P}]}(t)$ defined as in Eq. (3.43). The curve

$$\begin{aligned} \Gamma_{Gr}(\mathbf{P},\boldsymbol{\Xi},t) \colon \mathbb{R} &\to Gr(l,k),\\ t &\to \Upsilon_{[\boldsymbol{\Xi},\mathbf{P}]}(t)\mathbf{P}(\Upsilon_{[\boldsymbol{\Xi},\mathbf{P}]}(t))^\top \end{aligned} \tag{3.44}$$

is a second order approximation of the geodesic Eq. (3.38) around $\mathbf{P}$. Let $\boldsymbol{\mathcal{H}}_{\mathbf{P}} \in T_{\mathbf{P}}Gr(l,r)$

be a direction, then we define the following *retraction* on the Grassmann manifold as $\mathcal{R}_{\mathbf{P}} := \Gamma_{Gr}(\mathbf{P}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}, t = t_0)$, where t_0 denotes a fixed step size.

According to Eq. (3.39), the *Vector transport* of $\boldsymbol{\Xi}_{\mathbf{P}} \in T_{\mathbf{P}}Gr(l,k)$ along the curve $\Gamma_{Gr}(\mathbf{P}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}, t), \boldsymbol{\mathcal{H}}_{\mathbf{P}} \in T_{\mathbf{P}}Gr(l,k)$, is given by

$$\mathcal{T}_{Gr}(\boldsymbol{\Xi}_{\mathbf{P}}, \mathbf{P}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}, t) := \Upsilon_{[\boldsymbol{\mathcal{H}}_{\mathbf{P}}, \mathbf{P}]}(t) \boldsymbol{\Xi}_{\mathbf{P}} (\Upsilon_{[\boldsymbol{\mathcal{H}}_{\mathbf{P}}, \mathbf{P}]}(t))^{\top}. \tag{3.45}$$

Above we shortly reviewed the required concepts of optimization on matrix manifolds. In the Chapter 4 and the Chapter 5, we will discuss in detail on the concrete formulas and implementations for our optimization problems.

Chapter 4

Sparse Linear Dynamical Systems for Modeling Dynamic Textures

In Chapter 3, we have presented a two-layer representation learning framework. In this chapter, we discuss its first application for modeling and classifying videos. Video representation is an important and challenging task in the computer vision community. In this chapter, we consider the problem of modeling and classifying video sequences of dynamic scenes which could be modeled in a dynamic textures (DT) framework. At first, we assume that image frames of a moving scene can be modeled as a Markov random process. We propose a sparse coding framework, named Sparse Linear Dynamical Systems (*SLDS*), to model a video adaptively. By treating the sparse coefficients of image frames over a learned dictionary as the underlying "states", we learn an efficient and robust linear transition matrix between two adjacent frames of sparse events in time series. Hence, a dynamic scene sequence is represented by an appropriate transition matrix associated with a dictionary. In order to ensure the stability of *SLDS*, we impose several constraints on such transition matrix and dictionary. The developed framework is able to capture the dynamics of a moving scene by exploring both sparse properties and the temporal correlations of consecutive video frames. Moreover, such learned *SLDS* parameters can be used for various DT applications, such as DT synthesis and recognition. Experimental results demonstrate the strong competitiveness of our proposed *SLDS* approach in comparison with state-of-the-art video representation methods. Especially, it performs significantly better in dealing with DT synthesis and recognition on heavily corrupted data.

4.1 Introduction

Temporal or dynamic textures (DT) are video sequences that exhibit spatially repetitive and certain stationarity properties in time. This kind of sequences are typically videos of processes, such as moving water, smoke, swaying trees, moving clouds, or a flag blowing in the wind. Fig. 4.1 shows several examples of DT. Furthermore, consistent spatiotemporal motion, such as facial expressions, orderly pedestrian crowds, and vehicular traffic, can be seen as a generalization of DT. Study and analysis of DT attracts both theoretical and practical research efforts, such as video modeling [157, 114], DT segmentation [158, 115], video recognition [159], object tracking [160], saliency (e.g., emergency) detection [161] and video synthesis [114]. However, the continuous change in the shape and appearance of

a dynamic texture makes the application of traditional computer vision algorithms very challenging. Thus, finding an appropriate spatio-temporal generative representation model that can explore the evolution of the dynamic textured scenes, is the key to the success of many DT applications.

Figure 4.1: Eight examples of dynamic textures.

In the past several decades, various approaches have been proposed for modeling and synthesizing video sequences of dynamic textures [157, 162, 163, 114, 158, 164, 165]. Among them, one classical approach is to model dynamic scenes via the optical flow [157]. However, such methods require a certain degree of motion smoothness and parametric motion models. Non-smoothness, discontinuities, and noise inherent to rapidly varying, non-stationary DTs (e.g., fire) pose a challenge to develop optical flow based algorithms. Another technique, called particle filter [166], models the dynamical course of DTs as a Markov process. A reasonable assumption in DT modeling is that each observation is correlated to an underlying latent variable, or "state", and then derives the parameter transition operator between these states. Some approaches directly treat each observation as a state, and then focus on transitions between the observations in the time domain, cf. [163, 162, 164]. For instance, the work in [163] treats this transition as an approximation matrix of a target frame from its several nearest neighbors, and other methods construct a spatio-temporal autoregressive model (STAR) [162] or position affine operator for this transition [164]. However, natural images often have complex statistical structure with unknown distribution and are difficult to be explicitly parameterized. Therefore, some machine learning synthesis techniques, such as linear smooth regression, may not be directly used to model the transition of consecutive raw images.

Alternatively, representation-based models capture the intrinsic law and underlying structures of the observations by projecting the observations onto a low-dimensional representation space via feature extraction techniques, such as principle component analysis (PCA). G. Doretto et al. [167, 114] model the evolution of the dynamic textured scenes as a linear dynamical system (LDS) under a Gaussian noise assumption. As a popular method in dynamic textures, LDS and its derivative algorithms (e.g., kernel LDS) have been successfully used for various dynamic texture applications [114, 167, 168]. However, constraints are imposed on the types of motion and noise that can be modeled in LDS. For instance, it is sensitive to input variations due to various noise. Especially, it is vulnerable to non-Gaussian noise, such as missing data or occlusion of the dynamic scenes. Moreover, stability is also a challenging problem for LDS [169]. Additionally, another one possible challenge is that such a LDS may suffer a weak data reconstruction when observations's first several

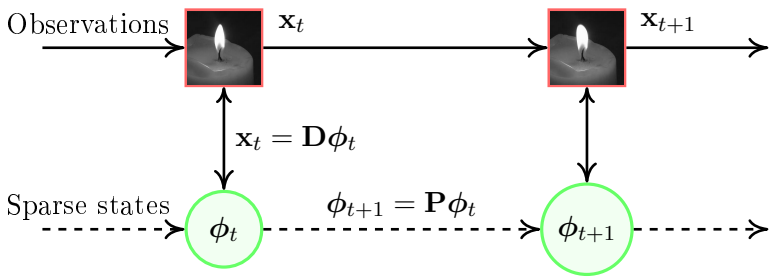


Figure 4.2: Pipeline of our proposed *SLDS* model. Therein, $\mathbf{x}_t$, $\boldsymbol{\phi}_t$, $\mathbf{D}$ and $\mathbf{P}$ denote the t^{th} observation, its hidden "state" or feature, the dictionary, and the state transition matrix, respectively.

largest singular values are not dominant. To tackle these challenges, the approach in this chapter is to explore an alternative method to model the DTs by appealing to the principle of sparsity. Instead of using the Principle Components (PCs) as the transition "states" in LDS, sparse coefficients over a learned dictionary are imposed as the underlying "states". In this way, the dynamic process of DTs exhibits a transition course of corresponding sparse events. These sparse events can be obtained via a recent technique on linear decomposition of data, called dictionary learning, which has been introduced in Chapter 2. Formally, these sparse representations $\boldsymbol{\phi} \in \mathbb{R}^k$ to a signal $\mathbf{x} \in \mathbb{R}^m$, can be written as Eq. (1.1). That is, the signal $\mathbf{x}$ can be sparsely represented only using a few elements from some dictionary $\mathbf{D}$.

Based on the sparse factorization of Eq. (1.1), our goal is to find a suitable and robust linear transition matrix between two adjacent frames of sparse representations in time series. With the aim of making the state transition stable and adapt to the sparse structures of image pairs from such adjacent frames, we enforce this linear transition matrix with moderate determinant and bounded largest eigenvalue. In this chapter, we start with a brief review of the dynamic textures model from the viewpoint of convex ℓ_2 optimization, and then deduce a combined regression associated with several regularizations for a joint process— "state extraction" and "state transition". Then we treat the solution of the above combined regression as a joint learning problem, i.e., jointly learning a sparsifying dictionary and a linear transition matrix, which can achieve two distinct yet tightly coupled tasks— efficiently reducing the dimensionality via sparse representation and robustly modeling the dynamic process. In the rest of the chapter, we refer to such a proposed model as Sparse Linear Dynamical Systems (*SLDS*). The pipeline is summarized in Fig. 4.2.

With the DT model at hand, this chapter also focuses on how to incorporate such a model into several video processing applications, such as synthesis, denoising and recognition on DT sequences. Note that, video synthesis and denoising could be achieved directly via basic *SLDS* model. Then, we are interested in the problem of categorization of DT sequences, i.e., identifying which class a query DT sequence belongs to. We propose a discriminative *SLDS* model that learns uniform *SLDS* parameters for each class, i.e., a dictionary associated with a transition matrix, which minimize the state transition error for intraclass DTs but maximize the state transition error for interclass DTs.

The rest of this chapter is organized as follows. In Section 4.2, we start with a brief review of linear dynamical systems. In Section 4.3, we construct a generic cost function for learning both the dictionary and the linear transition operator, and develop a geometric gradient descent algorithm on the underlying smooth manifold. Two classification algorithms are developed in Section 4.4. Numerical experiments on several applications of the proposed model are discussed in Section 4.5. Finally, conclusions and outlooks are given in Section 4.6.

4.2 Modeling Dynamical Textures using Linear Dynamic Systems

As mentioned earlier, one popular way to model the time series data is representing observed information about the past through a real-valued hidden state vector, known as state-space models (SSM), such as various descendants of either hidden Markov models (HMM) or stochastic LDS, cf. [170]. HMM represent information about the past of a sequence through a single discrete random variable – the hidden state. The prior probability distribution of this state is derived from the previous hidden state using a stochastic transition matrix. Knowing the state at any time makes the past, present and future observations statistically independent. The dependency between the present state vector and the previous state vector is specified through the dynamic equations of the system and the noise model. When these equations are linear and the noise model is Gaussian, the state-space model is also known as a LDS or Kalman filter model, in which the dynamics can transition in a discrete manner from one linear operating regime to another.

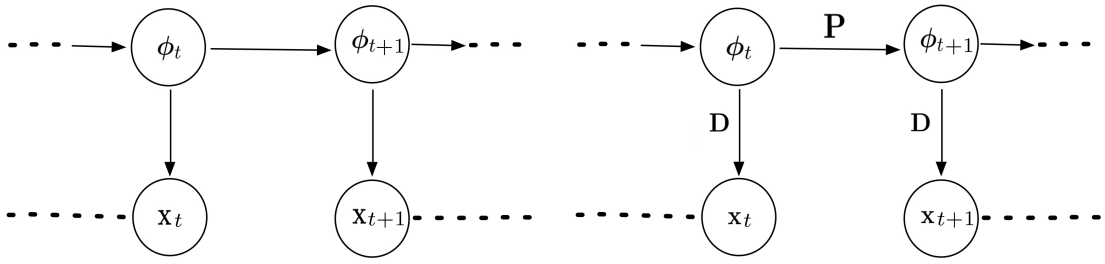
A state-space model defines a probability density over time series of real-valued observation vectors $\{\mathbf{x}_t\}$ by assuming that the observations were generated from a sequence of hidden state vectors $\{\boldsymbol{\phi}_t\}$. In particular, the state-space model specifies that given the hidden state vector at one time step, the corresponding observation vector is statistically independent from all other observation vectors, and that the hidden state vectors obey the Markov independence property. The joint probability for the sequences of states $\{\boldsymbol{\phi}_t\}$ and observations $\{\mathbf{x}_t\}$ can therefore be factored as:

$$\mathbb{P}(\{\boldsymbol{\phi}_t, \mathbf{x}_t\}) = \mathbb{P}(\boldsymbol{\phi}_1)\mathbb{P}(\mathbf{x}_1|\boldsymbol{\phi}_1) \prod_{t=2}^{T} \mathbb{P}(\boldsymbol{\phi}_t|\boldsymbol{\phi}_{t-1})\mathbb{P}(\mathbf{x}_t|\boldsymbol{\phi}_t). \tag{4.1}$$

The conditional independencies specified by Eq. (4.1) can be expressed graphically in the form of Fig. 4.3(a).

The simplest and most commonly used models of this kind assume that the transition and output functions are linear and time-invariant, and the distributions of the state and observation variables are multivariate Gaussian. We will use the term state-space model, i.e., LDS, to refer to this simple form of the model, see Fig. 4.3(b).

Let us denote a given sequence of $(T+1)$ frames by $\mathbf{X} := [\mathbf{x}_0, \ldots, \mathbf{x}_T] \in \mathbb{R}^{m\times(T+1)}$, where the time is indexed by $t = 0, 1, \ldots, T$. The evolution of a LDS is often described by the



(a) General SSM model (b) LDS model

Figure 4.3: The dynamical process of DTs exhibits a transition course of corresponding state events.

following two equations

$$\begin{cases} \boldsymbol{\phi}_{t+1} = \mathbf{P}\boldsymbol{\phi}_t + \mathbf{w}_t \\ \mathbf{x}_t = \mathbf{D}\boldsymbol{\phi}_t + \mathbf{v}_t, \end{cases} \tag{4.2}$$

where $\mathbf{x}_t \in \mathbb{R}^m$, $\boldsymbol{\phi}_t \in \mathbb{R}^k$, $\mathbf{w}_t \in \mathbb{R}^k$ and $\mathbf{v}_t \in \mathbb{R}^m$ denote the observation, its hidden state or feature, state noise, and observation noise, respectively. Therein, $\mathbf{w}_t$ and $\mathbf{v}_t$ are assumed to be zero-mean Gaussian with covariance matrix $\mathbf{Q}$. The system is described by the dynamics matrix $\mathbf{P} \in \mathbb{R}^{k\times k}$, and the modeling matrix $\mathbf{D} \in \mathbb{R}^{m\times k}$. Here we are interested in estimating the system parameters $\mathbf{P}$ and $\mathbf{D}$, together with the hidden states, given the sequence of observations $\mathbf{X}$.

The problem of learning the LDS in Eq. (4.2) can be considered as a coupled linear regression problem [169]. Let us denote $\boldsymbol{\Phi} = [\boldsymbol{\phi}_0, \ldots, \boldsymbol{\phi}_T] \in \mathbb{R}^{k\times(T+1)}$, $\boldsymbol{\Phi}_0 = [\boldsymbol{\phi}_0, \ldots, \boldsymbol{\phi}_{T-1}] \in \mathbb{R}^{k\times T}$, and $\boldsymbol{\Phi}_1 = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_T] \in \mathbb{R}^{k\times T}$. The system dynamics and modeling matrix are expected to be obtained by solving the following minimization problem,

$$\min_{\mathbf{P},\mathbf{D},\boldsymbol{\Phi}} \left\|\boldsymbol{\Phi}_1 - \mathbf{P}\boldsymbol{\Phi}_0\right\|_F^2, \quad s.t. \left\|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\right\|_F^2 \leq \varepsilon, \tag{4.3}$$

where ε is a small positive constant. Therein, $\|\cdot\|_F$ denotes the Frobenius norm of matrices.

Conventional LDS methods [114, 160, 169] often encode the observations as an *under-complete* representation over a dictionary with orthogonal columns, i.e., $\mathbf{X} := \mathbf{D}\boldsymbol{\Phi}$, with $\mathbf{D} \in St(m,k)$. Here, $St(m,k)$ denotes the Stiefel manifold defined in Eq. (3.3). Hence the solutions to the problem (4.3) relies on the so called singular value decomposition (SVD) of observations, i.e.,

$$\mathbf{X} \approx \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

with $\mathbf{U} \in St(m,k)$ and $\mathbf{V} \in St(k, T+1)$. Therein, $\boldsymbol{\Sigma} = \mathrm{diag}\{\delta_1, \cdots, \delta_k\}$ contains the first k largest non-negative singular values with $k < m$. Finally, one can obtain suboptimal estimates of $\mathbf{D}$ and $\boldsymbol{\Phi}$ as follows:

$$\tilde{\mathbf{D}} = \mathbf{U} \qquad \text{and} \qquad \tilde{\boldsymbol{\Phi}} = \boldsymbol{\Sigma}\mathbf{V}^\top. \tag{4.4}$$

The estimate of $\mathbf{P}$ is

$$\tilde{\mathbf{P}} = \boldsymbol{\Phi}_1 \boldsymbol{\Phi}_0^{\dagger}, \tag{4.5}$$

where † denotes the Moore-Penrose inverse.

4.3 Sparse Linear Dynamical Systems

In this section, we develop a joint learning framework for modeling dynamic textured sequences, i.e., jointly learning a dictionary and a transition matrix to represent a DT sequence.

4.3.1 A Dictionary Learning Model for Dynamic Scene

In our approach, we assume that all observations $\mathbf{x}_t$ admit a sparse representation with respect to an unknown dictionary $\mathbf{D} \in S(m,k)$, i.e.,

$$\mathbf{x}_t = \mathbf{D}\boldsymbol{\phi}_t, \qquad \text{for all } t = 0, 1, \ldots, T, \tag{4.6}$$

where $\boldsymbol{\phi}_t \in \mathbb{R}^k$ is sparse.

Finally, by adopting the common sparse coding framework to problem (4.3), we have the following minimization problem

$$\min_{\mathbf{P},\mathbf{D},\boldsymbol{\Phi}} \left\|\boldsymbol{\Phi}_1 - \mathbf{P}\boldsymbol{\Phi}_0\right\|_F^2 + \mu_1 \left\|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\right\|_F^2 + \mu_2 \|\boldsymbol{\Phi}\|_1, \tag{4.7}$$

with $\mathbf{D} \in \mathcal{S}(m,k)$, $\mathbf{P} \in \mathbb{R}^{k\times k}$, $\boldsymbol{\Phi} \in \mathbb{R}^{k\times(T+1)}$, $\mu_1 \in \mathbb{R}^+, \mu_2 \in \mathbb{R}^+$. The parameter $\mu_2 \in \mathbb{R}^+$ weighs the sparsity measurement against the two residual terms.

Solving the minimization problem as stated in Eq. (4.7) is a very challenging task. In this chapter, we employ an idea similar to *subspace identification methods* [171, 169], which treats the "state" as a function of $(\mathbf{P}, \mathbf{D})$.

Here, we confine ourselves to the sparse solution of a Lasso/Elastic Net problem, as Eq. (3.16), which is introduced in Chapter 3.3.3. Therein, $g(\boldsymbol{\phi}) = \lambda_1\|\boldsymbol{\phi}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\phi}\|_2^2$ with $\lambda_1 \gg \lambda_2 > 0$. A very small $\lambda_2 > 0$ is chosen to ensure stability and uniqueness of the sparse solution. By a slight abuse of notations, we denote by $\boldsymbol{\phi}^*_{\mathbf{x}_t}(\mathbf{D})\colon \mathbf{x}_t \mapsto \boldsymbol{\phi}_{\mathbf{x}_t}$ the sparse solutions to Eq. (3.1) with specific $\mathbf{D}$. We further denote by $\mathbf{X}_0 = [\mathbf{x}_0, \ldots, \mathbf{x}_{T-1}]$ and $\mathbf{X}_1 = [\mathbf{x}_1, \ldots, \mathbf{x}_T]$. In this way, by an abuse of notation, we define

$$\begin{aligned} \boldsymbol{\Phi}_0(\mathbf{D})\colon \mathcal{S}(m,k) &\to \mathbb{R}^{k\times T} \\ \mathbf{D} &\mapsto [\boldsymbol{\phi}^*_{\mathbf{x}_0}(\mathbf{D}), \ldots, \boldsymbol{\phi}^*_{\mathbf{x}_{T-1}}(\mathbf{D})]. \end{aligned} \tag{4.8}$$

In a similar way, $\boldsymbol{\Phi}_1(\mathbf{D})$ is defined by

$$\begin{aligned} \boldsymbol{\Phi}_1(\mathbf{D})\colon \mathcal{S}(m,k) &\to \mathbb{R}^{k\times T} \\ \mathbf{D} &\mapsto [\boldsymbol{\phi}^*_{\mathbf{x}_1}(\mathbf{D}), \ldots, \boldsymbol{\phi}^*_{\mathbf{x}_T}(\mathbf{D})]. \end{aligned} \tag{4.9}$$

By regarding such sparse events as the underlying "states" of observations, the dynamic

course of a moving scene can be modeled as a linear square regression problem with respect to a time-invariant transition matrix $\mathbf{P}$ and a dictionary $\mathbf{D}$, i.e.,

$$\begin{aligned} f\colon \mathbb{R}^{k\times k} \times \mathcal{S}(m,k) &\to \mathbb{R} \\ (\mathbf{P},\mathbf{D}) &\mapsto \tfrac{1}{2T}\left\|\boldsymbol{\Phi}_1(\mathbf{D}) - \mathbf{P}\boldsymbol{\Phi}_0(\mathbf{D})\right\|_F^2 . \end{aligned} \tag{4.10}$$

An illustration of such a process is described in Fig. 4.2.

The linear dynamic system referring to Eq. (4.10) may suffer from the three aspects as following: i) The learned linear transition matrix may not adapt to the distribution of sparse "states". ii) The instability of the learning system. iii) The high coherence of non-orthogonal atoms in dictionary may result in an ambiguity of sparse representation.

With the aim of building a solvable and stable learning procedure for the problem (4.10), in the following, we further regularize the problem by imposing several constraints on $\mathbf{P}$ and $\mathbf{D}$.

The Choice of Dictionary

Recalling the fact that the Eq. (3.18) exists if $\left(\mathbf{D}_\Lambda^\top\mathbf{D}_\Lambda + \lambda_2\mathbf{I}_r\right)^{-1}$ holds, i.e., $\mathbf{D}_\Lambda$ is full rank for all $r < m$ or the dictionary $\mathbf{D}$ is suitably incoherent with $\lambda_2 > 0$. A critical choice for $\mathbf{D}$ is a set of orthonormal atoms, i.e., $\mathbf{D} \in St(m,k)$ with $k < m$, and the problem (4.10) is simply solved by Eq. (4.4) and Eq. (4.5). Such a dictionary can efficiently project the observations into an low-dimensional orthogonal subspace, but it may yield a bad approximation for data reconstruction, i.e., $\mathcal{E}(\mathbf{x} \parallel \boldsymbol{\phi}, \mathbf{D}) := \left\|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\right\|_F^2$ may exceed the allowable limit. Let us denote by δ_i the i^{th} largest singular value of $\mathbf{X}$, and further define "Ratio of singular values" as $t_k = \sum_{i=1}^{k}\delta_i / \sum_{j=1}^{m}\delta_j$. An DT example is shown in Fig. (4.4). Fig. (4.4(g)) shows that it is difficult to choose a small low dimension k associated with a suitable t_k. Similarly, from Fig. (4.4(h)), it is easy to see that finding a small k often results in a high reconstruction error. For such a DT sequence, finding a suitable low dimensional subspace may not be achieved by Eq. (4.4) and Eq. (4.5). On the other hand, practically, finding a sparse representation over an orthogonal dictionary is often a challenge for some natural images.

Now, we relax the orthogonal constraint on $\mathbf{D}$ to a general $\mathbf{D} \in \mathcal{S}(m,k)$ under appropriate incoherence conditions. Let us define the mutual coherence of $\mathbf{D}$ as follows

$$\mu(\mathbf{D}) := \max_{1\le i<j\le k} |\mathbf{d}_i^\top \mathbf{d}_j|.$$

In order to prevent solution dictionaries from being highly coherent, we employ a log-barrier function on the scalar product of all dictionary columns to control the mutual coherence of the learned dictionary $\mathbf{D}$, cf. [25], i.e.,

$$\kappa(\mathbf{D}) := -\sum_{1\le i<j\le k} \log(1 - (\mathbf{d}_i^\top \mathbf{d}_j)^2). \tag{4.11}$$

(a) 1st frame

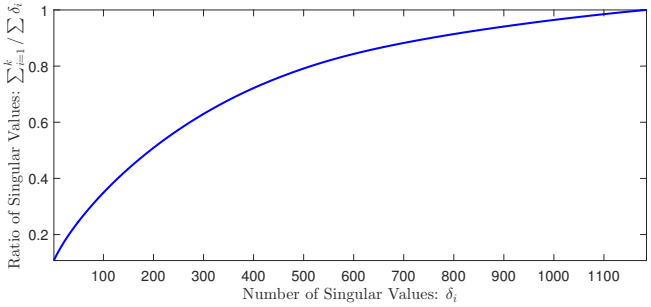
(b) 201st frame

(c) 401st frame

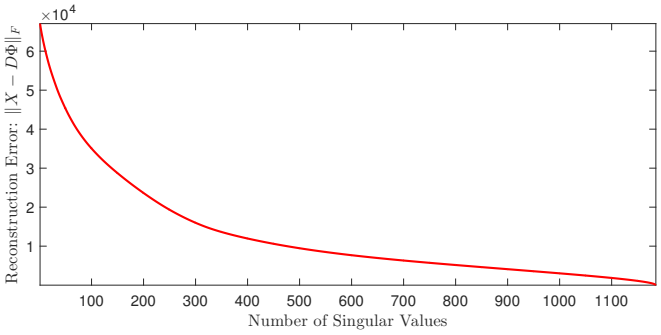
(d) 601st frame

(e) 801st frame

(f) 1001st frame



(g) Ratio of top k largest eigenvalues against all eigenvalues



(h) Reconstruction error against low dimensions

Figure 4.4: Two ducks on the surface of the lake, (Nr.645b410) from DynTex database [2]. (a)-(f) are six image examples in different time. (c) plots $t_k = \sum_{i=1}^{k} \delta_i / \sum_{j=1}^{m} \delta_j$ with increasing k. (d) depicts the reconstruction error $\left\|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\right\|_F$ with increasing k.

It is easy to see that a non-zero dictionary $\mathbf{D} \in \mathcal{S}(m,k)$ with $\kappa(\mathbf{D}) = 0$ indicates $\mathbf{D} \in St(m,k)$.

Stability Analysis

The stability is a desirable characteristic for LDS problems (4.2) and (4.3), especially when simulating long sequences from the system in order to generate representative data or infer stretches of missing values.

Recalling that we use the mixture of ℓ_1 norm and ℓ_2 norm to measure the sparsity, as Eq. (3.16) with $\lambda_2 \to 0^+$. $\lambda_2 \to 0^+$ indicates that the prior distribution for the elements of each coefficient vector $\boldsymbol{\phi}$ is zero-mean i.i.d. with standard Symmetric Laplace in $\mathbb{R}$, which could be defined as

$$p(\boldsymbol{\phi}) = \prod_{j=1}^{k} p(\varphi_j),\ p(\varphi_j) = \frac{\lambda_1}{2}\exp\{-\lambda_1|\varphi_j - \mu|\}. \tag{4.12}$$

where $\boldsymbol{\phi} = [\varphi_1, \ldots, \varphi_k]^\top \in \mathbb{R}^k$, $\lambda_1 \in \mathbb{R}^+$ is a scale parameter, and $\mu = 0$ is the location parameter. Let us denote by $\boldsymbol{\phi}_{\mathbf{x}} \sim \mathcal{L}(\mu, \lambda_1)$ the *Univariate Symmetric Laplace distribution* for $\boldsymbol{\phi}_{\mathbf{x}}$ with parameters $\mu = 0$ and $\lambda_1 \in \mathbb{R}^+$.

Let us consider the sparse representations matrices $\boldsymbol{\Phi}_0(\mathbf{D})$ and $\boldsymbol{\Phi}_1(\mathbf{D})$ of the data $\mathbf{X}_0$ and $\mathbf{X}_1$. The multidimensional extension of the generative model of Eq. (4.12) for vectors set $\boldsymbol{\Phi} := \boldsymbol{\Phi}(\mathbf{D})$ is straightforward. Here, we adopt the setting for multivariate Laplace (ML) distribution as shown in [172], which defines the formulation of ML distribution as a scale mixture of a multivariate Gaussian given by $\boldsymbol{\phi} = \sqrt{z}\boldsymbol{\Sigma}^{1/2}\boldsymbol{\theta}$. Therein, $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_k)$, $\boldsymbol{\Sigma}^{1/2} \in \mathbb{R}^{k\times k}$ is a positive definite, i.e., covariance matrix of $\boldsymbol{\Phi}$, and z is drawn from a univariate exponential distribution with probability density function (pdf) $p_Z(z) = \lambda_1 \exp(-\lambda_1 z)$. The integrated distribution of $\{\boldsymbol{\phi}_t\}$ over the prior distribution $p_Z(z)$ is given by

$$\begin{aligned} p_{\boldsymbol{\Phi}}(\boldsymbol{\phi}) &= \int_0^\infty p_{\boldsymbol{\Phi}|Z}(\boldsymbol{\phi}|Z = z)p_Z(z)dz \\ &= \int_0^\infty \frac{1}{(2\pi z)^{k/2}} \exp(-\frac{1}{2z}q(\boldsymbol{\phi}))p_Z(z)dz \\ &= \frac{2\mathbf{K}_{(k/2)-1}\left(\sqrt{\frac{2}{\lambda_1}q(\boldsymbol{\phi})}\right)}{(2\pi)^{(k/2)}\lambda_1\left(\sqrt{\frac{\lambda_1}{2}q(\boldsymbol{\phi})}\right)^{(k/2)-1}}, \end{aligned} \tag{4.13}$$

with $q(\boldsymbol{\phi}) = (\boldsymbol{\phi} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\phi} - \boldsymbol{\mu})$, and $\mathbf{K}_d(x)$ denotes the modified Bessel function of the second kind and order d, evaluated at $\boldsymbol{\phi}$. In what following, we will use the notation $\boldsymbol{\Phi} \sim \mathcal{ML}(\boldsymbol{\mu}, \lambda_1, \boldsymbol{\Sigma})$ to denote that $\boldsymbol{\Phi}$ is an ML distributed variable with parameters $\boldsymbol{\mu}$, λ_1, and $\boldsymbol{\Sigma}$. The model parameters of the Eq. (4.13) could be estimated using classical maximum-likelihood approach, e.g., iterative EM-type algorithm.

Let matrices $\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$ define the internal covariance structure of the variables of $\boldsymbol{\Phi}_0$ and

$\boldsymbol{\Phi}_1$, respectively. Now, let

$$\boldsymbol{\Phi}_1 = \mathbf{P}^\top \boldsymbol{\Phi}_0 + \boldsymbol{\zeta} \tag{4.14}$$

with $\boldsymbol{\zeta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}')$, be an arbitrary linear transformation of a $\mathcal{ML}(\boldsymbol{\mu}_1, \lambda_1, \boldsymbol{\Sigma}_1)$ random vector $\boldsymbol{\Phi}_0$, where $\mathbf{P} \in \mathbb{R}^{k\times k}$. The transformed variable $\boldsymbol{\Phi}_1$ admits $\boldsymbol{\Phi}_1 \sim \mathcal{ML}(\boldsymbol{\mu}_2, \lambda_1', \boldsymbol{\Sigma}_2)$ with

$$\begin{cases} \boldsymbol{\Sigma}_2 = \mathbf{P}^\top \boldsymbol{\Sigma}_1 \mathbf{P} |\det(\mathbf{P})|^{-(2/k)}, \\ \lambda_1' = \lambda_1 |\det(\mathbf{P})|^{(1/k)}, \\ \boldsymbol{\mu}_2 = \mathbf{P}^\top \boldsymbol{\mu}_1 + \boldsymbol{\mu}(\zeta), \end{cases} \tag{4.15}$$

where $\boldsymbol{\mu}(\boldsymbol{\zeta})$ is the mean vector of $\boldsymbol{\zeta}$ and assumed to be $\mathbf{0}$ in this chapter. Finding $\widetilde{\mathbf{A}}$ given $\boldsymbol{\Phi}_0$ and $\boldsymbol{\Phi}_1$ is triple approximation problems of (4.15).

In this chapter, we assume the length of sequence is sufficiently big. Thus, $\boldsymbol{\Phi}_0 := \boldsymbol{\Phi}_0(\mathbf{D})$ and $\boldsymbol{\Phi}_1 := \boldsymbol{\Phi}_1(\mathbf{D})$ share the same distribution as sparse coefficients set $\boldsymbol{\Phi}$. We assume $\boldsymbol{\Phi} \sim \mathcal{ML}(\boldsymbol{\mu}, \lambda_1, \boldsymbol{\Sigma})$ with $\boldsymbol{\mu} = \mathbf{0}$. From the definition of $\boldsymbol{\Phi}_0(\mathbf{D})$ and $\boldsymbol{\Phi}_1(\mathbf{D})$, it easily infers that $\boldsymbol{\Phi}_0(\mathbf{D}) \sim \mathcal{ML}(\boldsymbol{\mu}, \lambda_1, \boldsymbol{\Sigma})$, $\boldsymbol{\Phi}_1(\mathbf{D}) \sim \mathcal{ML}(\boldsymbol{\mu}, \lambda_1, \boldsymbol{\Sigma})$. Therefore, the linear transformation satisfies

$$\begin{cases} \boldsymbol{\Sigma} = \mathbf{P}^\top \boldsymbol{\Sigma} \mathbf{P} |\det(\mathbf{P})|^{-(2/k)}, \\ \lambda_1 = \lambda_1 |\det(\mathbf{P})|^{(1/k)}. \\ \boldsymbol{\mu} = \mathbf{P}^\top \boldsymbol{\mu}, \end{cases} \tag{4.16}$$

Eq. (4.16) shows that a stable transition process implies that $\det(\mathbf{P}) = 1$ and $\|\mathbf{P}^\top \mathbf{P}\|_2 = 1$ with $\|\cdot\|_2$ denoting the ℓ_2 norm of matrices.

Given data sequence $\{\boldsymbol{\phi}_t \in \mathbb{R}^k\}_{t=0}^T$, we hope to learn a stable linearity of expectation for Eq. (4.14), i.e., the latent variable fitting Eq. (4.16). Eq. (4.16) shows that a stable linear transformation requires a moderate $\det(\mathbf{P})$ and a moderate $\|\mathbf{P}^\top \mathbf{P}\|_2$. Given a square matrix $\mathbf{P} \in \mathbb{R}^{k\times k}$, it is known that $\|\mathbf{P}\|_F \geq \|\mathbf{P}\|_2$. Thus we can enforce constraints on $\mathbf{P}$ with the penalty functions

$$h(\mathbf{P}) = \frac{1}{4\log(k)} \Big(\log\big(\eta + \det(\mathbf{P}^\top \mathbf{P})\big)\Big)^2, \tag{4.17}$$

$$\rho(\mathbf{P}) = \frac{1}{2k^2} \|\mathbf{P}\|_F^2, \tag{4.18}$$

with $\eta \in (0, 1)$ being a small smoothing parameter. $h(\mathbf{P})$ is provided to void the worse case of $\det(\mathbf{P}^\top \mathbf{P})$ being exponentially big.

Let $\{\sigma_i\}_{i=1}^k$ denote the singular values of a transition matrix $\mathbf{P} \in \mathbb{R}^{k\times k}$ in decreasing order of magnitude, and $\sigma(\mathbf{P})$ denotes the largest one. It is known that $\|\mathbf{P}\|_F^2 = \sqrt{\sum_{i=1}^k \sigma_i^2} \geq \sigma(\mathbf{P})^2$. Thus, imposing a penalty as in Eq. (4.18) could result in a small $\sigma(\mathbf{P})$. On the other hand, $\mathbf{P}$ is expected to be full rank, and the Gram matrix $\mathbf{P}^\top \mathbf{P}$ is positive definite, which implies $\det(\mathbf{P}^\top \mathbf{P}) > 0$. Recalling that $\det(\mathbf{P}^\top \mathbf{P}) = \prod \sigma_i^2$ and $0 < \eta \ll 1$, thus the constraint term in Eq. (4.17) is imposed to restrict all singular values around 1. Such two constraints

are similar, but more critical, to the conventional work in [171, 169], which states that an LDS with dynamics matrix $\mathbf{P}$ is stable if all of $\mathbf{P}$'s eigenvalues have magnitude at most 1.

The Objective Function

By combining the regularizers discussed above, we construct the following cost function to jointly learn both the dictionary and the linear transition matrix, i.e.,

$$\begin{aligned} &J\colon \mathbb{R}^{k\times k} \times \mathcal{S}(m,k) \to \mathbb{R} \\ &(\mathbf{P},\mathbf{D}) \mapsto\ f(\mathbf{P},\mathbf{D}) + \gamma_1\rho(\mathbf{P}) + \gamma_2 h(\mathbf{P}) + \gamma_3\kappa(\mathbf{D}), \end{aligned} \tag{4.19}$$

where the weighting factors $\gamma_1, \gamma_2, \gamma_3 \in \mathbb{R}^+$ control the influence of three constraints on the final solution. Our experiments have verified that the regularizers $\rho(\mathbf{P})$ and $h(\mathbf{P})$ ensure solutions of the global cost function J defined in Eq. (4.19) to be self explanatory to the data, and guarantees stable performance towards the task of learning. On the other hand, such a learned $\mathbf{P}$ could capture the dynamic course of a moving scene and it is the key parameter for dynamic scenes synthesizing and classification, cf. [114, 167].

4.3.2 Optimization Algorithm for *SLDS*

In this section, we employ a gradient descent (GD) algorithm to minimize Eq. (4.19). Let $\mathcal{M} := \mathbb{R}^{k\times k} \times \mathcal{S}(m,k)$ be a product manifold of a Riemannian submanifold of $\mathbb{R}^{k\times k} \times \mathbb{R}^{m\times k}$, and let $J\colon \mathcal{M} \to \mathbb{R}$ be the differentiable cost function of Eq.(4.19). The general solution to optimization problem in Eq. (4.19) on matrix manifold, an element of $\mathcal{M}$, is denoted by $\boldsymbol{\Theta} \in \mathcal{M}, \boldsymbol{\Theta} := (\mathbf{P},\mathbf{D})$. For a detailed overview on optimization on matrix manifolds, we refer the interested reader to [102, 25]. Before introducing the technical details of the GD algorithm on $\mathcal{M}$, we first compute the Riemannian gradients of J with respect to $\mathbf{P}$ and $\mathbf{D}$.

Since all measures above on $\mathbf{P}$ and $\mathbf{D}$ are differentiable, thus, the current key challenge for Eq. (4.19) is the differentiability of $\boldsymbol{\phi}_{\mathbf{x}}(\mathbf{D})$ with respect to $\mathbf{D}$. Let us denote $\mathbf{K} := \mathbf{D}_{\Lambda}^{\top}\mathbf{D}_{\Lambda} + \lambda_2\mathbf{I}_k$ and $\mathbf{u} := \mathbf{D}_{\Lambda}^{\top}\mathbf{x} - \lambda_1\mathbf{s}_{\Lambda}$. The first derivative of $\boldsymbol{\phi}^*_{\mathbf{x}}(\mathbf{D})$ with respect to $\mathbf{D}$ in the direction $\boldsymbol{\mathcal{H}} \in T_{\mathbf{D}}\mathcal{S}(m,k)$ is

$$\mathcal{D}\,\boldsymbol{\phi}^*_{\mathbf{x}}(\mathbf{D})\boldsymbol{\mathcal{H}} = \mathbf{K}^{-1}\boldsymbol{\mathcal{H}}^{\top}\mathbf{x} - \mathbf{K}^{-1}(\mathbf{D}_{\Lambda}^{\top}\boldsymbol{\mathcal{H}} + \boldsymbol{\mathcal{H}}^{\top}\mathbf{D}_{\Lambda})\mathbf{K}^{-1}\mathbf{u}. \tag{4.20}$$

Therein, $T_{\mathbf{D}}\mathcal{S}(m,k)$ denotes the tangent space of $\mathcal{S}(m,k)$ at $\mathbf{D}$.

Then, by computing the first derivation of J at $(\mathbf{P},\mathbf{D})$ in tangent direction $(\boldsymbol{\mathcal{H}}_{\mathbf{P}}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}) \in T_{(\mathbf{P},\mathbf{D})}\mathcal{M}$, we get the Riemannian gradient of J at $(\mathbf{P},\mathbf{D})$ as

$$\operatorname{grad} J(\mathbf{P},\mathbf{D}) = \left(\nabla_J(\mathbf{P}), \mathit{\Pi}_{\mathbf{D}}\left(\nabla_J(\mathbf{D})\right)\right),$$

where $\nabla_J(\mathbf{P})$ and $\nabla_J(\mathbf{D})$ are the Euclidean gradients of J with respect to the two arguments, respectively. Therein, the map $\mathit{\Pi}_{\mathbf{D}}\colon \mathbb{R}^{m\times k} \to T_{\mathbf{D}}\mathcal{S}(m,k)$ is defined in Eq. (3.32).

Using the shorthand notation, for all $t = 0, 1, \ldots, T$, let $\mathit{\Lambda}_{t+1}$ be the support of nonzero entries of $\boldsymbol{\phi}_{t+1}(\mathbf{D})$, and denote $\mathbf{u}_{t+1} := \mathbf{D}_{\mathit{\Lambda}_{t+1}}^{\top}\mathbf{x}_{t+1} - \lambda_1\mathbf{s}_{\mathit{\Lambda}_{t+1}}$, $\Delta\boldsymbol{\phi}_{t+1} := \boldsymbol{\phi}_{t+1}(\mathbf{D}) - \mathbf{P}_{\mathit{\Lambda}_t}\boldsymbol{\phi}_t(\mathbf{D})$,

Algorithmus 1 : A *GD-SLDS* Algorithm.

Input : Given training set $\{\mathbf{x}_t \in \mathbb{R}^m\}_{t=0}^T$, parameters $\gamma_1, \gamma_2, \gamma_3$ and λ_1 ;
Output: $(\mathbf{P}^*, \mathbf{D}^*) \in \mathbb{R}^{k\times k} \times \mathcal{S}(m,k)$;
Step 1: Generate initialization for $(\mathbf{P}^{(0)}, \mathbf{D}^{(0)})$, and set $j = -1$;
Step 2: Set $j = j + 1$;
Step 3: Update sparse codes $\boldsymbol{\Phi}_{\mathbf{X}}(\mathbf{D}^{(j)})$ for each $\boldsymbol{\phi}_t(\mathbf{D}^{(j)})$ using Lasso/Elastic Net in Eq. (3.16);
Step 4: Update $\boldsymbol{\mathcal{H}}^{(j)} := (\boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(j)}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(j)}) \leftarrow -\operatorname{grad} J(\mathbf{P}^{(j)}, \mathbf{D}^{(j)})$;
Step 5: Find step size $t^{(j)}$ via a backtracking line search along retractions or geodesics, cf. [25, 144];
Step 5: Update $\mathbf{D}^{(j+1)} \leftarrow \mathit{\Gamma}_{\mathcal{S}}(\mathbf{D}^{(j)}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(j)}, t^{(j)})$, cf. Eq. (3.33);
Step 6: Update $\mathbf{P}^{(j+1)} \leftarrow t^{(j)}\mathbf{P}^{(j)} + \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(j)}$;
Step 7: If $\left\|\boldsymbol{\mathcal{H}}^{(j)}\right\|$ is small enough, stop. Otherwise, go to Step 2;

and $\mathbf{q}_t := \mathbf{u}_t \mathit{\Delta}\boldsymbol{\phi}_{t+1}^\top$, the Euclidean gradient $\nabla_J(\mathbf{D})$ of J with respect to $\mathbf{D}$ is

$$\begin{aligned}\nabla_J(\mathbf{D}) = &\sum_{t=0}^{T-1} \frac{1}{T}\mathcal{V}\{\Big(\mathbf{x}_{t+1}\mathit{\Delta}\boldsymbol{\phi}_{t+1}^\top - \mathbf{D}_{\mathit{\Lambda}_{t+1}}\mathbf{K}_{t+1}^{-1}(\mathbf{q}_t + \mathbf{q}_t^\top)\Big) \cdot \mathbf{K}_{t+1}^{-1}\} \\ &+ \frac{1}{T}\mathcal{V}\{\big(\mathbf{D}_{\mathit{\Lambda}_t}(\mathbf{K}_t)^{-1}(\mathbf{P}_{\mathit{\Lambda}_t}\mathbf{q}_t + \mathbf{q}_t^\top\mathbf{P}_{\mathit{\Lambda}_t}^\top) - \mathbf{x}_t(\mathit{\Delta}\boldsymbol{\phi}_{t+1})^\top\mathbf{P}_{\mathit{\Lambda}_t}\big)(\mathbf{K}_t)^{-1}\} + \gamma_3\nabla_\kappa(\mathbf{D})\end{aligned}$$

with

$$\nabla_\kappa(\mathbf{D}) = \mathbf{D}\sum_{1\le i<j\le k} \frac{2\mathbf{d}_i^\top\mathbf{d}_j}{1-(\mathbf{d}_i^\top\mathbf{d}_j)^2}(\mathbf{E}_{ij} + \mathbf{E}_{ji}) \tag{4.21}$$

being the gradient of the logarithmic barrier function Eq. (4.11). Therein, $\mathcal{V}\{\cdot\}$ denotes the full length vector of sparse coefficients $\{\cdot\}$.

Finally, the Euclidean gradient $\nabla_J(\mathbf{P})$ is computed as

$$\nabla_J(\mathbf{P}) = \sum_{t=0}^{T} \frac{1}{T}\boldsymbol{\phi}_{t+1}\mathit{\Delta}\boldsymbol{\phi}_{t+1}^\top + \gamma_1\nabla_\rho(\mathbf{P}) + \gamma_2\nabla_h(\mathbf{P}) \tag{4.22}$$

with

$$\begin{aligned}\nabla_h(\mathbf{P}) &= \frac{\eta}{\log(k)}\mathbf{P}(\eta\mathbf{P}^\top\mathbf{P})^{-1}, \\ \nabla_\rho(\mathbf{P}) &= \frac{1}{k^2}\mathbf{P}.\end{aligned}$$

Then, we denote by $\mathbf{G} := \operatorname{grad} J(\mathbf{P}, \mathbf{D})$, $\boldsymbol{\mathcal{H}} \in T_{(\mathbf{P},\mathbf{D})}\mathcal{M}$ the Riemannian gradient of J and

the gradient direction for update. Given $\dim \mathcal{S}(m,k) = k(m-1)$, we summarize a gradient descent algorithm for minimizing the function J as defined in Eq. (4.19), cf. Algorithm 1.

4.4 DTs Classification using *SLDS* Model

In the previous section, we proposed a generic regularized cost function to model the evolution of a temporal DT sequence, namely, *SLDS*. In this section, we present one application of the proposed *SLDS* model, to demonstrate its validity for DTs classification.

It is observed that the DTs from the same class exhibit similar spatial and temporal dynamics, which show strong dissimilarity for DTs from different classes, cf. [173, 174, 175, 176, 177]. In order to capture the similarity of dynamics of intraclass DTs, we propose to learn one unified *SLDS* model for all samples in such a class. At the same time, such learned class-wise *SLDS* parameters are expected to be against the dynamics of DTs outside the class. The key idea behind our development is to minimize the dissimilarity of dynamics of intraclass DTs, and simultaneously maximize the dissimilarity of dynamics of interclass DTs.

In this section, we consider to independently learn one *SLDS* classifier, i.e., one dictionary and one transition matrix, for each class. In what follows, at first, we introduce the *SLDS* classifier that suits for modeling each whole video sequence using a single *SLDS* model, which is called global *SLDS* classifier in the rest of the chapter. However, in the practical applications of visual recognition, one issue often challenges most sparse coding based algorithms, i.e., the linear system of Eq. (1.1) might become prohibitively expensive when the dimensionality of the raw image of input DT is huge. To address such a challenge, we then consider to learn one *SLDS* classifier for the small spatiotemporal patches extracted from the DT videos, which is simply called patch-based *SLDS* classifier.

4.4.1 Global *SLDS* Classifier

Let us denote by images set $\boldsymbol{\mathcal{Y}} = [\mathbf{X}_1, \ldots, \mathbf{X}_n] \in \mathbb{R}^{m \times n \times (T+1)}$ with each $\mathbf{X}_i$ denoting one DT sequence. The corresponding sparse coefficients are denoted by $\boldsymbol{\mathcal{X}} = [\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_n] \in \mathbb{R}^{k \times n \times (T+1)}$ with each $\boldsymbol{\Phi}_i$ being one sequence of $(T+1)$ sparse events. We now assume that each training sequence $\{\mathbf{X}_i \in \mathbb{R}^{m \times (T+1)}\}$ is associated with a indicator vector $z_i \in \mathbb{R}$, which indicates the corresponding class label. Let $c > 1$ denote the number of classes, n_j denotes the number of data samples in the j-th class with $n = \sum_{j=1}^{c} n_j$. Let $\mathcal{S}_{c_j}$ refer to the subset of $\{\mathbf{X}_i\}_{i=1}^{n}$ in the j^{th} class.

Let us denote $\mathbf{P}_j$ and $\mathbf{D}_j$ as parameters for modeling samples from the j^{th} class. Minimizing the dissimilarity of intraclass DTs can be read as an optimization problem to minimize

$$E_w^j = \frac{1}{2Tn_j} \sum_{i \in \mathcal{S}_{c_j}} \sum_{t=1}^{T} \left\| \boldsymbol{\phi}_{\mathbf{x}_{i,t}}(\mathbf{D}_j) - \mathbf{P}_j \boldsymbol{\phi}_{\mathbf{x}_{i,t-1}}(\mathbf{D}_j) \right\|_2^2 \tag{4.23}$$

with $\mathbf{x}_{i,t}$ being the $(t+1)^{\text{th}}$ frame of the i^{th} DT sequence $\mathbf{X}_i$. On the contrary, maximizing the dissimilarity of interclass DTs can be cast as an optimization problem to maximize

$$E_b^j = \frac{1}{2T(n-n_j)} \sum_{i \notin \mathcal{S}_{c_j}} \sum_{t=1}^{T} \left\| \boldsymbol{\phi}_{\mathbf{x}_{i,t}}(\mathbf{D}_j) - \mathbf{P}_j \boldsymbol{\phi}_{\mathbf{x}_{i,t-1}}(\mathbf{D}_j) \right\|_2^2 . \tag{4.24}$$

By combining Eq. (4.23) and Eq. (4.24), learning the j^{th}-class predictive model parameters $\{\mathbf{D}_j, \mathbf{P}_j\}$ could be formulated as the following minimization problem

$$(\mathbf{P}_j, \mathbf{D}_j) := \arg\min \{E_w^j - \gamma_4 E_b^j + \gamma_5 \kappa(\mathbf{D}_j)\} \tag{4.25}$$

with $\gamma_4, \gamma_5 \in \mathbb{R}^+$ being two tuning parameters. Specifically, $\gamma_5 \in \mathbb{R}^+$ is introduced to avoid the repeated atoms of $\mathbf{D}_j$.

Minimizing (4.25) could endow the model parameters ($\mathbf{P}_j$ and $\mathbf{D}_j$) with discrimination, but it does not take advantage of the sparse structure of "states" set $\boldsymbol{\mathcal{X}}$. Various works have verified that the sparse coefficients carry rich discriminative information, cf. [26, 146]. In order to explore the useful information from the sparse structure of $\boldsymbol{\mathcal{X}}$, in the following, we improve the classification model (4.25) by imposing a constraint on $\mathbf{P}$.

Sparse Transition Matrix

Let us focus on the problem of DTs classification, the stability for sparse state transition in (4.10) is not necessary. Our goal is to build an efficient mapping between the sparse coefficients of the current and previous images in time, and this mapping could capture the discriminative information hidden in sparse coefficients.

Works in [173, 175, 177] find that there exist strong spatial homogeneity and temporal periodicity in a single moving scene or motion, which implies that the DT patterns from one sequence are repetitive and often show a similar sparse structure over a suitable dictionary. On the other hand, the sparse events $\{\boldsymbol{\Phi}_i\}_{i=1}^{n_i}$ from the same class are often ideally assumed to share the similar essential sparse structure. Therefore, capturing such a similarity of sparse events of intraclass DTs provides a good way to help DTs classification, and hence the suitable choice of transition matrix $\mathbf{P}$ is sparse. The nonzero support of $\mathbf{P}$ is dominated by the support of nonzero entries in the sparse representations of consecutive images from intraclass DT sequences. In other words, the sparse structure of $\mathbf{P}^j$ is shared by sparse "states" of all DT sequences in the j-th class.

Here, we admit this assumption, and enforce the sparsity of each row of $\mathbf{P}$ as minimizing a ℓ_p norm with $0 \leq p \leq 1$. In this chapter, we use the following term to measure the overall sparsity of $\mathbf{P} := \{\mathbf{p}_{ij}\}$, i.e.,

$$r(\mathbf{P}) = \frac{1}{2k} \sum_{i=1}^{k} \sum_{j=1}^{k} \log(1 + \nu \mathbf{p}_{ij}^2) \tag{4.26}$$

with $0 < \nu < 1$ being a weighting parameter.

Essentially, for each DT sequence, the sparse transition matrix is expected to capture the dynamics of sparse events in time. Correspondingly, a learned highly row sparse matrix $\mathbf{P}_j$ could push the consecutive sparse vectors $\{\boldsymbol{\phi}_t, \boldsymbol{\phi}_{t+1}\}$ in j^{th} class tending to the similar permutation of nonzero entries. Therefore, it logically infers that such a sparse transition matrix could capture the similarity hidden in sparse representation sequences from the same class. Simultaneously, such a transition matrix may also promote the discrepancy of structures of interclass sparse events.

The Objective Function

By taking advantage of the sparse constraint on $\mathbf{P}$, i.e., Eq. (4.26), we modify the optimization problem (4.25) by minimizing

$$\begin{aligned} &\mathcal{L}\colon \mathcal{S}(m,k) \times \mathbb{R}^{k\times k} \to \mathbb{R}, \\ &\quad \mathcal{L}(\mathbf{P}_j, \mathbf{D}_j) := E_w^j - \gamma_4 E_b^j + \gamma_5 \kappa(\mathbf{D}_j) + \gamma_6 r(\mathbf{P}_j), \end{aligned} \tag{4.27}$$

where $\gamma_6 > 0$ is introduced to promote the sparse structure of $\mathbf{P}_j$. Our experiments have verified that an appropriately sparse $\mathbf{P}$ can significantly improve the results of DTs classification.

Classification

For the multi-class classification problem, i.e., $c > 2$, we use the one-against-all or one-against-one strategy to learn $\{\mathbf{P}_j, \mathbf{D}_j\}$. Let us consider one example which adopts the one-against-all strategy. When the training parameters $\{\mathbf{P}_j, \mathbf{D}_j\}_{j=1}^{c}$ are learned, classifying a test DT sequence $\mathbf{X} := \{\mathbf{x}_t\}_{t=0}^{T}$ can be formulated as finding

$$\text{identity}(\mathbf{X}) = \underset{j}{\operatorname{argmin}} \sum_{t=1}^{T} \left\| \boldsymbol{\phi}_{\mathbf{x}_t}(\mathbf{D}_j) - \mathbf{P}_j \boldsymbol{\phi}_{\mathbf{x}_{t-1}}(\mathbf{D}_j) \right\|_2^2,$$

for all $j = 1, 2, \cdots, c$.

4.4.2 Patch-based *SLDS* Classifier

Given a set DT sequences of j^{th} class with each sequence $\mathbf{X} \in \mathbb{R}^{a\times b\times (T+1)}$, $m = a \times b$, we divide it into non-overlapping spatiotemporal volumes of size $p \times p \times \tau$ where p represents the spatial size and τ represents the temporal size. The patch size is set according to the resolution of training sequences to ensure that we utilized the entire video sequence while extracting non-overlapping patches and not disregard any region. We randomly select n_j patches $\{\widetilde{\mathbf{X}}_i\}_{i=1}^{n_j}$ from each category for training its sub-dictionary $\mathbf{D}_j$ with the size of $p^2 \times k$. Therefore, for j^{th} class, we learn one dictionary $\mathbf{D}_j$ and n_j sparse transition matrices $\{\mathbf{P}_i\}_{i=1}^{n_j}$

by minimizing

$$\frac{1}{2(\tau-1)n_j}\sum_{i=1}^{n_j}\sum_{t=2}^{\tau}\left\|\boldsymbol{\phi}_{\widetilde{\mathbf{x}}_{i,t}}(\mathbf{D}_j)-\mathbf{P}_i\boldsymbol{\phi}_{\widetilde{\mathbf{x}}_{i,t-1}}(\mathbf{D}_j)\right\|_2^2+\gamma_7\kappa(\mathbf{D}_j)+\gamma_8 r(\mathbf{P}_j), \tag{4.28}$$

with $\gamma_7\in\mathbb{R}^+,\gamma_8\in\mathbb{R}^+$ and $\widetilde{\mathbf{x}}_{i,t}$ being the $(t+1)^{\text{th}}$ frame of the i^{th} patch sequence $\widetilde{\mathbf{X}}_i$. We shortly denote the j^{th} class *SLDS* parameters by $\mathcal{M}_j=(\mathbf{D}_j,\{\mathbf{P}\}_{i=1}^{n_j})$.

With the learned *SLDS* parameters $\{\mathcal{M}_j\}_{j=1}^{c}$ at hand, some standard classification methods can be employed. Here, $c\in\mathbb{Z}^+$ denotes the number of classes. In this section, the classification is performed by *SLDS* associated with the sparse representation-based classifier (SRC) [3, 178], called *SLDS-SRC*, which is discussed in detail as follows.

Before performing *SLDS-SRC*, we combine all the sub-dictionaries $\{\mathbf{D}_j\}_{j=1}^{c}$ as a shared dictionary $\mathbf{D}$ with the size of $p^2\times(ck)$.

Given a query DT sequence $\mathbf{X}$, we first divide it into N spatiotemporal patches $\widetilde{\mathbf{X}}_i:=\{\widetilde{\mathbf{x}}_{it}\}\in\mathbb{R}^{p^2\times\tau}$ and for each patch we obtain its sparse coefficients set $\widetilde{\boldsymbol{\Phi}}_i:=\{\widetilde{\boldsymbol{\phi}}_{it}\}\in\mathbb{R}^{(ck)\times\tau}$ via performing Eq. (3.16) with respect to $\mathbf{D}$. Let us denote an operator $\delta^j:\mathbb{R}^{ck}\to\mathbb{R}^{ck}$ be the characteristic function which selects the coefficients associated with the j^{th} class, cf. [3]. For our learned $\widetilde{\boldsymbol{\phi}}$, $\delta^j(\widetilde{\boldsymbol{\phi}})\in\mathbb{R}^{ck}$ denotes the sparse codes of class j, i.e., all entries are set to zero if they do not belong to class j. By using the sparse codes from j^{th} class, we calculate the reconstruction error via

$$R_j(\widetilde{\mathbf{X}},\mathbf{D})=\frac{1}{N\tau}\sum_{i=1}^{N}\sum_{t=1}^{\tau}\|\widetilde{\mathbf{x}}_{it}-\mathbf{D}\delta^j(\widetilde{\boldsymbol{\phi}}_{it})\|_2. \tag{4.29}$$

Similarly, by using $\{\mathbf{P}\}_{i=1}^{n_j}$ from j^{th} class, we approximate the temporal dynamic process by solving the following optimization problem

$$\min_{\boldsymbol{\alpha}}\sum_{i=1}^{N}\sum_{t=1}^{\tau-1}\|\widetilde{\boldsymbol{\phi}}_{i(t+1)}-\sum_{\iota=1}^{L}\boldsymbol{\alpha}_{i\iota}\mathbf{P}_i\widetilde{\boldsymbol{\phi}}_{it}\|+\lambda\sum_{i=1}^{N}\|\boldsymbol{\alpha}_i\|_1$$

with $L=cn_j$. In our experiments, in order to reduce the computation cost, we set $L=c(n_j')$ with $n_j'\in\mathbb{Z}^+$, $n_j'<n_j$.

With the sparse vectors $\{\boldsymbol{\alpha}_{i\iota}\}_{i=1,\iota=1}^{i=N,\iota=L}$ at hand, we calculate the approximate error by

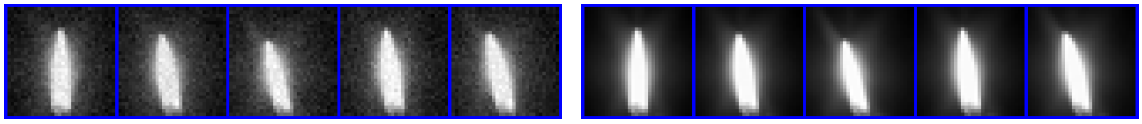
$$R_j(\widetilde{\boldsymbol{\Phi}},\mathbf{P})=\frac{1}{N(\tau-1)}\sum_{i=1}^{N}\sum_{t=1}^{\tau-1}\|\widetilde{\boldsymbol{\phi}}_{i(t+1)}-\sum_{i=1}^{L}\delta_j(\boldsymbol{\alpha}_{i\iota})\mathbf{P}_i\widetilde{\boldsymbol{\phi}}_{it}\|_2.$$

Therein, $\delta_j(\boldsymbol{\alpha}_{i\iota})$ keeps the value of $\boldsymbol{\alpha}_{i\iota}$ if $\mathbf{P}_i$ belongs to j^{th} class, $\delta_j(\boldsymbol{\alpha}_{i\iota})=0$ otherwise.

Hence, we classify a query DT sequence $\mathbf{X}$ as follows

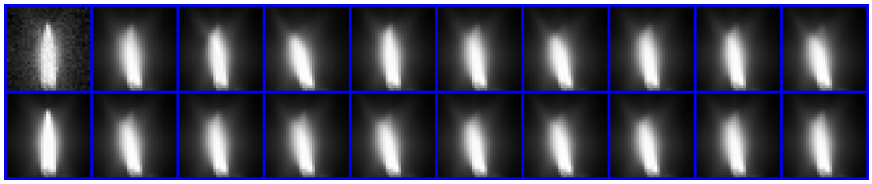
$$\text{identity}(\mathbf{X})=\underset{j}{\text{argmin}}\, R_j(\widetilde{\mathbf{X}},\mathbf{D})+\gamma R_j(\widetilde{\boldsymbol{\Phi}},\mathbf{P}),$$

where γ is a tuning parameter to balance the two residual terms. We set $\gamma = 1$ in our following experiments. Therein, $\widetilde{\mathbf{X}}$ and $\widetilde{\boldsymbol{\Phi}}$ denote the patches set of $\mathbf{X}$ and the corresponding sparse coefficients set, respectively.

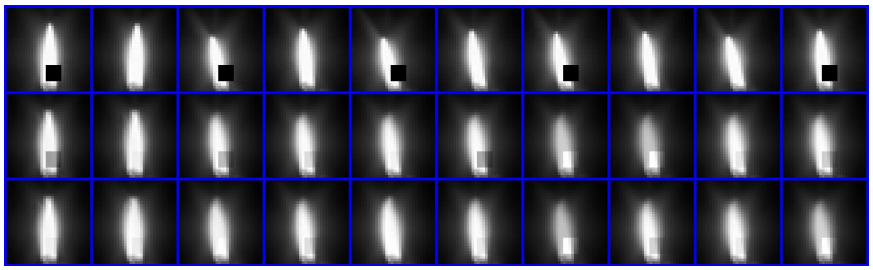


(a) Corrupted original sequence

(b) Reconstructed sequence



(c) Synthesized video using LDS and *SLDS* on DTs with Gaussian noise



(d) Synthesized video using LDS and *SLDS* on DTs with missing data

Figure 4.5: Reconstruction and synthesizing on the candle scene. (a), (b) are $(t = 1, 64, 128, 512, 1024)th$ frame of the corrupted data by Gaussian noisy and the reconstructed data using *SLDS*, respectively. (c) The top row is the synthesized sequence using LDS (128PCs), and the bottom row is the synthesized sequence using *SLDS* ($(t = 2, 1024, 3072, 5120, \ldots, 20480)th$ frame). (d) The top row is the sequence with missing data. The middle row the synthesized sequence using LDS, and the bottom row is the synthesized sequence using *SLDS*.

4.5 Numerical Experiments for Evaluating the *SLDS* Model

In this section, we carry out several experiments on natural image sequences data to demonstrate the practicality of the proposed algorithm. Our test dataset comprises of videos from several benchmark datasets, and data from internet sources (for instance, YouTube).

4.5.1 Datasets

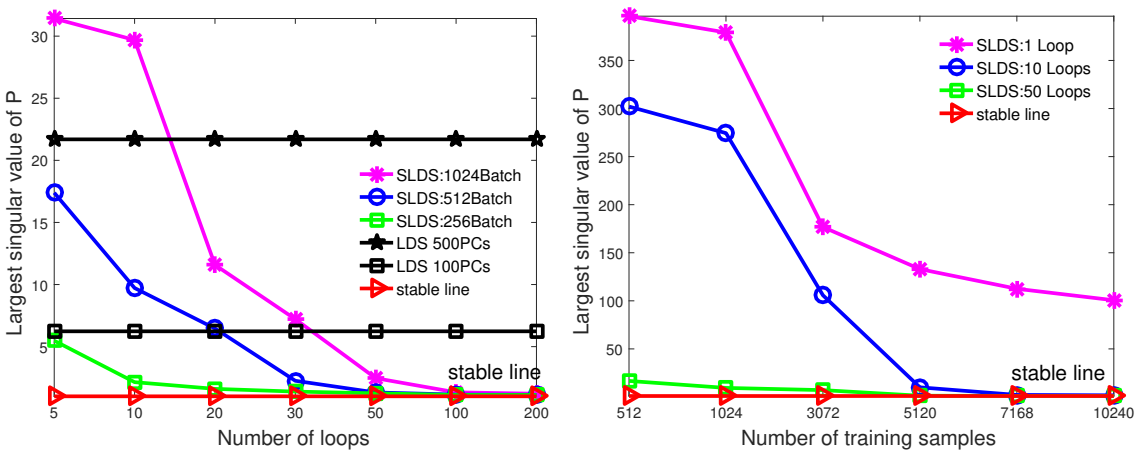
So far, two basic databases have been widely used for DT analysis: the UCLA-DT database [114] and the DynTex database [2].

The UCLA-DT database originally consists of 200 DT sequences with 50 categories, and each category contains 4 video sequences captured from different viewpoints. Its DT sequences have already been pre-processed from their raw form, whereby each sequence is cropped to show its representative dynamics in absence of any static or dynamic background. For each DT sequence, there is only a single DT is present. Each sequence has $T = 75$ frames with $m = 48 \times 48$ pixels.

The DynTex database is a large pool of DT sequences and consists of a total of 656 AVI video sequences with the size of 720×576. It aims to serve as a standard database for dynamic texture research and to accommodate the needs for assessing the different research issues, such as texture synthesis, detection, segmentation and recognition, cf. [2].

DynTex++ [179] is a well-designed dataset from original DynTex database and is often used for evaluating DT classification algorithms. It eliminated sequences that contained more than one DT, contained dynamic background, included panning/zooming, or did not depict much motion. The remaining sequences were then labeled as 36 classes. Each class has 100 subsequences of length 50 frames with 50×50 pixels cropped from the original sequences.

As the color information is not our focus, all images will be normalized to the grayscale between 0 and 1. Throughout all experiments, we consistently set $\lambda_2 = 10^{-5}$ in Eq. (3.16).



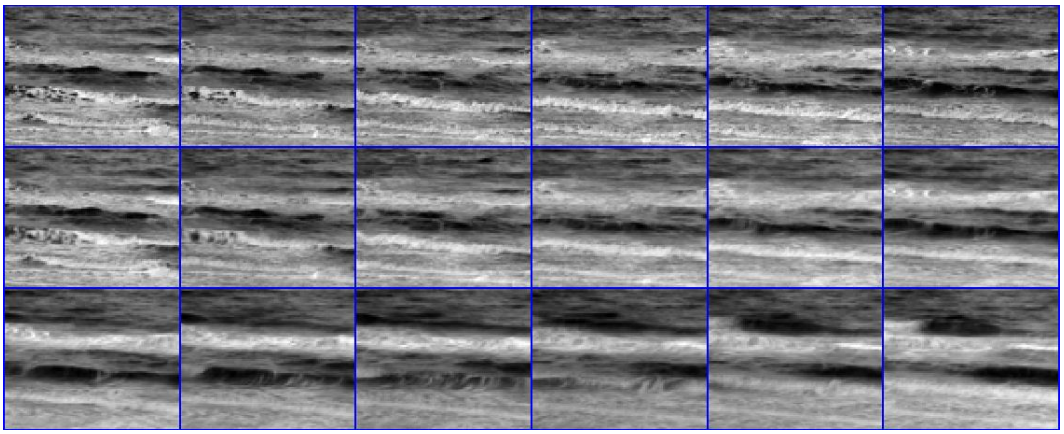
(a) The largest singular value of $\mathbf{P}$ for *SLDS* and LDS with increasing loops.

(b) The largest singular value of $\mathbf{P}$ for *SLDS* with increasing number of training samples

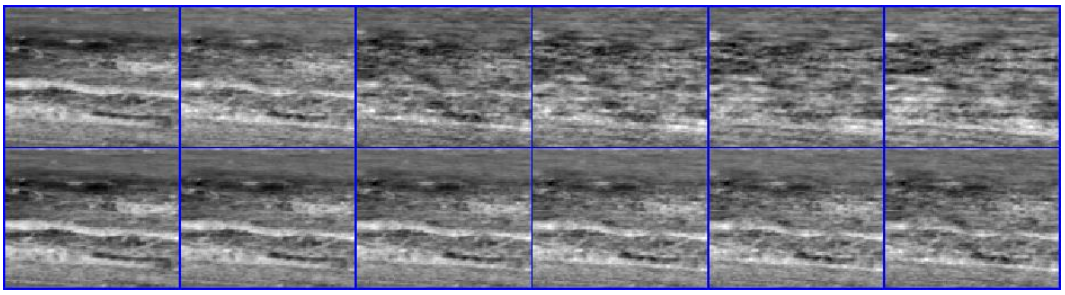
Figure 4.6: The maximum singular value of $\mathbf{P}$ for *SLDS* and LDS. The "stable line" denotes the boundary for stable $\mathbf{P}$, in which the singular value is equal to 1. (a). Comparing the largest singular value of $\mathbf{P}$ with increasing loops, on candle video. (b). Largest singular value of $\mathbf{P}$ with increasing training samples, on candle video, $n = 512, 1024, 3072, 5120, 7168, 10240$. Both select the 1024×512 dictionary.

Table 4.1: Synthesizing results on sequence of burning candle.

Instance	LDS, (PCs)			*SLDS*, (loops)				
	64	128	256	1	50	100	200	400
Compression rate (%)	6.25	12.50	25.00	1.02	3.29	3.41	3.50	3.55
σ	0.9802	0.9833	0.9849	1.78	1.06	0.9992	0.9994	0.9994
$e_{\mathbf{x}}$	135265	135138	135060	1360	60.2	58.8	56.0	71.3
$e_{\boldsymbol{\phi}}$	101.58	135.88	168.95	37500	171.99	75.52	61.96	46.18



(a) Tidewater and synthesized data (bottom two rows)



(b) synthesized data using LDS and *SLDS*

Figure 4.7: Tidewater from DynTex database. (a) (Original) Tidewater sequence ($m = 40 \times 56, T = 3297 - 1$) and reconstructed data via *SLDS* (bottom 2 rows ($t = 1, 21, 41, \ldots, 101$)$st$ frame). (b)The top row is synthesized sequence using LDS (200PCs), and the bottom row is synthesized sequence using *SLDS*, (($t = 4001, 5351, 6401, \ldots, 8551$)$st$ frame).

4.5.2 Dynamic Textures Synthesis

DT synthesis is the process of creating a longer or infinite DT sequence using a video exemplar as input. This can be achieved starting either from a model of a physical phenomenon or from existing video sequences, cf. [180]. This chapter focuses on the image-based methods, i.e., finding a mathematical model of the video, e.g., LDS or *SLDS*, that can explain the dynamic process of generation of a DT sequence. Once this model is at disposal, it can generate longer video sequences, by just producing new video frames using this model. In what follows, we test our *SLDS* on DT synthesis in comparison with classical LDS method.

Firstly, we show the performance on reconstruction and synthesis with a grayscale video of burning candle from YouTube, which is corrupted by Gaussian noise or occlusion. This video has 10240 frames with size of 32×32, as seen in Fig. 4.5(a). But in the first experiment, we select its first 1024 frames as training sequence. The initial dictionary is with the size of 1024×512. We set $\lambda_1 = 0.2$, $\gamma_1 = 0.5$, $\gamma_2 = 0.02$, and $\gamma_3 = 0.0005$. After obtaining $\mathbf{D}$ and $\mathbf{P}$ by minimizing Eq. (4.19), the synthetic data can be generated easily by $\boldsymbol{\phi}_{t+1} = \Gamma_\beta(\mathbf{P}\boldsymbol{\phi}_t)$, where Γ_β is the element-wise hard thresholding operator which keeps the elements whose magnitudes are larger than β while setting the rest zeros. We also use a following convex formulation to estimate $\boldsymbol{\phi}_{t+1}$, i.e.,

$$\min_{\boldsymbol{\phi}_{t+1}} \frac{1}{2}\|\boldsymbol{\phi}_{t+1} - \mathbf{P}\boldsymbol{\phi}_t\|_2^2 + \lambda_1\|\boldsymbol{\phi}_{t+1}\|_1 + \lambda_2\|\boldsymbol{\phi}_{t+1}\|_2.$$

Table 4.1 shows the DT synthesis performance on burning candle with Gaussian noise. The error pairs $(e_{\mathbf{x}}, e_{\boldsymbol{\phi}})$ are defined as $e_{\mathbf{x}} = \sum_t \|\mathbf{x}_t - \mathbf{D}\boldsymbol{\phi}_t\|$, $e_{\boldsymbol{\phi}} = \sum_t \|\boldsymbol{\phi}_{t+1} - \mathbf{P}\boldsymbol{\phi}_t\|$, and the largest eigenvalue of $\mathbf{P}$ is denoted by σ. The compression rate for *SLDS* is the sparsity of $\boldsymbol{\phi}$ to m, and for LDS is the number of PCs to m. Table 4.1 shows that *SLDS* can obtain the stable dynamic matrix $\mathbf{P}$ ($\sigma \leq 1$), smaller compression rate and smaller error $(e_{\mathbf{x}}, e_{\boldsymbol{\phi}})$ of cost function (4.19), by increasing the number of main loops in Algorithm 1. Stability for (4.2) and (4.3) will be achieved while the largest singular value is bounded by $\mathbf{1}$, cf. [169]. The main formulation (4.19) with constraints on $\mathbf{P}$ has enforced stability on $\mathbf{P}$, but doesn't guarantee all the maximum of singular values are less than $\mathbf{1}$. However, this goal can be reached while the training samples are huge or increasing the number of main loops in Algorithm 1, as seen in Fig. 4.6.

Fig. 4.5 ($a \sim c$) is the visual comparison between LDS and *SLDS*. *SLDS* performs well on denoising against corruption by Gaussian noise. In the case of occlusion in Fig. 4.5 (d), random 50 frames of the 1024 burning candle video are corrupted by a (6×7) rectangle. The length of both synthesizing data is 1024, based on the first frame of the burning candle. The experimental results show that 87.01% of the synthesizing data from LDS are corrupted by this rectangle, but only 9.47% are slightly corrupted by this rectangle for *SLDS*. The synthesizing images are shown in the bottom two lines of Fig. 4.5 (d).

Similar to Fig. 4.5, we then perform *SLDS* on another DT sequence, namely Tidewater, from DynTex database. The synthesizing experiments are depicted in Fig. 4.7. Fig 4.7(a) shows that *SLDS* can model and synthesize such DT sequence. For synthesizing a longer videos in Fig 4.7(b), compared with LDS, *SLDS* also performs better.

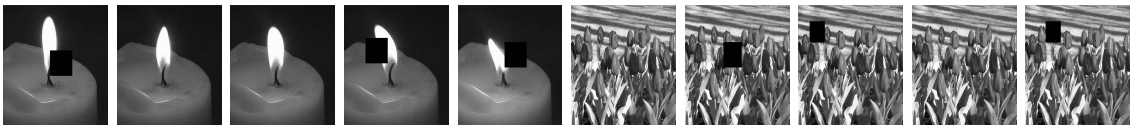


Figure 4.8: Examples of some training samples. The top line images set is from the class of "candles" and the bottom one is from the class of "flowers".

Table 4.2: DT recognition rates on the DynTex++ database with occlusion.

Occlusion rate (%)	0	5	15	30
LDS-NN (20PCs)	69.72	45.00	25.14	14.17
LDS-SRC (20PCs)	73.14	56.66	29.04	15.26
MMDL [179]	63.7	-	-	-
SLDS-NN	70.28	64.72	44.44	22.36
global *SLDS*	88.28	88.08	84.11	71.34
SLDS-SRC	90.64	88.82	83.21	69.10

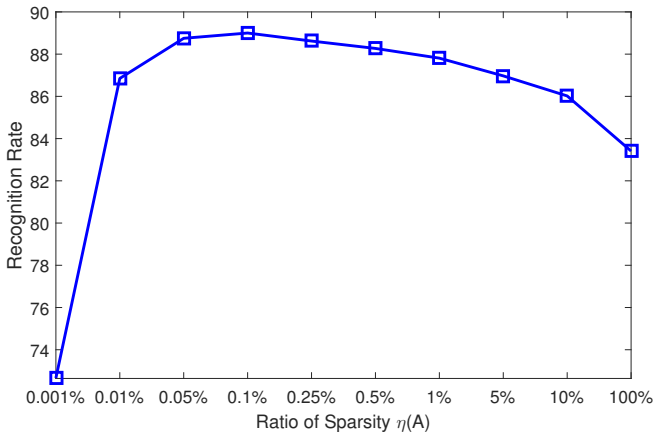


Figure 4.9: Applying *SLDS* classifier (global) on DynTex++ with different choices of $\eta(\mathbf{P})$.

4.5.3 Dynamic Textures Classification

In this section, the performance of the proposed *SLDS* is evaluated for DT classification on DynTex++ and UCLA-DT 50 databases. We address the multi-class classification problem with a one-against-all strategy. All recognition experiments are repeated ten times with different randomly selected training and test subsets, and the average of per-class recognition rates is recorded for each run.

The first classification experiment is applied on DynTex++. We use total 3600 videos with $c = 36$ and randomly choose 50 videos per class for training, and the rest 50 videos for test. For *SLDS-SRC* classifier in Section 4.4.2, we set $\lambda_1 = 0.1$, $\gamma_7 = 0.005$, $\gamma_8 = 0.2$, $p = 10$,

Table 4.3: DT recognition rates on the UCLA-DT 50 database with missing pixels.

The rates of missing pixels	LDS-NN (25PCs)	LDS-SRC (25PCs)	MMDL [179]	*SLDS-NN*	global *SLDS*	*SLDS-SRC*
0%	88.08	94.32	99.00	90.22	96.18	97.23
5%	83.72	85.10	89.40	88.13	96.02	96.84
15%	55.06	64.26	70.64	84.34	94.44	94.68
30%	17.90	26.28	18.18	78.42	91.12	84.48
50%	12.88	20.18	14.42	58.65	83.56	74.00

$\tau = 25$, and $c \times k = 36 \times 20$. For DT sequences from j^{th} class, we train its sub-dictionary $\mathbf{D}_j$ with the size of 100×20. We then combine all the sub-dictionaries as dictionary $\widehat{\mathbf{D}}$ with the size of 100×720. For global *SLDS* classifier in Section 4.4.1, we set $\gamma_4 = 1$, $\gamma_5 = 0.002$, and $\gamma_6 = 0.5$. We choose the dictionary size as $m = 2500, k = 36 \times 20$. Note that, LDS-NN, *SLDS-NN*, LDS-SRC, *SLDS-SRC* are classification methods that employ Nearest Neighbors (NN) classifier or SRC classifier to classify the model parameters $(\mathbf{P}, \mathbf{D})$ learned by LDS or SLDS. In order to evaluate the robustness of *SLDS* to non-Gaussian noise, Table 4.2 depicts the recognition results with increasing occlusion rates for test data. Compared to LDS-NN and LDS-SRC, Table 4.2 shows the proposed global *SLDS* and *SLDS-SRC* classifiers perform better while the test videos are corrupted by increasing occlusion. Some DT examples corrupted by occlusion are shown in Fig. 4.8. In addition, compared with *SLDS-SRC* classifier, the global *SLDS* classifier achieves a higher recognition rate while having a heavy occlusion (e.g., 30% occlusion).

Let us denote by $\eta(\mathbf{P}) = s/(k^2)$ the sparsity ratio of the transition matrix $\mathbf{P}^{k \times k}$ with s denoting the sparsity of $\mathbf{P}$. Fig. 4.9 depicts the recognition rate against $\eta(\mathbf{P})$ using global *SLDS* classifier. It is easy to see that the setting of a suitable sparsity of $\mathbf{P}$ can improve the recognition rate.

The second classification experiment is performed on UCLA-DT 50 database. Since these 50 classes contain the same DTs at different viewpoints, they can be grouped together to form 9 classes, as in [181]. For *SLDS-SRC* classifier, we set $\lambda_1 = 0.1$, $\gamma_7 = 0.005$, $\gamma_8 = 0.25$, $p = 10$, $\tau = 25$, and $c \times k = 50 \times 20$. The size of each sub-dictionary is set with 100×20. We also combine all the sub-dictionaries as dictionary $\widehat{\mathbf{D}}$ with the size of 100×1000. For global *SLDS* classifier, we set $\gamma_4 = 1$, $\gamma_5 = 0.001$, and $\gamma_6 = 0.2$. We choose the dictionary size as $m = 2304, k = 50 \times 20$. The missing pixels for an image is another kind of non-Gaussian noise. Note that, the image with a% missing pixels means that we set the values of random a% pixels in such an image to zero. Table 4.3 shows the recognition results for DTs corrupted by missing pixels, in comparison of classical methods, i.e., LDS-NN and LDS-SRC. For LDS-SRC, we choose 25PCs for "states" which achieved the best performance recorded in [178]. As shown in Table 4.3, for the DT classification without missing pixels, *SLDS* classifiers perform better than classical LDS-NN and LDS-SRC, and behind the current record achieved by MMDL in [179]. But for classification with increasing

missing pixels, *SLDS* based methods (i.e., *SLDS-NN*, global *SLDS* and *SLDS-SRC*) decease slowly, compared with the dramatically decreasing performance of LDS-NN, LDS-SRC and MMDL.

Overall, the results in this subsection suggest that the proposed *SLDS* classifiers can achieve a good performance on DTs classification, especially when the DTs are corrupted by heavy non-Gaussian noise.

4.6 Summary

This chapter has presented an alternative method, called *SLDS*, to model the dynamic process of DTs. In *SLDS*, the sparse events over a dictionary are imposed as transition "states". A constrained transition matrix is learned to represent each DT sequence. It has been demonstrated that the proposed method is much more robust in synthesizing and reconstruction on DTs corrupted by Gaussian noise. To enable the *SLDS* in DT's classification, we proposed two discriminative *SLDS* algorithms associated with a sparse transition matrix. Our experiments have shown that an appropriately sparse transition matrix could well capture the discrimination of DT sequences. Especially, *SLDS* and *SLDS* classifiers become more powerful in the case of test data corrupted by non-Gaussian noise, such as occlusion or missing pixels. For instance, in one test case, the recognition rate of *SLDS-SRC* decreased from 97.23% to 84.48% while conventional LDS-SRC approach decreased from 94.32% to 26.28% when 30% missing pixels occur.

Chapter 5

Sparse Low Dimensional Representation Learning

By adopting the representation learning framework presented in Chapter 3, this chapter focuses on the problem of finding appropriate low dimensional image representations to facilitate the specific problem of learning. The core concept of our development is to disentangle sparse representations of images by employing the trace quotient criterion. As already discussed in previous chapters, sparse representation is a convenient, powerful tool to identify underlying self-explanatory factors of data, and the trace quotient criterion is known for disentangling underlying discriminative factors in data. We construct a unified cost function for jointly learning both a sparsifying dictionary and a dimensionality reduction transformation. The cost function is widely applicable to various algorithms in three classic machine learning scenarios, namely, unsupervised, supervised, and semi-supervised learning. Our proposed optimization algorithm leverages the efficiency of geometric optimization on Riemannian manifolds and the differentiability of sparse solutions with respect to dictionary. Performance of our proposed framework is investigated on several machine learning tasks, such as 3D data visualization, face/digit/cartoon recognition and object/scene categorization.

5.1 Introduction

In Chapter 1, we have discussed that finding appropriate low dimensional representation of data, i.e., *low dimensional representation learning* could facilitate the learning task of interest. Specifically, for image processing, appropriate low dimensional representation of images has demonstrated its prominent capability and convenience in various applications, such as images visualization [116, 182, 13], segmentation [117, 119], clustering [116, 18, 183] and classification [184, 118, 185, 186, 187]. This Chapter focuses on the problem of constructing effective low dimensional representation algorithms to disentangle underlying explanatory information in the data for solving various computer vision problems. One key difficulty in learning low dimensional image representations lies in the observation that different representations of image can disentangle different explanatory information or factors, which are supposed to promote the specific machine learning task [5, 188]. Disentangling appropriate explanatory information that can explain internal intricate structure in high dimensional image sets become a challenging problem in image processing [116, 184, 14]

To address such a challenge, in Chapter 3, a two-layer representation learning framework was proposed to disentangle useful information hidden in sparse representations. Since sparse representations of data have been observed to contain rich explanatory information of the data with respect to certain learning tasks. In this chapter, we follow the framework depicted in Fig. 3.1. By employing an effective disentangling instrument, we propose to jointly disentangle a sparsifying dictionary and underlying factors hidden in image sparse representations. Among various low dimensional representation learning instruments, the trace quotient criterion is a simple but powerful framework for disentangling underlying discriminative factors in data. This generic criterion is shared by various classic dimensionality reduction (DR) methods, which include PCA, Linear Discriminant Analysis (LDA) [186], Linear Local Embedding (LLE) [13], Marginal Fisher Analysis (MFA) [187], Orthogonal Neighbourhood Preserving Projection (ONPP) [15], Locality Preserving Projections (LPP) [183], Orthogonal LPP (OLPP) [189], Spectral Clustering (SC) [18], semi-supervised LDA (SDA) [190], etc.

Our main construction in this chapter is to apply the trace quotient criterion to further disentangle sparse representations for triple supervised, unsupervised and semi-supervised learning tasks. In the rest of the chapter, we refer to such a model as SPARse LOW dimensional representation learning (*SparLow*). Then, we construct a differentiable cost function to jointly learn a sparsifying dictionary and a DR transformation, which are defined *on the product manifold of the product of unit spheres and the Grassmann manifold*. Finally, we develop a conjugate gradient (CG) algorithm for minimizing the cost function.

The chapter is organized as follows. Section 5.2 provides a brief review on low dimensional representations based on the trace quotient criterion. In Section 5.3, we first construct a generic cost function for learning both the sparsifying dictionary and the orthogonal DR transformation, and then develop a geometric conjugate gradient algorithm on the underlying smooth manifold. Several applications of the proposed generic model are discussed in Section 5.4, together with their experimental evaluations presented in Section 5.5. Finally, conclusions and outlooks are given in Section 5.6.

5.2 Optimization of Trace Quotient Criterion

In this section, we briefly review some state of the art results in trace quotient optimization based dimensionality reduction.

Classic dimensionality reduction methods aim to find a lower-dimensional representations $\mathbf{y}_i \in \mathbb{R}^l$ of given data samples $\mathbf{x}_i \in \mathbb{R}^m$ with $l < m$, via a mapping $\mu\colon \mathbb{R}^m \to \mathbb{R}^l$, which captures certain desired properties of the data to facilitate the specific applications. Many classic DR methods restrict the mapping μ to an orthonormal transformation. Let us define the set of $m \times l$ matrices, consisting of l orthonormal columns in $\mathbb{R}^m$, by $St(l, m)$ as Eq. (3.3). In this work, we confine ourselves to the form of orthonormal linear mapping as $\mu\colon \mathbb{R}^m \to \mathbb{R}^l,\ \mu(\mathbf{x}) := \mathbf{V}^\top \mathbf{x}$. This model covers a wide range of classic supervised and unsupervised learning methods, such as LDA, MFA, PCA, OLPP, and ONPP. Further details are given in section 4.3.

One generic algorithmic framework to find the orthogonal transformation $\mathbf{V} \in St(l,m)$ is formulated as a maximization problem of the so-called trace quotient or trace ratio, i.e.,

$$\underset{\mathbf{V}\in St(l,m)}{\operatorname{argmax}} \frac{\operatorname{tr}(\mathbf{V}^\top \mathbf{A}\mathbf{V})}{\operatorname{tr}(\mathbf{V}^\top \mathbf{B}\mathbf{V}) + \sigma}, \tag{5.1}$$

where matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times m}$ are often symmetric positive semidefinite, and constant $\sigma > 0$ is chosen to prevent the denominator from being zero. Both matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times m}$ are constructed according to the specific problems [14, 15], and examples will be given and discussed in Section 4.3.

It is obvious that solutions of the problem in (5.1) are rotation invariant, i.e., let $\mathbf{V}^* \in St(l,k)$ be a solution of the problem (5.1), then so is $\mathbf{V}^*\boldsymbol{\Theta} \in St(l,k)$ for any $\boldsymbol{\Theta} \in \mathbb{R}^{l\times l}$ being orthogonal. Namely, the solution set of the problem in (5.1) is the set of all l-dimensional linear subspace in $\mathbb{R}^m$. In order to cope with this structure, we employ the Grassmann manifold, which can be alternatively identified as the set of all m-dimensional rank-l orthogonal projectors, as defined in Eq. (3.30). Thus, the trace quotient maximization problem can be reformed as

$$\underset{\mathbf{P}\in Gr(l,m)}{\operatorname{argmax}} \frac{\operatorname{tr}(\mathbf{A}\mathbf{P})}{\operatorname{tr}(\mathbf{B}\mathbf{P}) + \sigma}. \tag{5.2}$$

Although various efficient optimization algorithms over Riemannian manifolds have been developed to solve the trace quotient maximization problem [14, 191, 155, 192, 193], the construction described in the next section requires further constructive development.

5.3 The Proposed Joint Learning Framework

In this section, we firstly present a generic cost function, which adopts the sparsifying dictionary learning in the framework of trace quotient maximization in Section 5.2. Then a geometric conjugate gradient algorithm is presented in Section 5.3.2.

5.3.1 A Generic Cost Function

As suggested by the work of [130, 26, 125, 119], further processing on the sparse representation is capable of unveiling task-related underlying factors, potentially for both supervised and unsupervised learning tasks. In what follows, we construct a cost function, which allows to jointly learn both the sparsifying dictionary and the orthogonal transformation in the framework of trace quotient maximization.

Let us denote by $\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) := [\boldsymbol{\phi}_{\mathbf{D}}(\mathbf{x}_1), \ldots, \boldsymbol{\phi}_{\mathbf{D}}(\mathbf{x}_n)] \in \mathbb{R}^{k\times n}$ the sparse representation of the data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ for a given dictionary $\mathbf{D}$. The sparse representations are confined to the solutions of the sparse regression problem as in Eq. (3.1). Let $\mathcal{A}\colon \mathbb{R}^{k\times n} \to \mathbb{R}^{k\times k}$ and $\mathcal{B}\colon \mathbb{R}^{k\times n} \to \mathbb{R}^{k\times k}$ be two smooth functions that serve as generating functions for the matrices $\mathbf{A}$ and $\mathbf{B}$ in the trace quotient in Eq. (5.2). Constructions of the mappings $\mathcal{A}$ and $\mathcal{B}$ are exemplified in Section 5.4. Then we define a generic trace quotient function in sparse

representations as

$$\begin{aligned} f\colon \mathcal{S}(m,k)\times Gr(l,k) &\to \mathbb{R}, \\ f(\mathbf{D},\mathbf{P}) &:= \frac{\operatorname{tr}\left(\mathcal{A}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))\mathbf{P}\right)}{\operatorname{tr}\left(\mathcal{B}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))\mathbf{P}\right)+\sigma}. \end{aligned} \tag{5.3}$$

When the structure function $\mathcal{A}$ and $\mathcal{B}$ are smooth in its parameter, it is direct to conclude that the function f is locally differentiable on the product manifold $\mathcal{S}(m,k)\times Gr(l,k)$.

In order to prevent solution dictionaries from being highly coherent, which is critical for guaranteeing the local smoothness of the sparse solutions, we employ a log-barrier function on the scalar product of all dictionary columns to control the mutual coherence of the learned dictionary $\mathbf{D}$ [144], i.e., for dictionary $\mathbf{D}=[\mathbf{d}_1,\ldots,\mathbf{d}_k]\in\mathbb{R}^{m\times k}$,

$$h_c(\mathbf{D}) := -\sum_{1\le i<j\le k} \tfrac{1}{2}\log\left(1-(\mathbf{d}_i^\top\mathbf{d}_j)^2\right). \tag{5.4}$$

Furthermore, the authors in [121] argues that an appropriate dictionary of choice in sparse representation can reveal the semantics of the data. We propose to the following regularizer on the dictionary to be learned

$$h_d(\mathbf{D}) := \tfrac{1}{2}\|\mathbf{D}-\mathbf{D}^*\|_F^2, \tag{5.5}$$

where $\mathbf{D}^*$ is the optimal data-driven dictionary learned from the data $\mathbf{X}$, i.e., learning dictionary adapted to data reconstruction. It measures the distance between an estimated dictionary $\mathbf{D}$ and the optimal dictionary $\mathbf{D}^*$ in terms of Frobenius norm. Practically, we use a dictionary $\widehat{\mathbf{D}}$ produced by state of the art methods, such as K-SVD, to replace $\mathbf{D}^*$. Our experiments have verified that the heuristic regularizer h_d ensures solutions of the generic cost function J defined in Eq. (5.6) to be self explanatory to the data, and guarantees stable performance towards the task of learning.

To summarize, we construct the following cost function to jointly learn both the sparsifying dictionary and the orthogonal transformation, i.e.,

$$\begin{aligned} J\colon \mathcal{S}(m,k)\times Gr(l,k) &\to \mathbb{R}, \\ J(\mathbf{D},\mathbf{P}) &:= -f(\mathbf{D},\mathbf{P})+\mu_1 h_c(\mathbf{D})+\mu_2 h_d(\mathbf{D}), \end{aligned} \tag{5.6}$$

where the two weighting factors $\mu_1>0$ and $\mu_2>0$ control the influence of the two regularizers on the final solution.

5.3.2 A Geometric Conjugate Gradient Algorithm

In Chapter 3, we have investigated the (local) differentiability of the sparse representation in the dictionary from the perspective of global analysis. By leveraging such a benefit, in this subsection, we briefly present a geometric CG algorithm on the product manifold $\mathcal{M}:=\mathcal{S}(m,k)\times Gr(l,k)$ to maximize the generic cost function J, defined in Eq. (5.6). As

Algorithmus 2 : A *CG-SparLow* Algorithm.

Input : $\mathbf{X} \in \mathbb{R}^{m\times n}$ and functions $\mathcal{A}$: $\mathbb{R}^{k\times n} \to \mathbb{R}^{k\times k}$ and $\mathcal{B}$: $\mathbb{R}^{k\times n} \to \mathbb{R}^{k\times k}$ as specified in Section 5.4 ;

Output: $\mathbf{D}^* \in \mathcal{S}(m,k)$ and $\mathbf{P}^* \in Gr(l,k)$;

S1: Initialize $\mathbf{D}^{(0)} \in \mathcal{S}(m,k)$, $\mathbf{V}^{(0)} \in St(l,k)$, $\mathbf{P}^{(0)} = \mathbf{V}^{(0)}\mathbf{V}^{(0)\top}$, $\boldsymbol{\Omega}^{(0)} = [\mathbf{P}^{(0)}, \nabla J(\mathbf{P}^{(0)})]$, and set $i = -1$;

S2: Compute $\mathbf{G}^{(0)} = (\mathbf{G}_{\mathbf{D}}^{(0)}, \mathbf{G}_{\mathbf{P}}^{(0)})$, $\mathbf{H}^{(0)} = -\mathbf{G}^{(0)}$;

S3: Set $i = i + 1$;

S4: Update sparse representation matrix $\boldsymbol{\Phi}^{(i)}$ via Eq. (3.1);

S5: Find step size $t^{(i)}$ via a backtracking line search along geodesics, cf. Algorithm 3 and [144] ;

S6: Update $\mathbf{D}^{(i+1)} \leftarrow \Gamma_{\mathcal{S}}(\mathbf{D}^{(i)}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i)}, t^{(i)})$, cf. Eq. (3.33); Let $\mathbf{Q} := (\mathbf{I} + t^{(i)}\boldsymbol{\Omega}^{(i)})_Q$ via QR decomposition, then, update $\mathbf{V}^{(i+1)} = \mathbf{Q}\mathbf{V}^{(i)}$, $\mathbf{P}^{(i+1)} = \mathbf{V}^{(i+1)}\mathbf{V}^{(i+1)\top}$;

S7: Update $\mathbf{G}^{(i+1)} = (\mathbf{G}_{\mathbf{D}}^{(i+1)}, \mathbf{G}_{\mathbf{P}}^{(i+1)})$, where $\mathbf{G}_{\mathbf{D}}^{(i+1)} = \mathit{\Pi}_{\mathbf{D}}(\nabla_J(\mathbf{D}^{(i+1)})), \mathbf{G}_{\mathbf{P}}^{(i+1)} = \boldsymbol{\pi}_{\mathbf{P}}(\nabla_J(\mathbf{P}^{(i+1)}))$;

S8: Compute $\beta^{(i)}$ according to Eq. (3.29);

Update $\boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i+1)} = -\mathbf{G}_{\mathbf{D}}^{(i+1)} + \beta^{(i)}\mathcal{T}_{\mathcal{S},\boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i)}}$;

Let $\mathcal{T}_{Gr,\boldsymbol{\Omega}^{(i)}} := \mathbf{Q}[\boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i)}, \mathbf{P}^{(i)}]\mathbf{Q}^\top$,

$\boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i+1)} = -\mathbf{G}_{\mathbf{P}}^{(i+1)} + \beta^{(i)}[\mathcal{T}_{Gr,\boldsymbol{\Omega}^{(i)}}, \mathbf{P}^{(i+1)}]$,

$\boldsymbol{\Omega}^{(i+1)} = [\boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i+1)}, \mathbf{P}^{(i+1)}]$;

Update $\boldsymbol{\mathcal{H}}^{(i+1)} = (\boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i+1)}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i+1)})$;

S10: If $\left\|\mathbf{G}^{(i+1)}\right\|$ is small enough, stop. Otherwise, go to Step 3 (**S3**);

introduced in Section 3.4, it is known that CG algorithms offer prominent properties, such as a superlinear rate of convergence and the applicability to large scale optimization problems with low computational complexity, e.g., in sparse recovery [144, 25].

Although the technique of geometric optimization is nowadays popularly available, development of such an algorithm on a product manifold is not necessarily trivial. Note that, the required concepts of Geometry of Grassmann manifold and product of r unit spheres have been introduced in Chapter 3. Since the dimensions m, k and l are fixed throughout the rest of the paper, the product of k unit spheres is further on denoted by $\mathcal{S}$, and Grassmann manifold is denoted by Gr in some place.

In the following, we use the geometric CG algorithm to resolve the optimization problem (5.6), the solution of which is restricted to a product of Oblique manifold and Grassmann manifold. Let $\mathcal{M} := \mathcal{S}(m,k) \times Gr(l,k)$ be a product manifold of a Riemannian submanifold of $\mathbb{R}^{m\times k} \times \mathbb{R}^{k\times k}$, and let J: $\mathcal{M} \to \mathbb{R}$ be the differentiable cost function of Eq.(5.6). The

Algorithmus 3 : Backtracking Line Search on $\mathcal{M}$ in the i^{th} iteration

1: **Input:** $t_0^{(i)} > 0, 0 < c_1 < 1, 0 < c_2 < 0.5, \mu > 0, \boldsymbol{\Theta}^{(i)}, \mathbf{G}^{(i)}, \boldsymbol{\mathcal{H}}^{(i)}$
2: **Set:** $t \leftarrow t_0^{(i)}$
3: **while** $J(\Gamma_{\boldsymbol{\mathcal{M}}}(\boldsymbol{\Theta}^{(i)}, \boldsymbol{\mathcal{H}}^{(i)}, t)) > J(\boldsymbol{\Theta}^{(i)}) + c_2 t \left\langle \mathbf{G}^{(i)}, \boldsymbol{\mathcal{H}}^{(i)} \right\rangle$ **do**
4: $\quad t \leftarrow c_1 t$
5: **end while**
6: Output: $t^{(i)} \leftarrow t$

general solution to optimization problem (5.6) on matrix manifold, is an element of $\mathcal{M}$, denoted by $\boldsymbol{\Theta} \in \mathcal{M}, \boldsymbol{\Theta} = (\mathbf{D}, \mathbf{P})$.

By the product structure of $\mathcal{S}(m,k) \times Gr(l,k)$, the tangent space of $\mathcal{M}$ at a point $\boldsymbol{\Theta} \in \mathcal{M}$ is simply the product of all individual tangent spaces, i.e.,

$$T_{\boldsymbol{\Theta}}\mathcal{M} := T_{\mathbf{D}}\mathcal{S}(m,k) \times T_{\mathbf{P}}Gr(l,k). \tag{5.7}$$

Then we denote the Riemannian gradient of J at $(\mathbf{D}, \mathbf{P})$ by $\mathbf{G} := (\mathbf{G_D}, \mathbf{G_P})$, and the CG direction by $\boldsymbol{\mathcal{H}} \in T_{(\mathbf{D},\mathbf{P})}\mathcal{M}$. Therein,

$$\mathbf{G_D} := \mathit{\Pi}_{\mathbf{D}}\big(\nabla_J(\mathbf{D})\big)$$

and

$$\mathbf{G_P} := \mathit{\Pi}_{\mathbf{P}}\big(\nabla_J(\mathbf{P})\big),$$

where $\nabla_J(\mathbf{D})$ and $\nabla_J(\mathbf{P})$ are the Euclidean gradient of J with respect to $\mathbf{D}$ and $\mathbf{P}$, respectively. Therein, the maps, $\mathit{\Pi}_{\mathbf{D}} \colon \mathbb{R}^{m\times k} \to T_{\mathbf{D}}\mathcal{S}(m,k)$ and $\mathit{\Pi}_{\mathbf{P}} \colon \mathbb{R}^{k\times k} \to T_{\mathbf{P}}Gr(l,k)$, are defined in Eq. (3.32) and in Eq. (3.37), respectively.

The Euclidean gradient of J with respect to $\mathbf{P}$ is computed by

$$\nabla J(\mathbf{P}) = -\frac{(\mathrm{tr}(\boldsymbol{\mathcal{B}}\mathbf{P}) + \sigma)\boldsymbol{\mathcal{A}}^\top - \mathrm{tr}(\boldsymbol{\mathcal{A}}\mathbf{P})\boldsymbol{\mathcal{B}}^\top}{(\mathrm{tr}(\boldsymbol{\mathcal{B}}\mathbf{P}) + \sigma)^2}, \tag{5.8}$$

where $\boldsymbol{\mathcal{B}} := \mathcal{B}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))$, and $\boldsymbol{\mathcal{A}} := \mathcal{A}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))$.

Let us denote by $\mathbf{U} := [\mathbf{u}_1, \ldots, \mathbf{u}_n] = \nabla\varsigma(\boldsymbol{\Phi})$ with $\varsigma(\boldsymbol{\Phi}) := \boldsymbol{\mathcal{A}}\mathbf{P}$ and $\mathbf{Q} := [\mathbf{q}_1, \ldots, \mathbf{q}_n] = \nabla\varrho(\boldsymbol{\Phi})$ with $\varrho(\boldsymbol{\Phi}) := \boldsymbol{\mathcal{B}}\mathbf{P}$. By using the chain rule, the directional derivative of f in Eq. (4.25) with respect to $\mathbf{D}$ in direction $\boldsymbol{\mathcal{H}}$ can be given by

$$\begin{aligned} \mathrm{D}\, f(\mathbf{D})\boldsymbol{\mathcal{H}} = \sum_{i=1}^{n} \mathcal{V}\{ & \frac{(\mathrm{tr}(\boldsymbol{\mathcal{B}}\mathbf{P}) + \sigma)\mathbf{u}_{i,\Lambda_i}\, \mathrm{D}\, \boldsymbol{\phi_i}(\mathbf{D}_{\Lambda_i})\boldsymbol{\mathcal{H}}}{(\mathrm{tr}(\boldsymbol{\mathcal{B}}\mathbf{P}) + \sigma)^2} \\ & - \frac{\mathrm{tr}(\boldsymbol{\mathcal{A}}\mathbf{P})\mathbf{q}_{i,\Lambda_i}\, \mathrm{D}\, \boldsymbol{\phi_i}(\mathbf{D}_{\Lambda_i})\boldsymbol{\mathcal{H}}}{(\mathrm{tr}(\boldsymbol{\mathcal{B}}\mathbf{P}) + \sigma)^2} \} \end{aligned} \tag{5.9}$$

with $\mathrm{D}\, \boldsymbol{\phi_i}(\mathbf{D}_\Lambda)\boldsymbol{\mathcal{H}}$ being defined in Eq. (3.6). Therein, $\mathcal{V}\{\mathbf{z}\}$ denotes the full length vector

of sparse coefficients $\mathbf{z}$. By extracting $\boldsymbol{\mathcal{H}}$, the corresponding Euclidean gradient of f with respect to $\mathbf{D}$, i.e. $\nabla f(\mathbf{D})$, is easily derived from Eq. (5.9).

Taking an example for $\mathcal{A}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) = \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{A}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top$, and $\mathcal{B}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) = \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{B}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top$, where $\mathbf{A}$ and $\mathbf{B}$ are introduced in Section 5.2 and Section 5.4. Based on the solution in problem (3.16) and its directional derivative of Eq.(3.18), the gradient of f in (5.6) with respect to $\mathbf{D}$ can be given by

$$\begin{aligned}\nabla f(\mathbf{D}) =& 2\sum_{i=1}^{n}\mathcal{V}\{\frac{\mathbf{x}_i\mathbf{u}_{i,\Lambda_i}^\top\mathbf{K}_i^{-1} - \mathbf{D}_{\Lambda_i}\mathbf{K}_i^{-1}\widehat{\boldsymbol{\rho}}_i\mathbf{K}_i^{-1}}{\operatorname{tr}(\boldsymbol{\Phi}\mathbf{B}\boldsymbol{\Phi}^\top\mathbf{P})+\sigma} \\ &- \frac{\left(\mathbf{x}_i\mathbf{v}_{i,\Lambda_i}^\top\mathbf{K}_i^{-1} - \mathbf{D}_{\Lambda_i}\mathbf{K}_i^{-1}\widehat{\omega}_i\mathbf{K}_i^{-1}\right)\operatorname{tr}(\boldsymbol{\Phi}^\top\mathbf{U})}{(\operatorname{tr}(\boldsymbol{\Phi}\mathbf{B}\boldsymbol{\Phi}^\top\mathbf{P})+\sigma)^2}\}\end{aligned} \tag{5.10}$$

with

$$\begin{aligned}\boldsymbol{\rho}_i &:= (\mathbf{D}_{\Lambda_i}^\top\mathbf{x}_i - \lambda_1\mathbf{s}_{\Lambda_i})\mathbf{u}_{i,\Lambda_i}^\top,\ \widehat{\boldsymbol{\rho}}_i := \boldsymbol{\rho}_i + \boldsymbol{\rho}_i^\top, \\ \boldsymbol{\omega}_i &:= (\mathbf{D}_{\boldsymbol{\omega}_i}^\top\mathbf{x}_i - \lambda_1\mathbf{s}_{\Lambda_i})\mathbf{v}_{i,\Lambda_i}^\top,\ \widehat{\boldsymbol{\omega}_i} := \boldsymbol{\omega}_i + \boldsymbol{\omega}_i^\top.\end{aligned} \tag{5.11}$$

Finally, the Euclidean gradient $\nabla J(\mathbf{D},\mathbf{P})$ of J with respect to $\mathbf{D}$ is

$$\nabla J(\mathbf{D}) = -\nabla f(\mathbf{D}) + \mu_1\nabla h_c(\mathbf{D}) + 2\mu_2(\mathbf{D}-\mathbf{D}^*) \tag{5.12}$$

where $\nabla h_c(\mathbf{D})$ is the gradient of the logarithmic barrier function $h_c(\mathbf{D})$, defined in Eq. (4.21). Then the Riemannian gradient of J with respect to the first argument $\mathbf{D}$ and the second argument $\mathbf{P}$ are given by

$$\mathbf{G}_\mathbf{D}(\boldsymbol{\Theta}) = \nabla J(\mathbf{D}) - \mathbf{D}\operatorname{diag}(\mathbf{D}^\top\nabla J(\mathbf{D})), \tag{5.13}$$

and

$$\mathbf{G}_\mathbf{P}(\boldsymbol{\Theta}) = [\mathbf{P},[\mathbf{P},\nabla J(\mathbf{P})]], \tag{5.14}$$

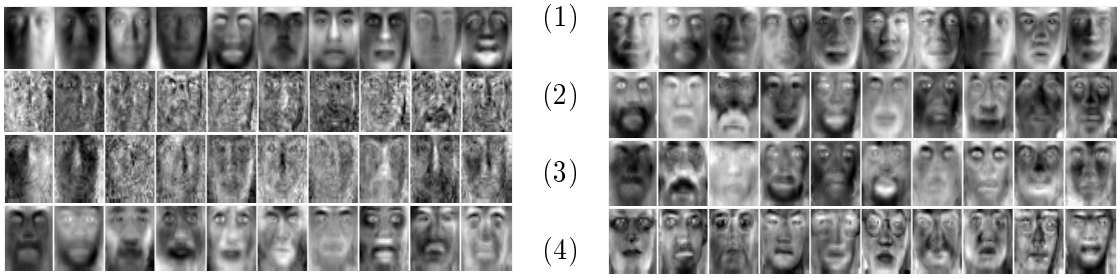
respectively.

By assembling the Riemannian gradients, geodesics and parallel transports on the underlying manifolds, a conjugate gradient (CG) algorithm on $\mathcal{S}(m,k)\times Gr(l,k)$ is straightforward. Given $\dim\mathcal{S}(m,k) = k(m-1)$ and $\dim Gr(l,k) = l(k-l)$, we summarize a CG algorithm for maximizing the function J as defined in Eq. (5.6), cf. Algorithm 2 and Algorithm 3.

Algorithm 2 is based on iterating the following line search scheme, see Fig. 3.5. Given an initial point $\boldsymbol{\Theta}^{(i)} \in \mathcal{M}$, a CG search direction $\boldsymbol{\mathcal{H}}^{(i)} := (\boldsymbol{\mathcal{H}}_\mathbf{D}^{(i)}, \boldsymbol{\mathcal{H}}_\mathbf{P}^{(i)}) \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$, and the step size $t^{(i)} \in \mathbb{R}$, the new data point is updated by

$$\boldsymbol{\Theta}^{(i+1)} = \mathcal{R}_{\boldsymbol{\Theta}^{(i)}}(t^{(i)}\boldsymbol{\mathcal{H}}^{(i)}) = \left(\Gamma_\mathcal{S}(\mathbf{D}^{(i)},\boldsymbol{\mathcal{H}}_\mathbf{D}^{(i)},t^{(i)}), \Gamma_{Gr}(\mathbf{P}^{(i)},\boldsymbol{\mathcal{H}}_\mathbf{P}^{(i)},t^{(i)})\right) \tag{5.15}$$

with $\Gamma_\mathcal{S}(\mathbf{D}^{(i)},\boldsymbol{\mathcal{H}}_\mathbf{D}^{(i)},t^{(i)})$ and $\Gamma_{Gr}(\mathbf{P}^{(i)},\boldsymbol{\mathcal{H}}_\mathbf{P}^{(i)},t^{(i)})$ being defined in Eq. (3.33) and Eq. (3.44), respectively. In the subsequent iterations, the CG direction $\boldsymbol{\mathcal{H}}^{(i+1)} := (\boldsymbol{\mathcal{H}}_\mathbf{D}^{(i+1)}, \boldsymbol{\mathcal{H}}_\mathbf{P}^{(i+1)})$ is a



(a) Features extracted from original data

(b) Features extracted from sparse representations

Figure 5.1: Visualization of facial features. The presented features are generated via Eq. (5.36). From top to bottom: (1) PCA eigenfaces; (2) Laplacianfaces; (3) LLEfaces; (4) Fisherfaces. It needs to draw clear expression, such as smile and pose.

linear combination of the Riemannian gradient $\operatorname{grad} J(\boldsymbol{\Theta}^{(i+1)})$, abbreviated as $\mathbf{G}^{(i+1)}$, and the previous search direction $\boldsymbol{\mathcal{H}}^{(i)} := (\boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i)}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i)})$. Since addition of vectors from different tangent spaces is not defined, we need to transport $\boldsymbol{\mathcal{H}}^{(i)}$ from $T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$ to $T_{\boldsymbol{\Theta}^{(i+1)}}\mathcal{M}$. This is done by *vector transport*

$$\mathcal{T}_{(\boldsymbol{\Theta}^{(i)}, t^{(i)}\boldsymbol{\mathcal{H}}^{(i)})}(\boldsymbol{\mathcal{H}}^{(i)}) :\ = \left(\mathcal{T}_{\mathcal{S}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i)}}, \mathcal{T}_{Gr, \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i)}}\right).$$

Therein, let $\mathcal{T}_{\mathcal{S}, \boldsymbol{\Xi}_{\mathbf{D}}} := \mathcal{T}_{\mathcal{S}}(\boldsymbol{\Xi}_{\mathbf{D}}, \mathbf{D}^{(i)}, \boldsymbol{\mathcal{H}}_{\mathbf{D}}^{(i)}, t^{(i)})$ and $\mathcal{T}_{Gr, \boldsymbol{\Xi}_{\mathbf{P}}} := \mathcal{T}_{Gr}(\boldsymbol{\Xi}_{\mathbf{P}}, \mathbf{P}^{(i)}, \boldsymbol{\mathcal{H}}_{\mathbf{P}}^{(i)}, t^{(i)})$ for any $\boldsymbol{\Xi} \in T_{\boldsymbol{\Theta}^{(i)}}\mathcal{M}$, cf. Eq. (3.34) and Eq. (3.45). Then, the new CG search direction is computed by Eq. (3.28) and Eq. (3.29) in Chapter 3.

5.4 Applications of the *SparLow* Model

In the previous section, we propose a generic regularized cost function, and develop a geometric CG algorithm to maximize the generic cost function J. In what follows, we present counterparts of several classic unsupervised, supervised and semi-supervised learning methods, namely, PCA, LLE, MFA, LDA, Semi-LDA, and more. Note that, the proposed *SparLow* is flexible and it is not limited to such methods depicted in the following. Experimental evaluations are conducted in Section 5.5, to illustrate the performance of our proposed framework, in comparison to several direct competitors.

5.4.1 Unsupervised Learning methods

At first, we briefly introduce four unsupervised learning methods that are investigated in the format of Eq. (5.3). In what follows, we define $d_{ij} = 1$, $\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/t)$ or Consine metric as the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$.

PCA-like *SparLow*

The standard PCA method computes an orthogonal transformation $\mathbf{V} \in St(l,m)$ such that the variance of the low dimensional representations is maximized, i.e., $\mathbf{V}$ is the maximizer of the problem

$$\max_{\mathbf{V}\in St(l,m)} \operatorname{tr}\left(\mathbf{V}^\top \mathbf{X}\mathbf{H}_n\mathbf{X}^\top\mathbf{V}\right). \tag{5.16}$$

In the framework of trace quotient, the denominator can be trivially considered to be $\operatorname{tr}(\mathbf{V}^\top\mathbf{B}_{pca}\mathbf{V})$ with $B_{pca} = \operatorname{tr}(\mathbf{X}\mathbf{H}_n\mathbf{X}^\top)\mathbf{I}_n$, which is a constant. By adopting the sparse representations $\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})$, we construct straightforwardly

$$\mathcal{A}_{pca}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{H}_n(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top, \tag{5.17}$$

and

$$\mathcal{B}_{pca}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \operatorname{tr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{H}_n(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top)\mathbf{I}_k. \tag{5.18}$$

LLE-like *SparLow*

The original LLE method aims to find low dimensional representations of the data via fitting directly the barycentric coordinates of a point based on its neighbors constructed in the original data space, cf. [13]. It is well known that the low dimensional representations in the LLE method can only be computed implicitly. In order to overcome this drawback, the so-called Orthogonal Neighborhood Preserving Projections (ONPP) is developed in [194], by introducing an explicit orthogonal transformation between the original data and its low dimensional representation.

Specifically, the ONPP method solves the problem

$$\min_{\mathbf{V}\in St(l,m)} \operatorname{tr}\left(\mathbf{V}^\top \mathbf{X}\mathbf{M}\mathbf{X}^\top\mathbf{V}\right), \tag{5.19}$$

where $\mathbf{M} = (\mathbf{I}_n - \mathbf{W})^\top(\mathbf{I}_n - \mathbf{W})$ with $\mathbf{W} \in \mathbb{R}^{n\times n}$ being the matrix of barycentric coordinates of the data. Similar to the construction in the previous subsection, we construct the following functions for an LLE-like *SparLow* approach, i.e.,

$$\mathcal{A}_{lle}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{M}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top, \tag{5.20}$$

and

$$\mathcal{B}_{lle}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \operatorname{tr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{M}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top)\mathbf{I}_k. \tag{5.21}$$

Laplacian *SparLow*

Another category of DR methods are the ones involving a Laplacian matrix of the data. It includes, for example, Locality Preserving Projection (LPP) [183], Orthogonal LPP (OLPP) [189], Linear Graph Embedding (LGE) [187] , and Spectral Clustering [18]. Similar to the

approaches applied in the previous two subsections, we adapt a simple formulation by setting

$$\mathcal{A}_{lap}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{M}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top, \tag{5.22}$$

with $\mathbf{M} := \{\mathbf{m}_{ij}\} \in \mathbb{R}^{n\times n}$ being a real symmetric matrix measuring the similarity between data pairs $(\mathbf{x}_i, \mathbf{x}_j)$, and

$$\mathcal{B}_{lap}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\boldsymbol{\Xi}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top, \tag{5.23}$$

with $\boldsymbol{\Xi} := \{\boldsymbol{\xi}_{ij}\} \in \mathbb{R}^{n\times n}$ being a diagonal matrix having $\boldsymbol{\xi}_{ii} := \sum_{j\neq i} \mathbf{m}_{ij}, \forall i, j$. Specifically, the similarity matrix $\mathbf{M}$ can be computed by applying a Gaussian kernel function on the distance between two data samples, i.e., $\mathbf{m}_{ij} = d_{ij}$ if $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ are adjacent, $\mathbf{m}_{ij} = 0$ otherwise. Note that, if following up a clustering approach, the learning model Eq. (5.3) associated with Eq. (5.22) and Eq. (5.23) could be viewed as a joint learning version of sparse subspace clustering method [119].

Specifically for LPP [183] and LGE [187], it involves a generalized orthogonal constraint $\mathbf{U}^\top\mathbf{C}\mathbf{U} = \mathbf{I}_l$ with $\mathbf{C} := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\boldsymbol{\Xi}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top$ and $\mathbf{U}^\top \in \mathbb{R}^{k\times l}$. $\mathbf{C}$ is assumed to be symmetric positive definite (PSD). Therefore, we can rewrite formulations Eq. (5.22) and Eq. (5.23) as

$$\mathcal{A}_{lpp}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \mathbf{C}^{-1/2}\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{M}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top\mathbf{C}^{-1/2},$$

and

$$\mathcal{B}_{lpp}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \mathrm{tr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\boldsymbol{\Xi}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top)\mathbf{I}_k,$$

with $\mathbf{U} = \mathbf{C}^{-1/2}\mathbf{V}, \mathbf{V} \in St(l,k)$.

UDP-like *SparLow*

Unsupervised discriminant projection (UDP) [195] is one extension of Laplacian matrix related methods with the purposes of classification. It addresses to maximize the ratio of nonlocal scatter to local scatter. The nonlocal scatter and the local scatter are characterized by a nonlocal Laplacian matrix and a local Laplacian matrix, respectively.

Let us define a global kernel matrix $\mathbf{K} := \{\mathbf{k}_{ij}\} \in \mathbb{R}^{n\times n}$ with $\mathbf{k}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/t)$ or $\mathbf{k}_{ij} = 1$, and a local kernel matrix $\mathbf{M} := \{\mathbf{m}_{ij}\} \in \mathbb{R}^{n\times n}$ with $\mathbf{m}_{ij} = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/t)$ or $\mathbf{m}_{ij} = 1$ if $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ are adjacent, $\mathbf{m}_{ij} = 0$ otherwise. The local Laplacian matrix is defined by $\mathbf{L}_L := \mathbf{W}^{\mathbf{M}} - \mathbf{M}$ with a diagonal $\mathbf{W}^{\mathbf{M}}, \mathbf{W}^{\mathbf{M}}_{ii} := \sum_{j\neq i}\mathbf{m}_{ij}$ for all $i = 1, \ldots, n$. The nonlocal Laplacian matrix is defined by $\mathbf{L}_N := \mathbf{W}^{\mathbf{K}} - \mathbf{K} - \mathbf{L}_L$ with a diagonal $\mathbf{W}^{\mathbf{K}}, \mathbf{W}^{\mathbf{K}}_{ii} := \sum_{j\neq i}\mathbf{k}_{ij}$ for all $i = 1, \ldots, n$.

Similar to Laplacian *SparLow*, we construct the formulation by setting

$$\mathcal{A}_{udp}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{L}_N(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top,$$

and

$$\mathcal{B}_{udp}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{L}_L(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top.$$

5.4.2 Supervised Learning methods

In the following, we consider a set of data samples $\mathbf{X}_i = [\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in_i}] \in \mathbb{R}^{m \times n_i}$ with $i = 1, \ldots, c$, where $c > 1$ indicates the number of classes and n_i refers to the number of data samples in the corresponding i-th class. The corresponding sparse coefficients are denote by $\boldsymbol{\Phi}_i := [\boldsymbol{\phi}_{i1}, \ldots, \boldsymbol{\phi}_{in_i}] \in \mathbb{R}^{k \times n_i}$, and $\boldsymbol{\Phi} := [\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_c] \in \mathbb{R}^{k \times n}$ with $n = \sum_{i=1}^{c} n_i$.

LDA *SparLow*

The goal of LDA [186] is to find a low-dimensional representation of the high dimensional data, so that the between-class scatter is maximized, while the within-class scatter is minimized.

Let us define by $\boldsymbol{\Pi}_k := \mathbf{I}_k - \frac{1}{k}\mathbf{1}_k\mathbf{1}_k^\top$ the centring projector, and $\overline{\boldsymbol{\phi}}_i \in \mathbb{R}^k$ be the centre of the i-th class. The with-in class scatter matrix is computed as

$$\begin{aligned} \mathcal{B}_{lda}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) =& \sum_{i=1}^{c} \sum_{j=1}^{n_i} (\boldsymbol{\phi}_{ij} - \overline{\boldsymbol{\phi}}_i)(\boldsymbol{\phi}_{ij} - \overline{\boldsymbol{\phi}}_i)^\top \\ =& \sum_{i=1}^{c} \boldsymbol{\Phi}_i \boldsymbol{\Pi}_{n_i} \boldsymbol{\Phi}_i^\top \\ =& \boldsymbol{\Phi} \mathbf{L}^w \boldsymbol{\Phi}^\top, \end{aligned} \tag{5.24}$$

with

$$\mathbf{L}^w = \begin{bmatrix} \boldsymbol{\Pi}_{n_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \boldsymbol{\Pi}_{n_c} \end{bmatrix} \tag{5.25}$$

being the Laplacian matrix for intraclass samples.

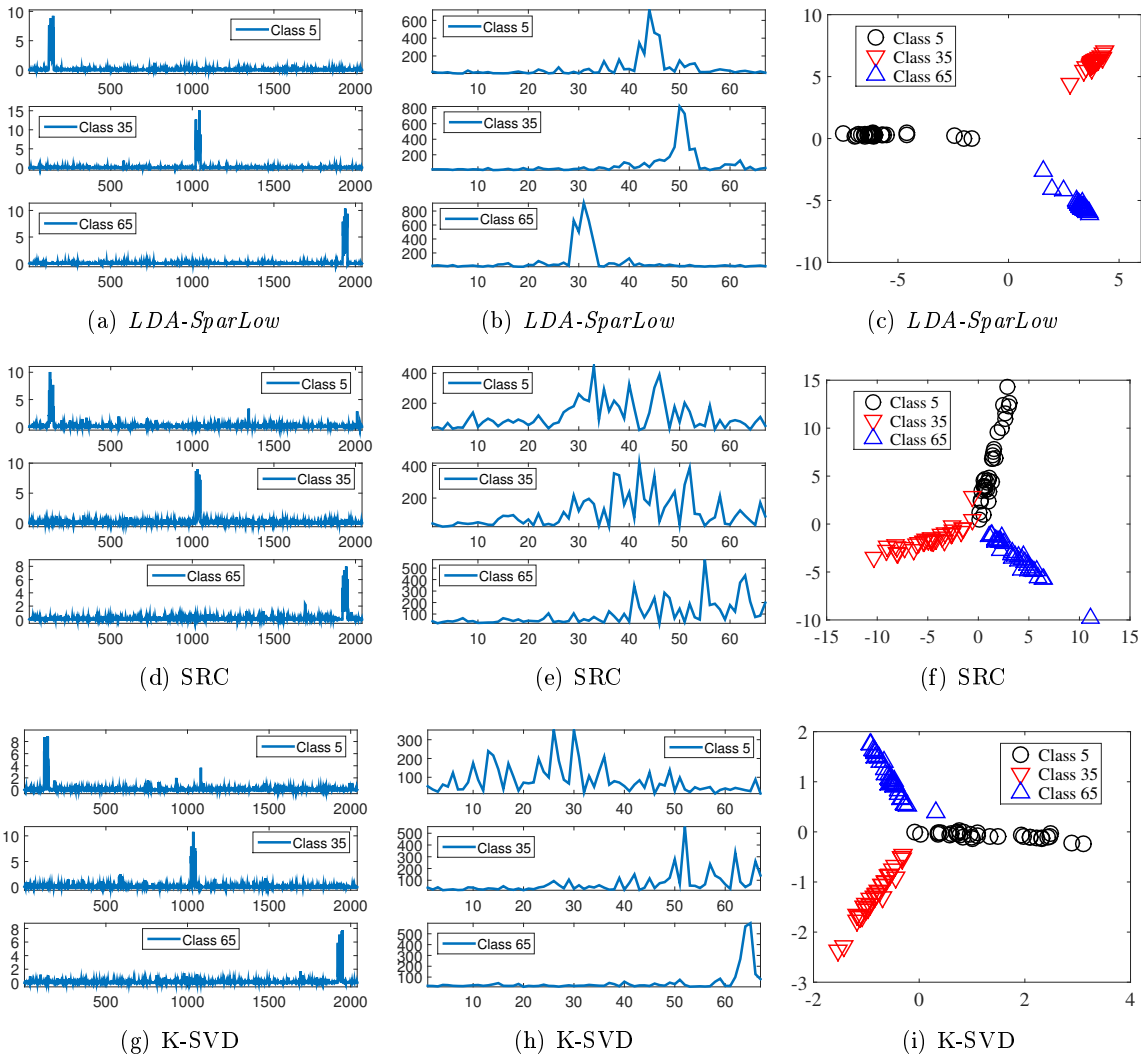
Let $\overline{\boldsymbol{\phi}} \in \mathbb{R}^k$ be the center of all classes. Similarly, we compute the between class scatter matrix as

$$\begin{aligned} \mathcal{A}_{lda}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) =& \sum_{i=1}^{c} n_i (\overline{\boldsymbol{\phi}}_i - \overline{\boldsymbol{\phi}})(\overline{\boldsymbol{\phi}}_i - \overline{\boldsymbol{\phi}})^\top \\ =& \left[\boldsymbol{\phi}_1 \tfrac{\mathbf{1}_{n_1}}{\sqrt{n_1}}, \ldots, \boldsymbol{\phi}_c \tfrac{\mathbf{1}_{n_c}}{\sqrt{n_c}}\right] \boldsymbol{\Pi}_c \left[\boldsymbol{\phi}_1 \tfrac{\mathbf{1}_{n_1}}{\sqrt{n_1}}, \ldots, \boldsymbol{\phi}_c \tfrac{\mathbf{1}_{n_c}}{\sqrt{n_c}}\right]^\top \\ =& \boldsymbol{\Phi} \mathbf{L}^b \boldsymbol{\Phi}^\top, \end{aligned} \tag{5.26}$$

with

$$\mathbf{L}^b = \begin{bmatrix} \frac{\mathbf{1}_{n_1}}{\sqrt{n_1}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\mathbf{1}_{n_c}}{\sqrt{n_c}} \end{bmatrix} \cdot \boldsymbol{\Pi}_c \cdot \begin{bmatrix} \frac{\mathbf{1}_{n_1}}{\sqrt{n_1}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\mathbf{1}_{n_c}}{\sqrt{n_c}} \end{bmatrix}^\top \tag{5.27}$$

being the Laplacian matrix for interclass samples.



(a) *LDA-SparLow* (b) *LDA-SparLow* (c) *LDA-SparLow*

(d) SRC (e) SRC (f) SRC

(g) K-SVD (h) K-SVD (i) K-SVD

Figure 5.2: Discriminative features illustration. Sparse codes, reduced fisher features and 2D fisher features using proposed *LDA-SparLow*, SRC [3] and K-SVD [4], respectively. From first column to third column, the pictures depict the sparse codes with $k = 2040$, the reduced fisher features with $l = 67$, and 2D visualization of fisher features, using *LDA-SparLow*, SRC, and K-SVD respectively. From first column to second column, each waveform indicates a sum of absolute values for different testing samples from the same class. The curves in the first, second, and third rows correspond to 5-th class, 35-th class and 65-th class.

MFA *SparLow*

Marginal Fisher Analysis (MFA) in [187], also called Linear Discriminant Embedding (LDE) in [196], is the supervised version of Linear Graph Embedding (LGE) [187]. The idea is to maintain the original neighbor relations of points from the same class while pushing apart the neighboring points of different classes.

Let $\mathbf{N}_{k_1}^{+}(\boldsymbol{\phi}_i)$ denote the set of k_1 nearest neighbors which share the same label with $\boldsymbol{\phi}_i$, and $\mathbf{N}_{k_2}^{-}(\boldsymbol{\phi}_i)$ denote the set of k_2 nearest neighbors among the data points whose labels are different to that of $\boldsymbol{\phi}_i$.

Let us define

$$\mathbf{W}_{ij}^{+} = \begin{cases} 1 \text{ or } \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/t), & \text{if } \boldsymbol{\phi}_j \in \mathbf{N}_{k_1}^{+}(\boldsymbol{\phi}_i) \text{ or } \boldsymbol{\phi}_i \in \mathbf{N}_{k_1}^{+}(\boldsymbol{\phi}_j) \\ 0, & \text{otherwise.} \end{cases}$$

and

$$\mathbf{W}_{ij}^{-} = \begin{cases} 1 \text{ or } \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2/t), & \text{if } \boldsymbol{\phi}_j \in \mathbf{N}_{k_2}^{-}(\boldsymbol{\phi}_i) \text{ or } \boldsymbol{\phi}_i \in \mathbf{N}_{k_2}^{-}(\boldsymbol{\phi}_j) \\ 0, & \text{otherwise.} \end{cases}$$

with $\boldsymbol{\Sigma}_{ii}^{+} = \sum_{j \neq i} \mathbf{W}_{ij}^{+}, \boldsymbol{\Sigma}_{ii}^{-} = \sum_{j \neq i} \mathbf{W}_{ij}^{-}, \forall i$ being diagonal.

Let us further define $\mathbf{L}^{-} = \boldsymbol{\Sigma}^{-} - \mathbf{W}^{-}$ and $\mathbf{L}^{+} = \boldsymbol{\Sigma}^{+} - \mathbf{W}^{+}$ by the Laplacian matrices for characterizing the interclass locality and the intraclass locality, respectively. Hence, we construct the following functions for an MFA-like *SparLow* approach, i.e.,

$$\mathcal{A}_{mfa}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{L}^{-} \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})^{\top}, \tag{5.28}$$

and

$$\mathcal{B}_{mfa}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{L}^{+} \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})^{\top}. \tag{5.29}$$

The similar idea is also shown in Supervised Neighborhood Embedding [183, 15], Supervised LPP in [15], Supervised NPP in [183, 15], etc.

MVR *SparLow*

In this subsection, we consider to learn low dimensional representation and task learning (e.g., multi-label classification) simultaneously. Considering the following multivariate ridge regression (MVR) model:

$$\min_{\mathbf{D}, \mathbf{V}, \mathbf{W}} \|\mathbf{Z} - \mathbf{W}^{\top} \mathbf{V}^{\top} \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})\|_F^2 + \mu \|\mathbf{W}\|_F^2, \tag{5.30}$$

where $\mathbf{Z} \in \mathbb{R}^{d \times n}$ is the target matrix, $\mathbf{V} \in St(l, k)$, $\mathbf{W} \in \mathbb{R}^{l \times d}$ and $\mu \in \mathbb{R}^{+}$. Fixed other parameters, minimizing Eq. (5.30) with respect to $\mathbf{W}$, it has a closed expression

$$\mathbf{W} = \left(\mathbf{V}^{\top} (\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))^{\top} + \mu \mathbf{I}_k) \mathbf{V}\right)^{-1} \mathbf{V}^{\top} \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{Z}^{\top}.$$

Using this closed expression to substitute the $\mathbf{W}$ in Eq. (5.30), we can rewrite Eq. (5.30) as the format of Eq. (5.3) with

$$\mathcal{A}_{mvr}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) := -\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{Z}^{\top} \mathbf{Z} \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})^{\top}, \tag{5.31}$$

and

$$\mathcal{B}_{mvr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \left(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})^{\top} + \mu\mathbf{I}_k\right). \tag{5.32}$$

Therein, $\mathbf{Z}$ could be the binary class labels of input signals, which is usually coded as $\mathbf{Z} \in \mathbb{R}^{c\times n}$ with $\mathbf{Z}_i = (\mathbf{Z}_{i1}, \cdots, \mathbf{Z}_{ic})^{\top}$, $\mathbf{Z}_{ij} = 1$ if $\mathbf{Z}_i$ is in class c_j, $\mathbf{Z}_{ij} = 0$ otherwise. $\mathbf{Z}$ is also could be some handcrafted indicator matrix according to labels, e.g., the "discriminative" sparse codes in [197, 109]. Some typical multivariate regression approaches, such as Orthogonal Partial Least Squares (OPLS), and Joint DR with Hinge Loss, cf. [197], could be modeled as their sparse formats according to Eq. (5.3).

5.4.3 Semi-supervised Learning methods

We now introduce that the *SparLow* Model is also well suited to exploit unlabeled data in a semi-supervised setting. In this section, we consider that we only have partially $n_{\mathcal{L}}$ labeled observed points and $n_{\mathcal{U}}$ unlabeled points, i.e., $\mathbf{X} := [\mathbf{X}^{\mathcal{L}} \in \mathbb{R}^{m\times n_{\mathcal{L}}}, \mathbf{X}^{\mathcal{U}} \in \mathbb{R}^{m\times n_{\mathcal{U}}}]$ with $n = n_{\mathcal{L}} + n_{\mathcal{U}}$.

The first assumption to support *semi-supervised* SparLow *model* is that the learned dictionary for specific class is also effective for learning good sparse features from unlabeled data, cf. [123, 26]. Second, we follow the way that learning semi-supervised DR settings associated with preserving the global data manifold structure, namely, nearby points will have similar lower-dimensional representations [190, 198] or labels [199, 200, 201]. For the whole dataset $\mathbf{X}$, let us define the graph Laplacian matrix $\mathbf{L} = \boldsymbol{\Xi} - \mathbf{M}$ in $\mathbb{R}^{n\times n}$ where $\mathbf{M} := \{\mathbf{m}_{ij}\}$ with $\mathbf{m}_{ij}$ weighting the edge between adjacency data pairs $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$, $\mathbf{m}_{ij} = 0$ otherwise. $\boldsymbol{\Xi} := \{\boldsymbol{\xi}_{ij}\}$ is diagonal and $\boldsymbol{\xi}_{ii} := \sum_{j\neq i} \mathbf{m}_{ij}, \forall i, j$.

Semi-supervised LDA *SparLow*

For labeled dataset $\mathbf{X}^{\mathcal{L}}$, we adopt the criterion of LDA and compute matrices $\mathbf{L}^w$ and $\mathbf{L}^b$ being same to section 5.4.2. Hence the total scatter matrix can be written as $\mathcal{S}_t(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}})\mathbf{L}^t\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}})^{\top}$ with $\mathbf{L}^t = \mathbf{L}^w + \mathbf{L}^b$ in $\mathbb{R}^{n_{\mathcal{L}}\times n_{\mathcal{L}}}$. Similar to the constructions of section 5.4.2, we construct the formulations of Semi-supervised LDA *SparLow* as

$$\mathcal{A}_{slda}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\widetilde{\mathbf{L}}^b(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^{\top}, \tag{5.33}$$

and

$$\mathcal{B}_{slda}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X})(\widetilde{\mathbf{L}}^t + \alpha\mathbf{L})(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^{\top}, \tag{5.34}$$

where $\alpha \in \mathbb{R}$ controls the influence of labeled Laplacian matrix $\widetilde{\mathbf{L}}^t \in \mathbb{R}^{n\times n}$ and global Laplacian matrix $\mathbf{L} \in \mathbb{R}^{n\times n}$, and $\widetilde{\mathbf{L}}^b$ and $\widetilde{\mathbf{L}}^t$ are the augmented matrices of $\mathbf{L}^b$ and $\mathbf{L}^t$, namely,

$$\widetilde{\mathbf{L}}^b = \begin{bmatrix} \mathbf{L}^b & \mathbf{0}_{n_{\mathcal{U}}\times n_{\mathcal{U}}} \\ \mathbf{0}_{n_{\mathcal{U}}\times n_{\mathcal{U}}} & \mathbf{0}_{n_{\mathcal{U}}\times n_{\mathcal{U}}} \end{bmatrix},$$

and

$$\widetilde{\mathbf{L}}^t = \begin{bmatrix} \mathbf{L}^t & \mathbf{0}_{n_{\mathcal{U}} \times n_{\mathcal{U}}} \\ \mathbf{0}_{n_{\mathcal{U}} \times n_{\mathcal{U}}} & \mathbf{0}_{n_{\mathcal{U}} \times n_{\mathcal{U}}} \end{bmatrix}.$$

Since the admissible set of the projection matrix $\mathbf{P}$ of our proposed SDA *SparLow* is well defined on $Gr(l,k)$, SDA *SparLow* could combine the many extensions of SDA, such as Trace Ratio LDA [202] and Trace Ratio Based Flexible SDA [203, 204, 201].

Semi-supervised MFA *SparLow*

We now consider the semi-supervised version of Laplacian *SparLow*. For the whole dataset $\mathbf{X}$, let us define the nonlocal graph Laplacian matrix $\mathbf{L}^N = \boldsymbol{\Xi}^N - \mathbf{M}^N$ in $\mathbb{R}^{n \times n}$ with $\mathbf{m}_{ij}^N$ weighting the edge between non-adjacency data pairs $(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$, $\mathbf{m}_{ij}^N = 0$ otherwise. $\boldsymbol{\Xi}^N := \{\boldsymbol{\xi}_{ij}^N\}$ is diagonal and $\boldsymbol{\xi}_{ii}^N := \sum_{j \neq i} \mathbf{m}_{ij}^N, \forall i, j$. For the labeled dataset $\mathbf{X}^{\mathcal{L}}$, we adopt the setting of section 5.4.2 and define interclass local Laplacian matrix $\mathbf{L}^- \in \mathbb{R}^{n_{\mathcal{L}} \times n_{\mathcal{L}}}$ and intraclass local Laplacian matrix $\mathbf{L}^+ \in \mathbb{R}^{n_{\mathcal{L}} \times n_{\mathcal{L}}}$. Hence, we construct the formulations of Semi-supervised Laplacian *SparLow* as

$$\mathcal{A}_{slap}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})(\widetilde{\mathbf{L}}^- + \alpha_1 \mathbf{L}^N)(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))^\top,$$

and

$$\mathcal{B}_{slap}(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X})(\widetilde{\mathbf{L}}^+ + \alpha_2 \mathbf{L})(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))^\top$$

with $\alpha_1 \in \mathbb{R}^+, \alpha_2 \in \mathbb{R}^+$ control the influence of labeled Laplacian matrix and unlabeled Laplacian matrix. Similar to the setting of Section 5.4.3, let $\widetilde{\mathbf{L}}^- \in \mathbb{R}^{n \times n}$ and $\widetilde{\mathbf{L}}^+ \in \mathbb{R}^{n \times n}$ denote the augmented matrices of $\mathbf{L}^-$ and $\mathbf{L}^+$.

Semi-supervised MVR *SparLow*

The supervised linear (label-based) regression (e.g., SVM) associated with a manifold regularization cf. [200, 201], is another one popular framework for resolving semi-supervised learning problem. In this section, we adopt a MVR model associated with a manifold regularization as

$$\begin{aligned} \min_{\mathbf{D},\mathbf{V},\mathbf{W}} & \|\mathbf{Z}^{\mathcal{L}} - \mathbf{W}^\top \mathbf{V}^\top \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}^{\mathcal{L}})\|_F^2 + \rho_1 \|\mathbf{W}\|_F^2 \\ & + \rho_2 \operatorname{tr}\left(\mathbf{W}^\top \mathbf{V}^\top \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{L} (\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))^\top \mathbf{V} \mathbf{W}\right), \end{aligned} \tag{5.35}$$

in which $\mathbf{Z}^{\mathcal{L}} \in \mathbb{R}^{d \times n_{\mathcal{L}}}$ is the target matrix for $\mathbf{X}^{\mathcal{L}}$, $\mathbf{V} \in St(l,k)$, $\mathbf{W} \in \mathbb{R}^{l \times d}$ and $\rho_1, \rho_2 \in \mathbb{R}^+$. Fixed other parameters, minimizing Eq. (5.35) with respect to $\mathbf{W}$, it has a closed expression

$$\begin{aligned} \mathbf{W} = & \big(\mathbf{V}^\top (\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}^{\mathcal{L}})(\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}^{\mathcal{L}}))^\top + \rho_1 \mathbf{I}_k \\ & + \rho_2 \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}) \mathbf{L} (\boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}))^\top) \mathbf{V}\big)^{-1} \mathbf{V}^\top \boldsymbol{\Phi}(\mathbf{D}, \mathbf{X}^{\mathcal{L}})(\mathbf{Z}^L)^\top. \end{aligned}$$

Using this closed expression to substitute the $\mathbf{W}$ in Eq. (5.35), we can rewrite Eq. (5.35) as the format of Eq. (5.3) with

$$\mathcal{A}_{smvr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := -\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}})(\mathbf{Z}^{\mathcal{L}})^{\top}\mathbf{Z}^{\mathcal{L}}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}}))^{\top},$$

and

$$\mathcal{B}_{smvr}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})) := \boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}})(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}^{\mathcal{L}}))^{\top} + \rho_1\mathbf{I}_k + \rho_2\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{L}(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^{\top}.$$

5.5 Experimental Evaluations

In this section, we investigate the performance of our proposed *SparLow* methods on real image data. We apply the *SparLow* methods to firstly learn low dimensional representations of real images, and then evaluate their performance in several aspects, such as (1) visualization and clustering, (2) the object/scene categorization using known class labels, (3) 2/3D visualization of disentangling factors learned by applying the *SparLow*, or parameters sensitivity.

5.5.1 Experimental Settings

In the following of the paper, we refer to the proposed *SparLow* methods proposed in Section 5.4.1, 5.4.1, 5.4.1, 5.4.1, 5.4.2, 5.4.2, 5.4.2, 5.4.3, 5.4.3 and 5.4.3 as *PCA-SparLow*, *LLE-SparLow*, *Lap-SparLow*, *UDP-SparLow*, *LDA-SparLow*, *MFA-SparLow*, *MVR-SparLow*, *SDA-SparLow*, *SMFA-SparLow* and *SMVR-SparLow*, respectively. Similarly, we refer to direct applications of the classic DR methods on sparse representations that are generated with respect to a fixed dictionary as *SparLDR*. Seven members of the *SparLDR* family are investigated in our experiments, namely, *SparPCA*, *SparOLPP*, *SparONPP*, *SparUDP*, *SparLDA*, *SparMFA*, *SparMVR*, *SparSLDA*, *SparSMFA* and *SparSMVR*, as the ten corresponding counterparts of the *SparLow*.

In our experiments with unsupervised setting, dictionaries are initialized as a column-wise normalized Gaussian matrix and then improved by employing the K-SVD algorithm [4]. In the cases of supervised and semi-supervised settings, we use K-SVD to learn a sub-dictionary for each class, and then combine all the sub-dictionaries as a shared dictionary $\widehat{\mathbf{D}}$. The learned dictionaries $\widehat{\mathbf{D}}$'s are used in the regularizer h, as defined in (5.5). Once an initial dictionary $\widehat{\mathbf{D}}$ is given, the orthogonal projection $P \in Gr(l,k)$ can be obtained by applying classical DR methods on the sparse representations. However, when the size of the training dataset is huge, directly performing classical DR methods is often prohibitive. In order to overcome this difficulty, we propose to randomly select a relatively small number of samples, and then to employ the classical DR methods on their sparse representations to obtain an estimation of the initial orthogonal projection $\mathbf{P}_0 \in Gr(l,k)$.

Throughout all experiments, we consistently set $\sigma = 10^{-3}$ in Eq. (5.3). We treat each image as an m-dimensional vector, and normalize it into a unit ball. Let n be the number of all signals which contain c classes, we use n_{train}, n_{test} to denote the number of total training

samples and the number of total testing samples for each class, respectively. Usually, we set $n_i = n_{\text{train}} + n_{\text{test}}$, $n = \sum_{i=1}^{c} n_i$ with n_i being the number of samples from i^{th} class. In addition, we denote $n_{\text{train}}^{\mathcal{L}}$ by the number of labeled samples from a training set. For datasets without standard division of training set and testing set, all recognition experiments are repeated ten times with different randomly selected training and test subsets, and the average of per-class recognition rates is recorded for each run.

By default, we employ elastic net method to solve the sparse coding problem (3.16). Note that, we also adapt other sparse coding methods, such as learning sparse based on KL-divergence, where we will give special annotations.

5.5.2 Evaluation of Unsupervised *SparLow*

Handwritten digit images

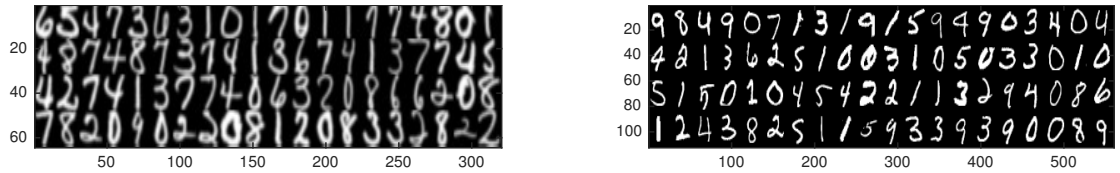
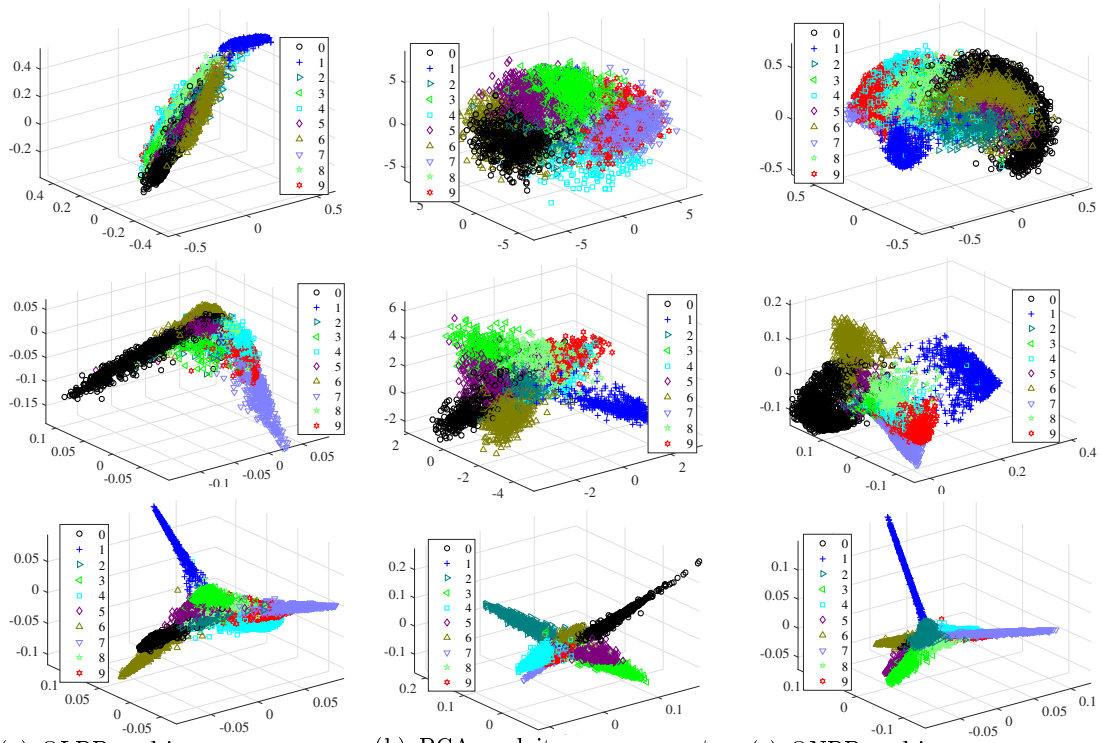


Figure 5.3: Digital databases. The left images set is from USPS dataset, and right one is from MNIST database.

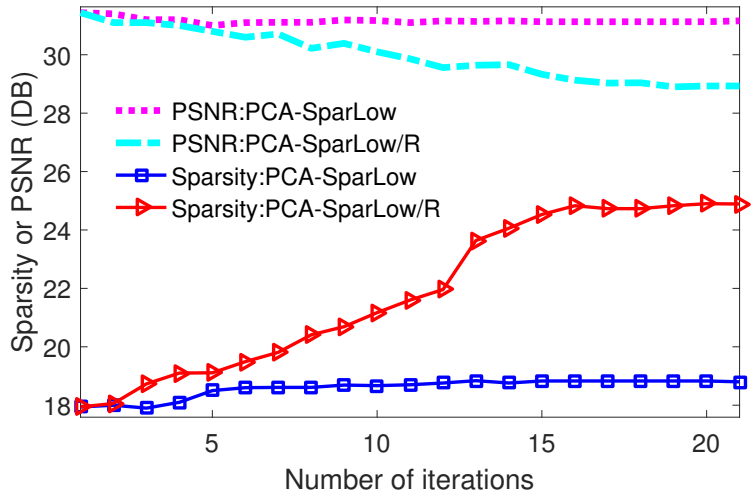
Table 5.1: Classification Performance (Accuracy (%)) for the MNIST & USPS datasets of the Proposed *SparLow* methods, with comparisons to some classical unsupervised DR approaches.

Methods	USPS (1NN)	USPS (GSVM)	MNIST (1NN)	MNIST (GSVM)
PCA [125]	86.40%, $l = 50$	92.43%, $l = 50$	84.62%, $l = 50$	94.63%, $l = 50$
OLPP [189]	84.11%	91.48%	83.12%	94.76%
ONPP [194]	87.39%	92.73%	85.01%	95.21%
KPCA [15]	89.19%, $l = 50$	93.27%, $l = 50$	–	–
LLE [13]	68.81%	90.43%	66.09%	93.11%
LE [15]	71.85%	91.93%	68.16%	93.90%
ISOMAP [15]	64.80%	90.13%	60.51%	91.67%
CS-PCA [125]	87.84%	94.22%	87.65%	96.04%
PCA-SparLow	92.18%, $l = 50$	96.82%, $l = 50$	91.23%, $l = 50$	97.12%
Lap-SparLow	91.83%	96.26%	89.32%	96.91%
LLE-SparLow	90.78%	96.16%	89.10%	96.93%
UDP-SparLow	92.85%	96.18%	90.10%	97.12%

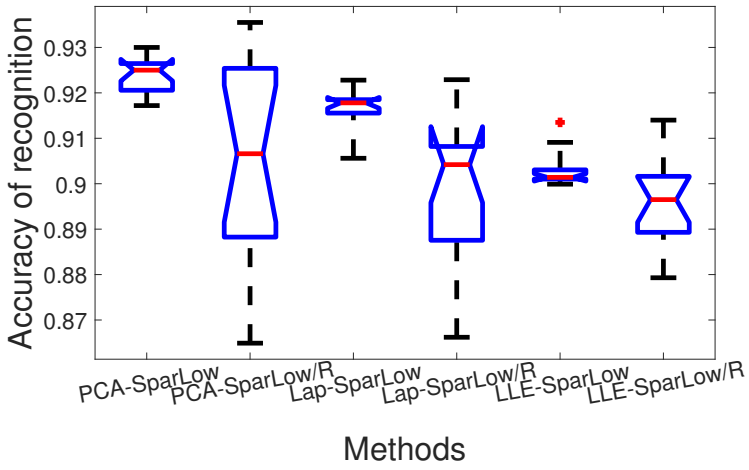


(a) OLPP and its sparse counterparts (b) PCA and its sparse counterparts (c) ONPP and its sparse counterparts

Figure 5.4: 3D visualization using OLPP, PCA and ONPP on USPS handwritten digits. From top to bottom: Applying OLPP/PCA/ONPP in original space, in sparse space with respect to initial dictionary $\widehat{\boldsymbol{D}}$, and in sparse space with respect to learned dictionary via *SparLow*, respectively.



(a) PSNR and Sparsity per image in learning *PCA-SparLow*



(b) Box plot of 1NN classification with or without Regularizations

Figure 5.5: Performing *SparLow* with or without developed regularizations on USPS database. *PCA-SparLow/R* denotes *PCA-SparLow* without regularizations, and in same way to *Lap-SparLow/R* and *LLE-SparLow/R*.

Our first experiment is performed on the handwritten digits from the MNIST database [1] and the USPS [205]. The MNIST database consists of $60,000$ handwritten digits images for training and $10,000$ digits images for testing. All images are grayscale between 0 and 1 and have a uniform size of 28×28 pixels. The USPS database has $7,291$ training images and $2,007$ testing images of size (16×16). Some examples are shown in Fig. 5.3. By vectorising the pixel intensity values of the images, each image is represented as a vector of dimension $m = 784$ or $m = 256$ for the MNIST database and the USPS database, respectively.

[1] http://yann.lecun.com/exdb/mnist/

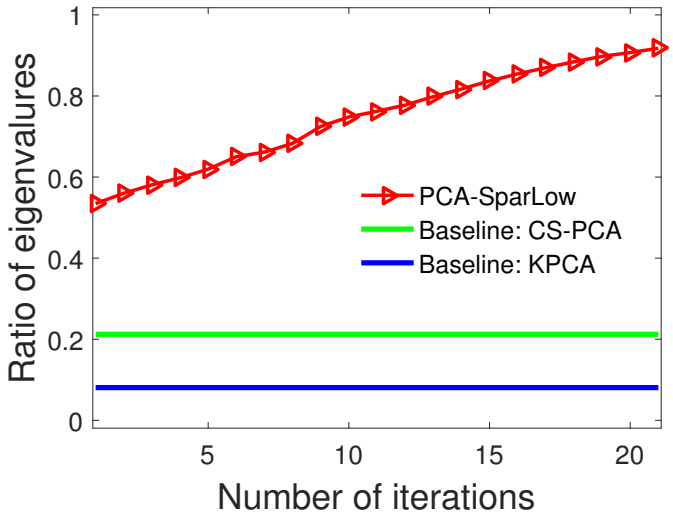
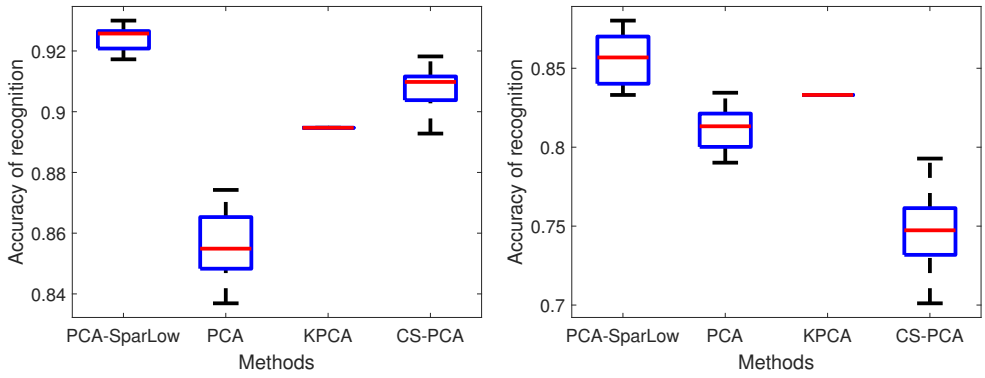


Figure 5.6: Ratio of top l largest eigenvalues against all eigenvalues in learning process of *PCA-SparLow*.



(a) USPS (b) MNIST

Figure 5.7: Comparison of 1NN classification using *PCA-SparLow*, PCA, KPCA, CS-PCA on MNIST & USPS database. Dictionary size is 1000.

In this experiment, the parameters for elastic net are set to be $\lambda_1 = 0.2$, $\lambda_2 = 2 \times 10^{-5}$, and $\mu_1 = 5 \times 10^{-3}$, $\mu_2 = 2.5 \times 10^{-4}$, for both experiments on the MNIST and USPS datasets. The size of dictionary is chosen to be $k = 1000$. For CS-PCA [125], we employ one common strategy of randomly choosing a certain number of data points as dictionary in a given set of training data, cf. [3].

To demonstrate the effectiveness of the proposed algorithms, experiments of 3D visualization were conducted on the USPS dataset, compared to the classic DR methods and the SparLDR methods, see Fig. 5.4. It is easily seen that the low dimensional representations captured in the original data space, shown in the first row in Fig. 5.4, are very hard to cluster

or group. In particular, the boundary between each pair of digits are completely entangled. Direct applications on the sparse representations for a given dictionary, i.e., the second row in Fig. 5.4, show a significant improvement in disentangling the class information. Furthermore, it is evidentially clear that visualization powered by the *SparLow*, i.e., the third row in Fig. 5.4, leads to direct clustering of the handwritten digits.

Let us denote by δ_i the i^{th} largest eigenvalue of $\boldsymbol{\Phi}(\mathbf{D},\mathbf{X})\mathbf{H}_n(\boldsymbol{\Phi}(\mathbf{D},\mathbf{X}))^\top$, and further define "Ratio of eigenvalues" in Fig. 5.6 as $t_l = \sum_{i=1}^{l}\delta_i / \sum_{j=1}^{r}\delta_j$. Fig. (5.6) shows that our proposed *PCA-SparLow* significantly increase the ratio t_l. It also can be seen, our t_l are consistently larger than those of CS-PCA and KPCA, which indicates that the *PCA-SparLow* method captures more structure information, which preserves power in the l dimensional subspace, cf. [189].

One obvious benefit of the proposed *SparLow* model is that the learned low dimensional representations share both reconstructive and discriminative capacities. In this experiment, after applying the *SparLow* methods on the images from the USPS database, we employ the 1NN method to classify the reduced features. Reconstruction errors in terms of Peak Signal-to-Noise Ratio (PSNR) are presented in Fig. 5.5(a). Fig. 5.5(b) shows the box plot of results of applying the 1NN classification ten times on the USPS database with random initialisations. It is clear that the regulariser h, defined in Eq. (5.5) has the capability of ensuring good reconstruction, and achieving stable discriminations.

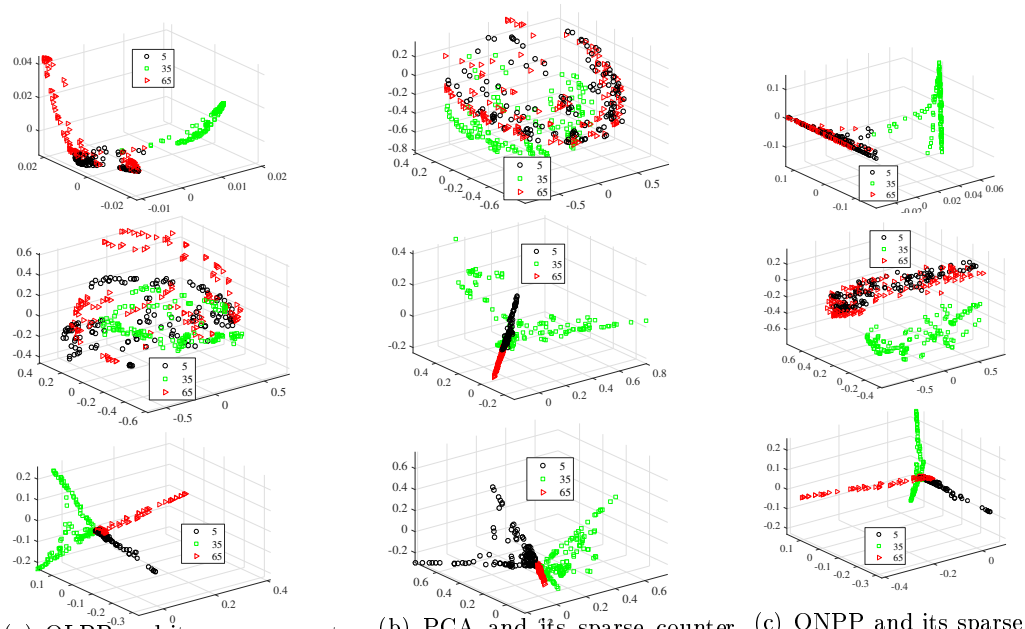
Finally, we compare the *SparLow* methods to several state of the art methods, on the task of 1NN and Gaussian SVM (GSVM) classification. For PCA, KPCA and *PCA-SparLow*, we set $l = 50$, for other methods, we set $l = 20$. For USPS, we use the full training and testing database. For MNIST, we randomly choose $30,000$ images for training, and use standard $10,000$ testing database. According to Fig. 5.7 and Table 5.1, it is obvious that the *SparLow* methods consistently outperform the state of the arts.

Figure 5.8: Face databases. The top line images set is from CMU PIE dataset, and the bottom one is from Yale B database.

CMU PIE faces analysis

In this subsection, we test the *SparLow* methods on the CMU PIE face database [206]. The CMU PIE face database contains 68 human subjects with $41,368$ face images. As suggested in [206], a subset containing $11,554$ PIE faces are chosen, all of which are manually aligned and cropped, thus we nearly get 170 images for each individual, with the scale 32×32 and 256 gray levels per pixel. All the face images are manually aligned and cropped, as shown in top row images in Fig. 5.8.

All experiments are repeated ten times with different randomly selected training and test



(a) OLPP and its sparse counterparts (b) PCA and its sparse counterparts (c) ONPP and its sparse counterparts

Figure 5.9: 3D visualization using OLPP, PCA and ONPP on PIE faces. From top to bottom: Applying OLPP/PCA/ONPP in original space, in sparse space with respect to initial dictionary, and in sparse space with respect to learned dictionary, respectively.

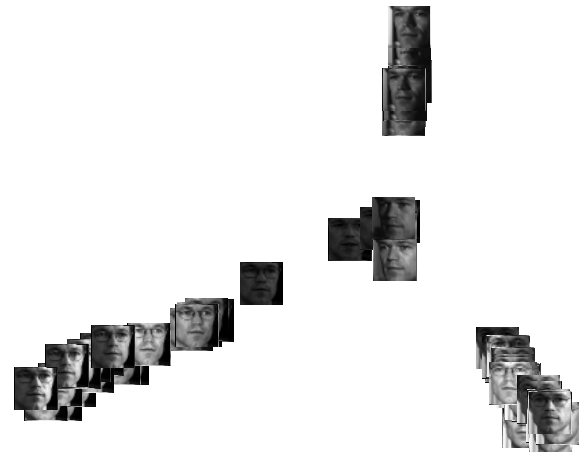
images, and the average of per-class recognition rates is recorded for each run. In our experiments, we set $\lambda_1 = 10^{-2}, \lambda_2 = 10^{-5}$, $\mu_1 = 2.5 \times 10^{-4}, \mu_2 = 5 \times 10^{-3}$.

First of all, similar to the experiments conducted on the handwritten digits, Fig. 5.9 gives the 3D visualization of low dimensional representations learned by the *SparLow* methods and their classical counterparts. It unveils a same message that the *SparLow* methods can disentangle the class information very clearly.

Then we perform 3D visualization on PIE faces without class information. We choose choose 70 faces from the class 5 with 3 variations, i.e., poses, illumination, and with/without glasses. As can be seen from the Fig. 5.10, the information referred to poses and illumination could be clearly disentangled, but the information related to glasses is roughly entangled. In Fig. 5.10(a), from left to right, the illumination become stronger. From top to bottom, the poses of faces change from left to right. The similar results are also shown in Fig. 5.10(b) and Fig. 5.10(c).

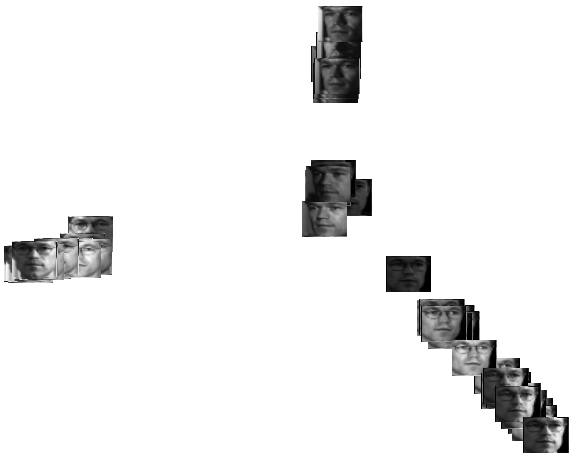
Fig. 5.11 illustrates the performance of *LDR*, *SparLDR* and *SparLow* in terms of recognition accuracy. It is easily seen that the *SparLow* methods outperform the state of the art algorithms, such as PCA, OLPP and ONPP.

Moreover, visualizing the facial features is a common approach to assess the performance



(a) *PCA-SparLow.*

(b) *OLPP-SparLow*



(c) *ONPP-SparLow*

Figure 5.10: 2D visualization of PIE faces (class 5). Applying OLPP/PCA/ONPP in sparse space with respect to learned dictionary by *SparLow*, respectively.

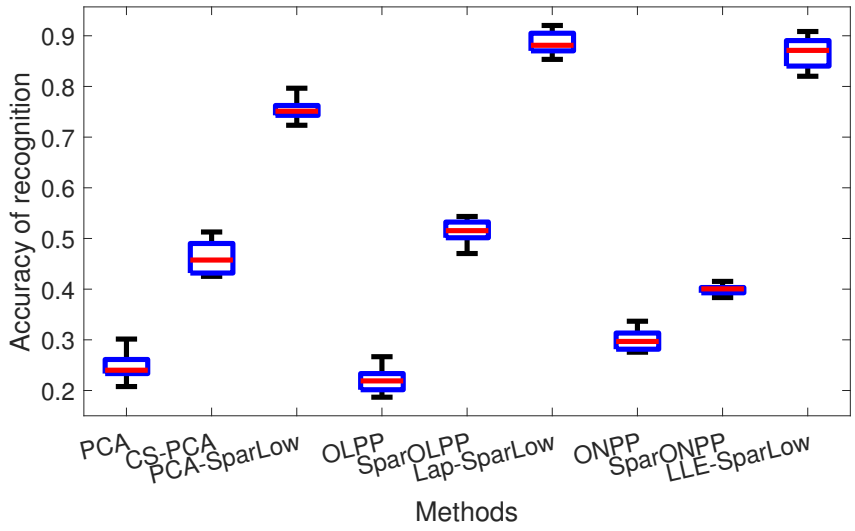


Figure 5.11: Face recognition on 68 class PIE faces. The classifier is 1NN. Randomly choose 8160 training samples and 3394 testing samples.

of DR methods. In order to facilitate this task, we define the j^{th} disentangling factor $\boldsymbol{v}_j$ as

$$\boldsymbol{v}_j = \mathbf{D}\mathbf{v}_j \in \mathbb{R}^m, \tag{5.36}$$

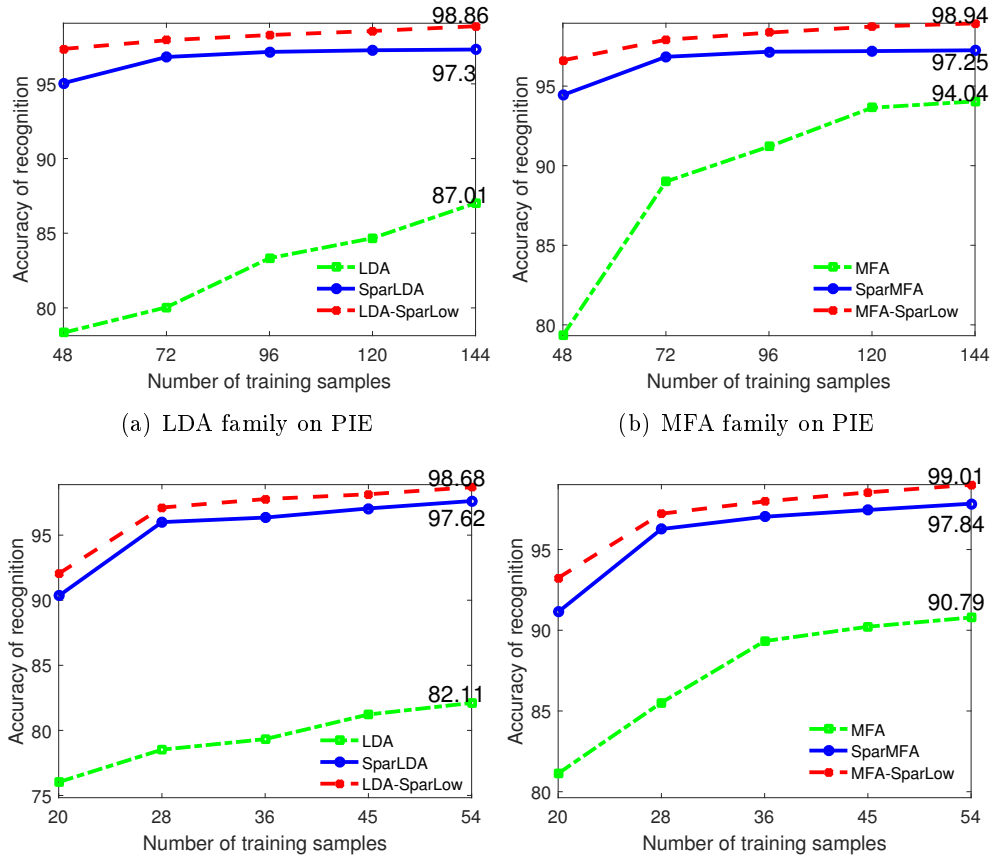
with $\mathbf{v}_j$ being the j^{th} column vector of projection matrix $\mathbf{V}$. This construction is similar to the concept of eigenfaces in [186], laplacianfaces in [183], orthogonal laplacianfaces in [189], and orthogonal LLEfaces in [13]. Fig. 5.1(b) gives the first 10 basis vectors of learned disentangling factors for *PCA-SparLow*, *Lap-SparLow* and *LLE-SparLow*. As for comparison, Fig. 5.1(a) shows the first 10 eigenfaces, laplacianfaces, and LLEfaces. It shows that (i) our learned facial features are more prominent, especially for laplacianfaces and LLE faces, (ii) our learned facial features captures richer information, such as varying pose and expression (e.g., smile).

5.5.3 Evaluation of Supervised *SparLow*

Faces Analysis

The experimentations reported here were performed on the CMU PIE face database [206] and extended Yale B database [207]. The information on CMU PIE face database has been introduced in Section 5.5.2. The Extended Yale-B face database contains 16128 images of 38 human subjects under 9 poses and 64 illumination conditions, as shown in bottom line of Fig. 5.8. In this experiment, we follow the setting of [208, 206] and choose the frontal pose and use all the images under different illumination, thus we get 64 images for each person with the resized scale 32×32.

We first compare the 1NN recognition performance of supervised DR in original domain and in sparse domain, shown in Fig. 5.12. We use LDA and MFA as the representatives



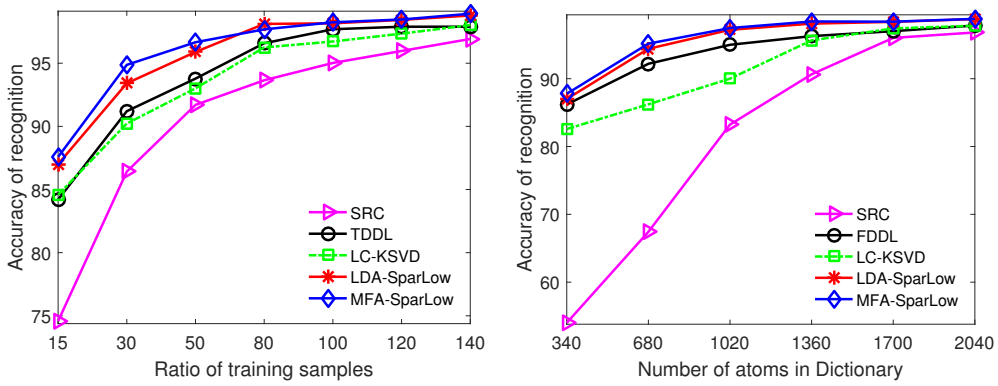
(a) LDA family on PIE (b) MFA family on PIE

(c) LDA family on Yale B (d) MFA family on Yale B

Figure 5.12: Performing the DR in original domain, *SparDR* and *SparLow* in sparse domain. The dictionary size $k = 2040, 1140$ for PIE and Yale-B, respectively. The classifier is 1NN.

Table 5.2: Training and testing computation time. m: minuet, ms

Methods	k	Training time (m)	Testing time (ms)	Accuracy (%)
FDDL [41]	2040	234.1	16.89	98.0
TDDL [26]	300 $\times$ 68	56.4	210.12	97.89
LC-KSVD[109]	2040	19.2	0.81	98.02
SRC [3]	6800	–	78.44	97.36
LDA-SparLow	2040	29.5	1.08	98.68 $\sim$ 98.88
MFA-SparLow	2040	28.3	0.96	98.82 $\sim$ 98.96
MVR-SparLow	2040	22.5	0.75	97.86 $\sim$ 98.06



(a) Comparison with different number of training samples

(b) Comparison with different dictionary size

Figure 5.13: Comparison on recognition results with different number of training samples and different dictionary size for PIE faces. The classifier is 1NN.

to test *SparLDA*, *LDA-SparLow*, *SparMFA*, and *MFA-SparLow* on PIE and Yale B faces dataset. For PIE faces, we choose $n_{\text{train}} = 48, 72, 96, 120, 144$, respectively, and the rest for testing. In the same way, for Yale B faces, we choose $n_{\text{train}} = 20, 28, 36, 45, 54$. The results show that (i) if $\widehat{\mathbf{D}}$ is learned K-SVD, applying DR (LDA and MFA) in sparse domain, i.e., *SparLDA* and *SparMFA*, have much better performance than DR in original domain; (ii) our proposed *LDA-SparLow* and *MFA-SparLow* could achieve much improvement on classification compared with *SparLDA* and *SparMFA* based on $\widehat{\mathbf{D}}$.

Fig. 5.13(a) shows the comparison between the proposed *LDA-SparLow*, *MFA-SparLow* with other famous dictionary learning based classification methods, such as SRC [3], TDDL [26, 129], and LC-KSVD [109]. In the rest of the paper, we apply TDDL ourselves using "one-versus-all" strategy with logistic regression. The dictionary size of TDDL for each class is fixed as $k = 300$ as recommended in [26]. Considering the fairness, the size of dictionary in SRC, LC-KSVD is same proposed *LDA-SparLow*. In Fig. 5.13(a), our proposed *LDA-SparLow* and *MFA-SparLow* show the strong competitive performance compared with SRC, TDDL, and LC-KSVD. Especially, when the training sample is limited, our methods perform much better. Fig. 5.13(b) plots the recognition rates of *LDA-SparLow*, *MFA-SparLow*, SRC, FDDL [41], and LC-KSVD with varying dictionary sizes (number of atoms). In all cases, the proposed methods perform better than SRC and FDDL, and give significant improvement to LC-KSVD and TDDL. This demonstrates that learning a compact and representative dictionary could highly improve the images recognition. By applying LDA on sparse coefficients learned via SRC, K-SVD and *LDA-SparLow*, 3D visualization of low dimensional representations is depicted in Fig. 5.2. It shows *LDA-SparLow* has more strong performance on disentangling the class information, in comparison of SRC and K-SVD.

When running on a 64-bit computer with double 3.5G HZ processors, Table (5.2) demonstrates the computation times for training the models and classifying one testing PIE face using *LDA-SparLow*, *MFA-SparLow*, *MVR-SparLow*, SRC, FDDL, LC-KSVD, and TDDL.

We set the threshold for accuracy rate is greater than 97% with $n_{\text{train}} = 120$, and k is the dictionary size. It shows that our methods take the less training time and testing time, but achieve a higher recognition accuracy.

Handwritten digits images classification

We then perform the proposed *SparLow* on the handwritten digits from the USPS [205] and MNIST database, see Fig. 5.3. For MNIST digits, we compare *SparLow* with some state-of-the-art DL methods, SDL [129] 98.95%, TDDL [26] 99.46%, and some deep leaning approaches: DCNN [209] 99.38% and DBN [140] 98.80% with two hidden layers. For MNIST

Table 5.3: Classification Performance for the MNIST & USPS datasets of the Proposed methods, *LDA-SparLow*, *MFA-SparLow* and *MVR-SparLow*, with comparisons to approaches from the literature.

Methods & MNIST	Accuracy (%)	Methods & USPS	Accuracy (%)
SVM	98.60	SVM-Gauss	95.80
KNN	95.00	KNN	94.80
SRC	96.80	SRC [3]	93.95
SDL[129]	98.95	SDL[129]	96.46
TDDL [26]	99.46	TDDL [26]	97.16
DCNN [209]	99.38	FDDL[41]	97.66
DBN [140]	98.80	DBN [140]	96.51
cFA [38]	99.11	JDL [210]	93.92
LDA-SparLow	98.92 $\sim$ 99.28	*LDA-SparLow*	97.57 $\sim$ 97.66
MFA-SparLow	98.80 $\sim$ 99.02	*MFA-SparLow*	97.52
MVR-SparLow	98.10 $\sim$ 98.12	*MVR-SparLow*	97.24

and USPS digits, we first use training samples in each category for training the subdictionary and then merge them as $\widehat{\mathbf{D}}$. In this experiment, the parameters for elastic net $\lambda_1 = 0.2, \lambda_2 = 2 \times 10^{-5}$, $\mu_1 = 5 \times 10^{-3}, \mu_2 = 2.5 \times 10^{-4}$ in (5.6), for both experiments on MNIST and USPS.

For MNIST dataset, our proposed *LDA-SparLow+NN* and *MFA-SparLow+NN* achieves 99.28%, 99.02% at peak and converged onto 99.07%, 98.80% at average, respectively. Compared with some state-of-the-art approaches on MNIST, our model shows strong competitive performance, and very close to the best results published on MNIST, such as TDDL [26] 99.46% and DCNN [209] 99.38%. It should be pointed out that TDDL and DLSI is the class-specific dictionaries learning approaches, i.e., learning dictionaries and projections for each class, while *LDA-SparLow* only learns a single global dictionary and a fixed rank projection for all the testing data. Similarly for the USPS, we set $\lambda_1 = 0.1$, and $\lambda_2 = \lambda_1/(10^4)$. *LDA-SparLow+NN* and *MFA-SparLow+NN* harvest the peak results as 97.66%, 97.48%, deleting the unstable values, we finally get averages 97.62%, 97.48%, respectively. To the best of our knowledge, this stable accuracy rates almost outperform all the existing results.

Cartoon images classification

COIL100 [211] is a famous color 3D shape dataset which consists of 100 objects (72 images per object). The images of each object were taken 5° apart as the object is rotated on a turntable, as shown in Fig. 5.14. We use the cropped gray scale images of size 32×32 and each image is represented by a 289-dimensional vector through the first order wavelet transformation. The size of dictionary is set as $k = 1000$, and $\lambda_1 = 6 \times 10^{-2}, \lambda_2 = 10^{-5}$, $\mu_1 = 2.5 \times 10^{-4}, \mu_2 = 5 \times 10^{-3}$. By taking 50 images per class for training and after 20 iterations, our *LDA-SparLow* model achieves an accuracy rate $97.14\% \sim 97.20\%$ for *LDA-SparLow*, and $97.88 \sim 98.05\%$ for *MFA-SparLow*. We also compare the performance with some state-of-the-art methods, such as Standard CNN [212] 79.77%, VTU [213] 89.90%, videoCNN [213] 92.50%, and NSC [214] 97.19%.

Figure 5.14: Some examples from COIL100 database.

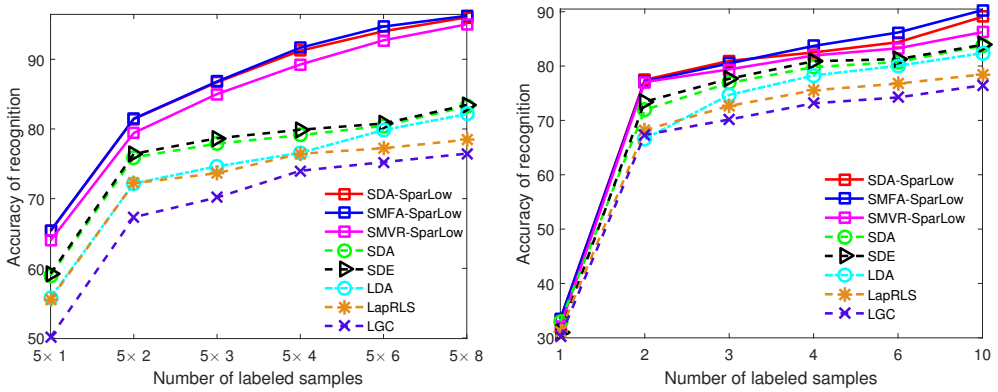
5.5.4 Evaluation of Semi-supervised *SparLow*

In this section, among the various manifold regularizations [200, 201], we confine us to construct the graph Laplacian matrix by $\mathbf{L} = \mathbf{W} - \mathbf{M}$ with $\mathbf{W}$ and $\mathbf{M}$ being same to Laplacian *SparLow* in Section 5.4.1. We perform the experiments on USPS digits and CMU PIE faces with the information as introduced before. We compare our proposed *SparLow* with corresponding similar approaches that applying in original data space, i.e., *SDA-SparLow* versus SDA [190], *SLap-SparLow* versus SDE [198], *SMVR-SparLow* versus label propagation methods, such as LapRLS [200] and LGC [199].

We normalize each image vector as the unit length. The dictionary $\widehat{\mathbf{D}}$ is initialized by Laplacian *SparLow*. For DR algorithms, SDA, SDE, and our *SDA-SparLow*, *SLap-SparLow*, 1NN classifier is performed in low dimensional space. For label propagation methods, LapRLS and LGC, we use the settings in [200, 199] for classification. We set $\mu_1 = 2.5 \times 10^{-4}, \mu_2 = 5 \times 10^{-3}$ for all the databases. $\lambda_1 = 0.02, \lambda_2 = 5 \times 10^{-4}$ and

$\lambda_1 = 0.2, \lambda_2 = 10^{-3}$ are set for PIE faces and USPS digits, respectively. We further set $\alpha = 0.05$ for *SDA-SparLow* and *SMVR-SparLow*, $\alpha_1 = 0.05, \alpha_2 = 0.01$ for *SMFA-SparLow*. The adjacency neighbors size for USPS, PIE are 20 and 12, respectively. For each class, we randomly choose $n_{\mathcal{L}}$ labeled samples and the rest for testing.

Fig. 5.15(a) gives the recognition results on PIE faces, in comparison of several state of the art methods. Similar to [190, 200, 201], for each individual with five poses, we randomly select $\eta_{\mathcal{L}}$ samples from each pose and totally $n^{\mathcal{L}}_{\text{train}} = 5 \times \eta_{\mathcal{L}}$ are chosen. It shows that our *SDA-SparLow* and *SLap-SparLow* outperform all other methods listed in Fig. 5.15(a), especially when the number of labeled sample is bigger. We also compare our methods on USPS digits, see Fig. 5.15(b). Similar to Fig. 5.15(a), it shows that our methods have more advantage when the number of labeled sample is increasing. Note that, for $n^{\mathcal{L}}_{\text{train}} = 1$ in Fig. 5.15(b), LDA can not be applied, cf. [186, 190].



(a) PIE faces (b) USPS digits

Figure 5.15: Recognition accuracy on unlabeled data.

5.5.5 Object Categorization

We now perform the experiments on objects or scenes that have different scales and complex backgrounds, such as the images from the datasets, namely, Caltech-101 [215], Caltech-256 [216], Pascal VOC 2007 [217], and 15-Scenes [29], see Fig. 5.16.

Building upon object categorization, we follow the pipeline depicted in Fig. 1.4. We use dense SIFT or dense DHOG (a fast SIFT implementation) [218] to detect the local image features. We shortly call it *SIFT/DHOG-SPP* representation. In this work, the local descriptor is extracted from $s \times s$ pixel patches densely sampled from each image. $s = 16$ for SIFT and $s = 16, 25, 31$ for DHOG. The dimension of each SIFT/DHOG descriptor is 128. A codebook with the size of $k = 1024, 2048$, or 4096 is learned for coding SIFT/DHOG descriptors, cf. [28]. We then divide the image into 4×4, 3×3 and 1×1 subregions, i.e., 21 bins. The spatial pooling procedure for each spatial sub-region is applied via the max pooling function associated with an "ℓ_2 normalization", cf. [27, 28, 109, 130]. The final

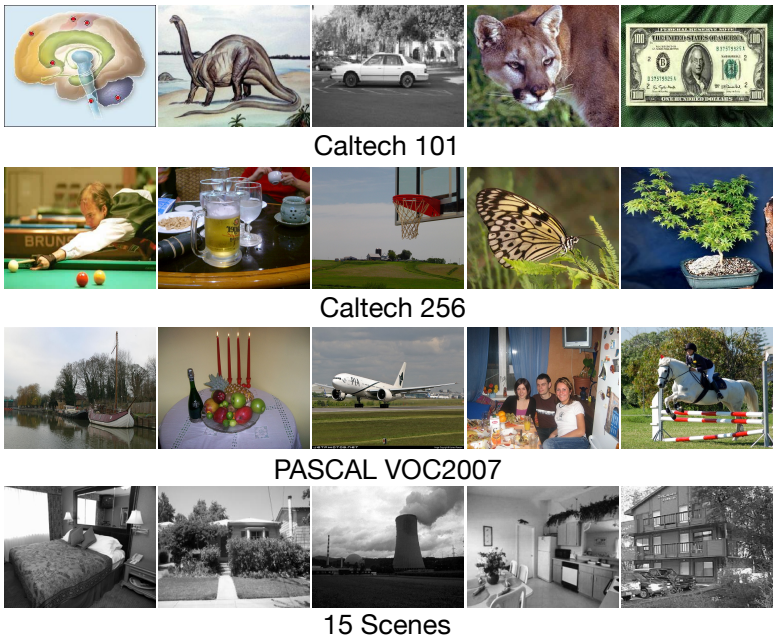


Figure 5.16: Examples from four datasets, i.e., Caltech-101, Caltech-256, PASCAL VOC2007 and Scene-15.

SPP representations are computed with the size $m = 21504$, 43008 or 96016, and hence are reduced into a low-dimensional PCA-projected subspace. The related codes are publicly online available, namely *VLFeat* software, [219]. In what follows, we denote m, m_{PCA}, r, l by dimension of SPP representations, PCA projected subspace, sparse codes, and our learned low dimensional representations, respectively.

Caltech-101 dataset

The Caltech-101 dataset [215] contains 9144 images from 102 classes (i.e., 101 object classes plus a background category), see Fig. 5.16. Most images are in medium resolution, i.e., about 300×300 pixels. The number of images per category varies from 31 to 800. This dataset is particularly challenging for learning-based systems, because the number of training samples in some categories is exceedingly small.

We set $\lambda_1 = 5 \times 10^{-2}, \lambda_2 = 10^{-5}$, $k = 1024$, and $k = 1020$. $l = 101, 287, 512$ are set for LDA, MFA, MVR related methods, respectively. We follow the common experimental setups in [27, 28, 29] and then randomly select 1, 5, 10, 15, 20 and 30 labeled images per category for training and the rest images for testing. For semi-supervised *SparLow*, the training set includes all labeled and unlabeled images. Table 5.4 gives the comparison of *LDA-SparLow*, *SLDA-SparLow* with approaches from the literature under different training samples. Note that, for $n^{\mathcal{L}}_{\text{train}} = 1$, *LDA-SparLow* and LDA+SVM can not be applied. It shows that our proposed approaches consistently outperform all the competing approaches. Especially,

the semi-supervised *SparLow* could significantly improve the recognition accuracy when the labeled training samples are limit. The possible reason is that some categories have large samples, e.g., the category *airplanes* has 800 samples, the $n_{\text{train}}^{\mathcal{L}} \leq 30$ is too much limit for training such a category. The semi-supervised *SparLow* could take advantage of all the data for training sparsifying dictionary, which is the key factor to promote the discrimination of sparse representations.

Fig. 5.17 plots the confusion matrix using *LDA-SparLow*. It shows that 12 categories in Caltech101 achieve the 100% classification accuracy, i.e., *accordion, snoopy, inline_skate, car_side, dollar_bill, garfield, metronome, okapi, pagoda, minaret, trilobite* and *stop_sign*. Several image examples from such 12 categories are shown in Fig. 5.18. On the other hand, the categories with highest confusion is *water lilly* (28.57%) and *lotus* (58.33%), i.e., 42.86% of *water lilly* are identified as *lotus* and 13.89% of *lotus* are identified as *water lilly*, see the bottom two lines of Fig. 5.18.

Table 5.4: Classification Performance (Average accuracy (%)) on Caltech-101.

Caltech-101	1	5	10	15	20	25	30
KSPM [29]	–	–	–	56.40	–	–	64.40
ScSPM+SVM [27]	–	–	–	67.0	–	–	73.2
LLC+SVM [28]	–	51.15	59.77	65.43	67.74	70.16	73.44
Griffin [216]	–	44.2	54.5	59.0	63.3	65.8	67.60
SRC [3, 109]	–	48.8	60.1	64.9	67.7	69.2	70.7
D-KSVD [108, 109]	–	49.6	59.5	65.1	68.6	71.1	73.0
LC-KSVD [109]	28.9	54.0	63.1	67.7	70.5	72.3	73.6
SSPIC [42]	–	55.1 ±0.2	62.1 ±0.2	65.0 ±0.2	67.7 ±0.2	68.9 ±0.2	71.5 ±0.2
LDA+GSVM	–	50.49	60.39	64.28	67.10	71.24	72.40
SparLDA	–	54.60	65.26	70.05	72.12	73.2	75.82
LDA-SparLow	31.23	56.44	67.12	73.82	74.70	76.20	76.86
SDA-SparLow	46.12	67.42	72.01	76.12	76.64	77.40	78.25
MFA-SparLow	32.43	57.52	68.44	73.95	75.56	76.63	77.32
SMFA-SparLow	46.02	68.66	72.92	76.02	77.24	77.80	78.42
MVR-SparLow	29.30	54.79	65.63	70.56	73.33	75.41	76.15
SMVR-SparLow	44.82	66.43	70.80	75.48	76.32	77.14	77.76

Similar to Fig. 5.5(a) and Fig. 5.5(b), we also test the reconstructive and discriminative performance of learned image representations for supervised *SparLow*. By running the *SparLow* with or without developed regularizations on the images from Caltech-101 database, Fig. 5.19(a) and Fig. 5.19(b) show the changing of reconstruction error and sparsity, and Fig. 5.19(c) and Fig. 5.19(d) compare the GSVM classification results. A similar conclusion is that the regularizers h_c and h_d, defined in Eq. (5.4) and Eq. (5.5), are imposed to ensure good reconstruction, and achieve stable discriminations.

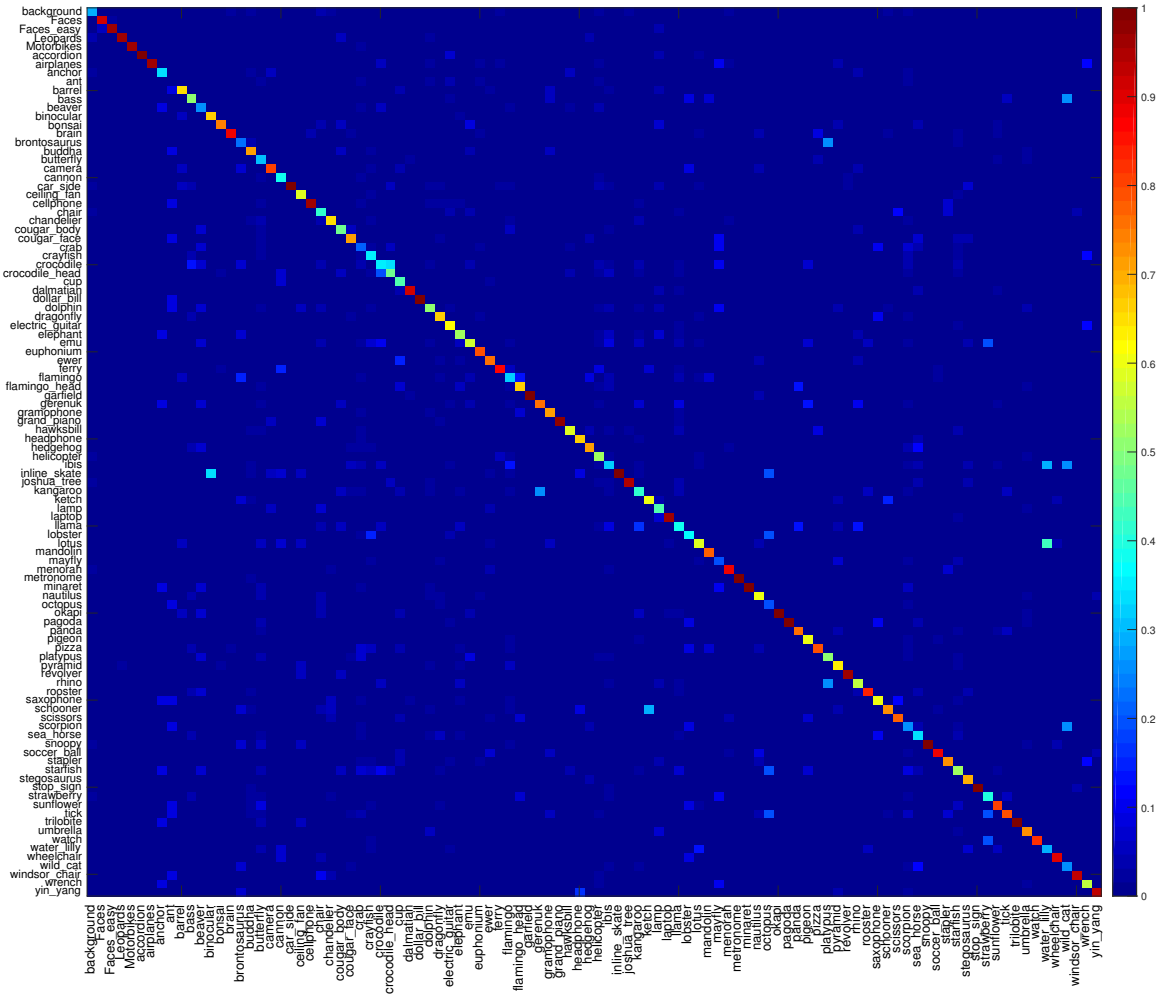


Figure 5.17: Confusion matrix for Caltech-101 with 30 training images per class, shown using the jet color scale from Matlab. Dark red indicates 100% while dark blue indicates 0%, with a gradient from warm to cool colors in between (see scale, right). A perfect matrix would be dark blue matrix except for a dark red diagonal.

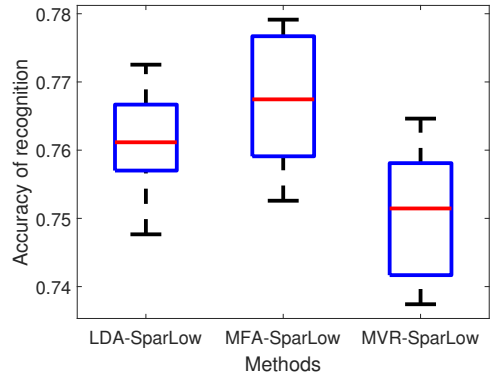
Caltech-256 and Pascal VOC 2007

The Caltech-256 dataset holds $30,607$ images falling into 256 categories with resolution from 113×150 to 960×1280. Each category has a minimum of 80 images. PASCAL VOC2007 consists of 20 categories with $5,011$ training and $4,952$ test images. It contains aero, bicycle, bird, boat, bottle, bus, car, cat, etc., with object instances occurring in a variety of scales, locations and viewpoints. The average size of VOC2007 is around 500×375 or 375×500. In contrast to Caltech-101, these two databases contain multiple objects in various poses at

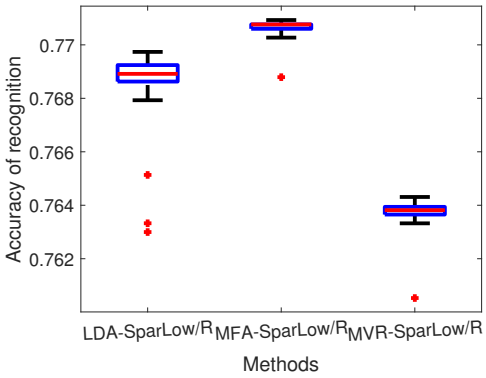
Figure 5.18: Performing *LDA-SparLow* on Caltech-101 with $n_{\text{train}} = 30$. This figure shows the examples from twelve categories with 100% accuracy and two categories with the highest confusion.

different locations within the image, with background clutter and occlusion, which results in higher intraclass diversity. Some examples are shown in Fig. 5.16.

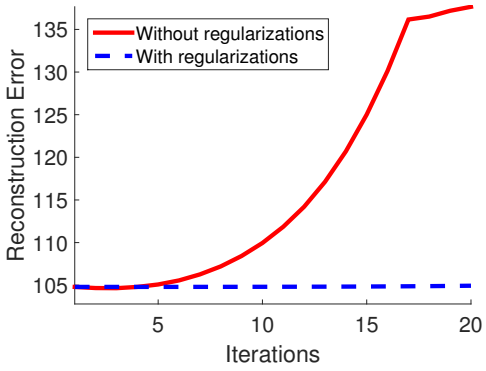
For Caltech-256, we carried our *DHOG-SPP SparLow* on randomly selected 15, 30, 45, 60 training images per class respectively. We set $m_{\text{PCA}} = 2560$, $k = 3072$, $l = 255$ for



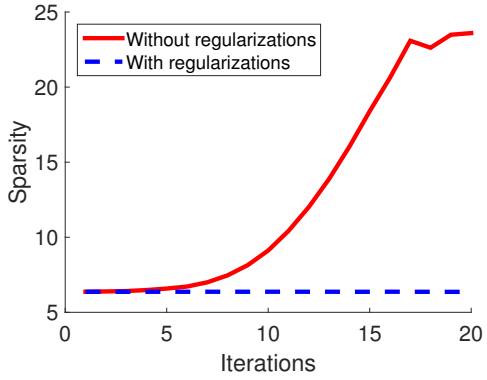
(a) Box plot of recognition accuracy without regularizers



(b) Box plot of recognition accuracy with regularizers



(c) Total Reconstruction Error



(d) Sparsity per image

Figure 5.19: Performing supervised *SparLow* with or without developed regularizers on Caltech-101. $n_{\text{train}} = 30$. Total Reconstruction Error is calculated by $\|\mathbf{X} - \mathbf{D}\boldsymbol{\Phi}\|_F^2$.

LDA-like *SparLow*. The codebook size for coding DHOG descriptors is set to be 128×4096. Finally, we use GSVM for classifying the low dimensional features. Table 5.5 shows our results outperform the state-of-the-art methods under all the cases. We also implemented TDDL for comparison, and the each sub-dictionary size of TDDL is fixed as $k = 200$. It shows that TDDL is far below our results. The possible reason is that TDDL associated with a binary classifier may suffer the huge number of classes (i.e., 256).

For Pascal VOC 2007, our recognition accuracy 64.12% and 66.68% achieved by *DHOG-SPP* for *LDA-SparLow* with $l = 19$ and *MFA-SparLow* with $l = 103$. These results are much higher than state-of-the-art results, such as LLC (acc. 57.66%) in [28], LDP (acc. 53.70%) in [220], and VQ (acc. 56.07%) in [29].

Table 5.5: Classification Performance (Average accuracy (%)) on Caltech-256 datasets.

Caltech-256	15	30	45	60
KSPM [29]	–	34.10	–	–
ScSPM+SVM [27]	27.73	34.02	37.46	40.14
LLC+SVM [28]	34.36	41.19	45.31	47.68
Griffin [216]	28.30	34.10	–	–
SRC [3, 109]	27.86	33.33	–	–
D-KSVD [108, 109]	–	27.79	–	32.67
LC-KSVD [109]	28.9	34.32	–	–
LSc [37]	29.99	35.74	38.47	40.32
TDDL [26]	30.20	36.44	38.89	46.42
SparLDA	33.09	36.65	43.74	48.02
LDA-SparLow	35.44	39.48	46.10	51.13
SparMFA ($l = 287$)	33.32	36.90	44.63	48.62
MFA-SparLow	36.02	40.21	48.02	52.66
SparMVR	31.01	34.92	42.44	47.01
MVR-SparLow	34.11	37.58	45.30	49.29

15-Scenes dataset

We finally evaluated our *SparLow* on the 15-Scenes dataset [29]. This dataset contains totally 4485 images falling into 15 categories, with the number of images in each category ranging from 200 to 400 and image size around 300×250 pixels. The image content is diverse, containing not only indoor scenes, such as bedroom, kitchen, but also outdoor scenes, such as Building and country, etc., see Fig. 5.16.

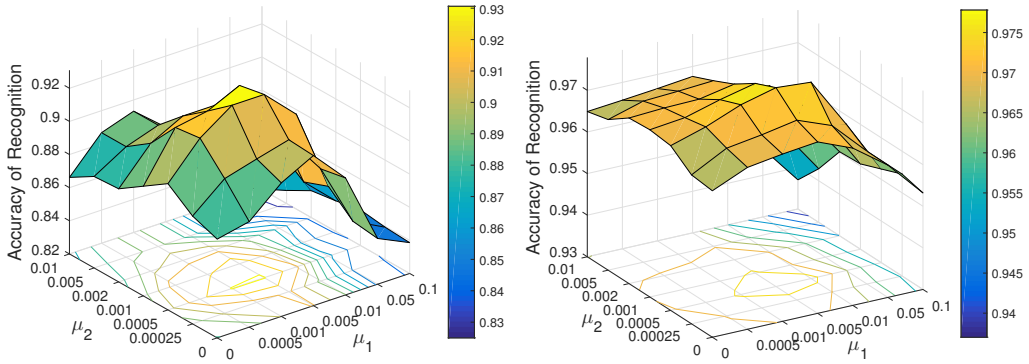
Table 5.6: Averaged classification Rate (%) comparison on 15-Scenes dataset. The classifier is 1NN for the third column if not specified.

Methods	Accuracy	Methods	Accuracy
LDP [220]	81.40	LDA	91.69
KSPM [29]	83.50	SparLDA	95.89
ScSPM+GSVM [27]	80.28	*LDA-SparLow*	97.47
LLC+GSVM [28]	89.2	MFA	92.82
SRC [3, 109]	91.8	SparMFA	96.65
LSc [37]	89.7	*MFA-SparLow*	98.46
K-SVD[4] + LDA	92.6	MVR ($l = 512$)	93.10
D-KSVD [108]	89.01	SparMVR	96.32
LC-KSVD [109]	92.9	*MVR-SparLow*	97.55
SDA [190]	97.28	*SDA-SparLow*	99.18
SDE [198]	97.66	*SLap-SparLow*	99.25
LapRLS [200]	94.86	*SMVR-SparLow*	99.12

Following the common experimental settings, we use *SIFT-SPP* as input with $k = 1024$ and $m_{\text{PCA}} = 2000$. For MFA, we set $k_1 = 70, k_2 = 100$, and $l = 50$. For SparMFA and *MFA-SparLow*, we set $k_1 = 30, k_2 = 100$ and $l = 60$. For all supervised and semi-supervised *SparLow* methods, the dictionary size $k = 750$. Table 5.6 compares our results with several sparse coding methods in [27, 3, 28, 37, 37, 108, 109] and others in [220, 29], which are all using SPP features as input data. As shown in Table 5.6, our approaches significantly outperform all state-of-the-art approaches. Note that, the bottom three lines are all semi-supervised methods, $n_{\text{test}} = n^{\mathcal{U}}_{\text{train}}$.

5.5.6 Parameters Sensitivity

In this section, we investigate the sensitivity of the performance while varying parameters, such as μ_1, μ_2 in Eq. (5.6), the dimension l of low-dimensional representations, and the dimension m of input features.

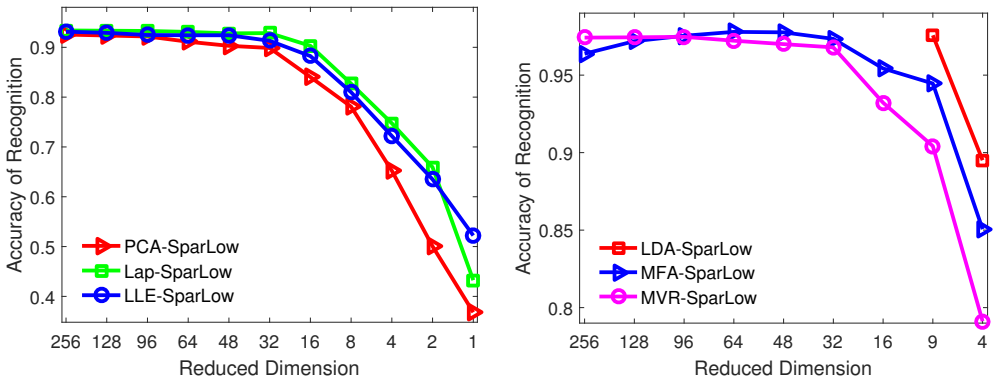


(a) *PCA-SparLow* (b) *LDA-SparLow*

Figure 5.20: *Sensitivity in recognition rate on USPS digits with respect to weighing factors μ_1 and μ_2.*

At first, we present our preliminary results on sensitivity of the *PCA-SparLow* and *LDA-SparLow* on the 1NN classification problem with respect to the weighing factors μ_1 and μ_2, cf. Fig. 5.20. The experiments are performed in USPS dataset with $k = 1000$, and $l = 50$ for *PCA-SparLow*, $l = 9$ for *LDA-SparLow*. Note that, the recognition accuracy is the average (converged) value after the algorithm running for 20 loops. It is easy to see that a suitable choice of μ_1 and μ_2 could improve the convergence of the *SparLow* system. Similarity, we next evaluate the sensitivity of the both unsupervised and supervised *SparLow* methods on the 1NN classification problem with different low dimensions l, cf. Fig. 5.21(a). The experimental settings are same to Fig. 5.20, also performed on USPS dataset. The low dimension $l > 32$ is a better choice for all unsupervised *SparLow*. For supervised *SparLow*, the best l for *MFA-SparLow* is around 64, for LDA is 9, for *MVR-SparLow* is $l > 32$.

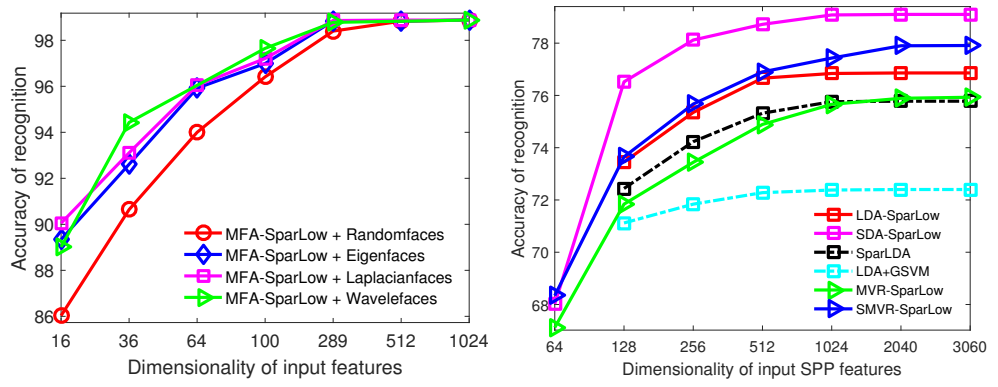
Considering one fact that the computational complexity of sparse coding depends on the choice of dictionary size, m and k, cf. [221]. One popular way is first to lead DR transformation on raw image and then learn or construct a dictionary on reduced space, cf. [3]. Following



(a) Unsupervised *SparLow* (b) Supervised *SparLow*

Figure 5.21: *Sensitivity in recognition rate with respect to Low dimension l.*

this way, in Fig. 5.22(a), we test the proposed *MFA-SparLow* in reduced feature space of PIE faces. We first reduce the dimensionality into $\mathbf{R}^d$ with $d = 512, 289, 100, 64, 36, 16$ by random Gaussian matrix (*Randomfaces*), *Eigenfaces* [186], *Laplacianfaces* [183], and wavelet transforms (*Waveletfaces*). We then learn the dictionary $\mathbf{D} \in \mathbb{R}^{d \times r}$ and $\mathbf{P} \in \mathbb{R}^{k \times k}$. Fig. 5.22(a) demonstrates that the recognition results of *MFA-SparLow* with increasing dimensionality of different reduced input features. It is clear from this result that, when the resolution is not very low ($m > 17 \times 17$), the recognition rates perform not significant recession. Fig. 5.22(b) plots the recognition results of supervised *SparLow* and semi-supervised *SparLow* on input PCA-projected input features. It shows that $l > 512$ is the better choice for such methods.



(a) (b)

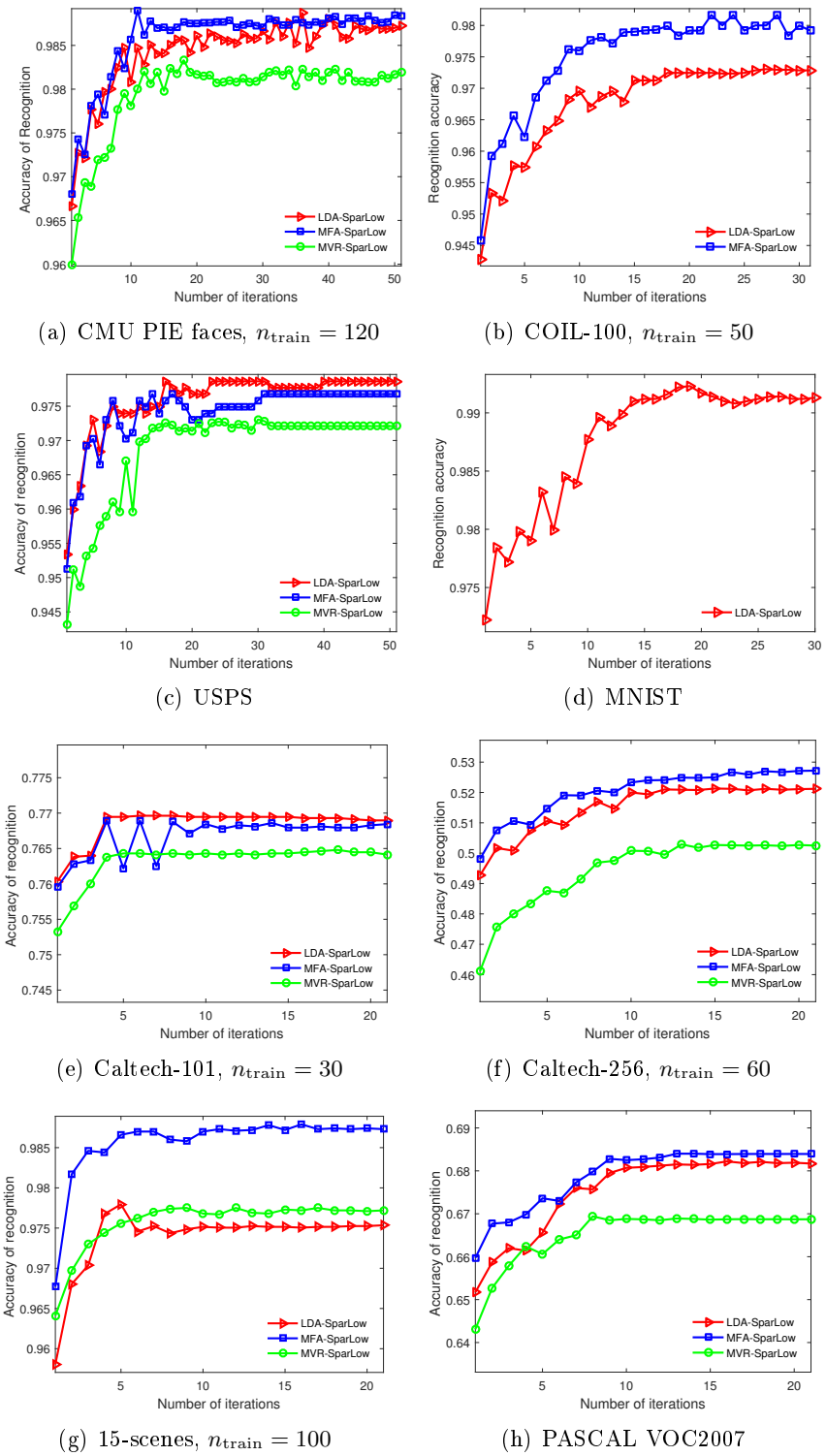
Figure 5.22: (a) Recognition results using proposed *MFA-SparLow* in feature space on PIE faces. $n_{\text{train}} = 120$, $k = 2040$. Note that, the dimensionality of Waveletfaces is only $16, 36, 100, 289, 1024$. (b) Recognition results using proposed *MFA-SparLow* in PCA projected subspace on Caltech-101 dataset. $n_{\text{train}} = 30$, $k = 3060$.

5.5.7 Optimization Process

With the aim of better understanding our proposed *SparLow*, in this section, we show the optimization process of supervised *SparLow* performed on CMU PIE faces, COIL100 cartoons, Handwritten digits (MNIST and USPS), and objects/scenes datasets (i.e., Caltech-101, Caltech-256, Pascal VOC2007 and Scene-15). For MNIST, USPS, and VOC2007, we use the standard splitting for training and testing subsets. For CMU PIE, COIL100, Caltech-101, Caltech-256 and Scene-15, n_{train} is shown in Fig. 5.23. For Caltech-101 and Caltech-256, GSVM is the classifier. For other datasets, we choose 1NN to classify a test sample. All parameter settings are described in previous Sections. All sub-figures in Fig. 5.23 show that the supervised *SparLow* have a good convergence after 10 or 15 iterations. Note that, the starting points demonstrate the recognition results by directly applying the LDA, MFA and MVR on sparse representations with respect to $\widehat{\mathbf{D}}$.

5.6 Summary

In this chapter, we present a low dimensional representation learning approach, coined here as *SparLow*, which leverages both the sparse representation and the trace quotient criterion. It can be considered as a two-step disentangling mechanism, which applies the trace quotient criterion on the sparse representations. Our proposed generic cost function is defined on a sparsifying dictionary and an orthogonal transformation, which form a product Riemannian manifold. A geometric CG algorithm is developed for optimizing the cost function. Our experimental results depict that in comparison with the state of the art unsupervised, supervised and semi-supervised representation learnings methods, our proposed *SparLow* method possesses promising performance in data visualization and classification.



(a) CMU PIE faces, $n_{\text{train}} = 120$ (b) COIL-100, $n_{\text{train}} = 50$

(c) USPS (d) MNIST

(e) Caltech-101, $n_{\text{train}} = 30$ (f) Caltech-256, $n_{\text{train}} = 60$

(g) 15-scenes, $n_{\text{train}} = 100$ (h) PASCAL VOC2007

Figure 5.23: This picture depicts the optimization process of supervised *SparLow* on different image datasets.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The focus of this dissertation is on the investigation of disentangling underlying explanatory information from sparse representations of the image data, that facilitates the learning task of interest. Technically, by exploring the differentiability of solutions of some sparse coding methods, we proposed a two-layer representation learning framework, which adopts sparse representations in a further learning mechanism to disentangle the useful information. In this dissertation, such a two-layer learning block has been successfully applied for solving two computer vision problems as follows, i) modeling the evolution of dynamic course of dynamic textures, and ii) constructing effective low dimensional image representations. In general, the whole learning process was treated as an optimization problem on jointly learning a sparsifying dictionary and a further problem-dependent disentanglement parameter, based on geometric gradient methods on suitable matrix manifolds.

Modeling dynamic scenes has been widely studied due to its importance in various video processing applications, e.g., video classification, decomposition and segmentation. This problem is extremely challenging because the shape and appearance of a dynamic texture are nonrigid and continuously change as an unknown function over time and space. Thus, finding appropriate "states" (or representation) is the key to explore the evolution of dynamic textured scenes. Moreover, the traditional system, i.e., LDS, is often vulnerable to non-Gaussian noise, such as missing data or occlusion of the dynamic scenes. To tackle these challenges, our proposed two-layer learning block is adopted to model the DT by appealing to the principle of sparsity. In Chapter 3, instead of using Principle Components (PCs) as the "states" in LDS, sparse coefficients over a learned dictionary were imposed as the underlying "states". In this way, the dynamical process of DTs exhibits a transition course of corresponding sparse events. We developed a combined regression associated with several regularizations for a joint process — "state extraction" and "state transition". Then we treated the solution of the above combined regression as an adaptive dictionary learning problem, called *SLDS*. To enable the *SLDS* in DT's classification, two discriminative *SLDS* algorithms associated with a sparse transition matrix were proposed. Compared with state-of-the-art video processing methods on several benchmark data sequences, the proposed method showed a robust performance on DT sequence synthesis, recognition and denoising.

On the other hand, natural images often have high dimensions and complex statistical structure with unknown distribution, and hence they are difficult to be explicitly parame-

terized. Therefore, directly modeling the raw images for tasks may challenge most machine learning techniques, e.g., the high-computational load. A suitable choice of low-dimensional data representation has been found to be a powerful preprocessing instrument to support effective machine learning tasks, such as classification and visualization. Based on such arguments, we employed the proposed two-layer learning block to find low dimensional representations of images in triple unsupervised, supervised and semi-supervised manners. Under this learning framework, a novel algorithm, called *SparLow*, was introduced in Chapter 4. By applying a trace quotient criterion on sparse coefficients, we developed a generic cost function for learning jointly a sparsifying dictionary and a dimensionality reduction transformation. It led to a wide range of counterparts of classic low dimensional representation methods, which include Principal Component Analysis, Local Linear Embedding, Laplacian Eigenmap, Linear Discriminant Analysis (LDA), Semi-supervised LDA, etc. Our proposed generic cost function was defined on a sparsifying dictionary and an orthogonal transformation, which form a product Riemannian manifold. A geometric CG algorithm was developed for optimizing the cost function. The proposed approach had been adapted to a wide variety of machine learning tasks, such as 3D data visualization, face/digit/cartoon recognition, object categorization and clustering. Our numerical experiments were compared with state-of-the-art data representation methods on several benchmark image datasets, to demonstrate the effectiveness of the proposed algorithm.

We have demonstrated that our proposed *SparLow* could be successfully applied on the raw images, i.e., pixel intensity values. However, in the practical numerical applications, one issue often challenges the most algorithms for images processing. This issue often occurs when objects or scenes having different scales or complex backgrounds, in which it is impossible to directly learn a uniform dictionary for all images. One popular way to cope with this issue is to first detect various local image features, such as SIFT, HOG or CNN-based features, and then encode these local features to generate a single fixed-length high-level vector that describes the entire image. We extended our proposed *SparLow* to such high-level features, and the recognition results showed its competitive performance on comparison with the state-of-the-art object categorization approaches.

6.2 Future Work

This dissertation focuses on the development of a two-layer representation learning block, as shown in Fig. 3.1. We also developed methods for its two applications, i.e., modeling dynamic textures and learning low dimensional image representations, called *SLDS* and *SparLow*, respectively. We believe that with a thorough understanding of the proposed framework, one can be well guided in integrating it into further sophisticated applications. On the other hand, the proposed framework also has a number of limitations.

According to these arguments, in the following, we introduce several potential future directions of the proposed two-layer building block.

i) High-computation cost is one significant drawback of the proposed two-layer representation learning block. For example, the training time will be prohibitive while learning low

dimensional representations of large scale image dataset, e.g., millions of training images/-patches, or modeling the dynamic course of a very long image sequence. Optimizing the optimization algorithms to speed up the learning convergence is one potential direction.

ii) The second possible future extension is to learn a dictionary for DT sequences or images with high resolution (or high dimension), since finding an appropriate faction of data as shown in Eq. (1.1) can be prohibitively expensive when the dimensionality of the raw input is huge. Extending our proposed two-layer learning block to tensor field is an interesting direction. For example, learning separable dictionaries in small sizes to explore a transition of 2D sparse events is worth a try.

iii) The proposed *SLDS* can handle the video with single dynamic textures, e.g., fire, moving water, smoke, swaying trees or moving clouds, but it fails to handle the video sequence containing complex moving scenes, such as the DTs under dynamic background or one video having many dynamic textures. Representing dynamic scenes as a mixture of *SLDS*s may cope with such a limitation.

iv) Specifically for DT modeling, another potential research direction would be to construct a hierarchical learning scheme, e.g., bag-of-systems (BoS) representation [222, 223, 224], on a collection of *SLDS* parameters to promote the task of interest, such as motion classification, detection and segmentation. The *SLDS* parameters mentioned here are dictionary matrices and sparse transition matrices.

v) Learning an arbitrary linear transformation between two random sparse vectors admitting the sparse distribution is also a pending question in the community, cf. [172]. For example, in order to build a solvable and stable learning system for linearly transiting consecutive sparse states, as introduced in Chapter 3, it needs to find a suitable searching set for such a transition parameter.

vi) The proposed *SparLow* is flexible and can be extended to more general cases of low dimensional representation learning models with orthogonal constraints.

vii) Integrating the proposed two-layer representation learning block into deep learning architectures is another potential research direction. For example, directly constructing a multi-layer *SparLow* may improve the performance of some specific image processing tasks. For another example, it can treat the proposed *SparLow* as one layer or a neuron in deep CNN framework.

Bibliography

1. Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

2. Renaud Péteri, Sándor Fazekas, and Mark J Huiskes. Dyntex: A comprehensive database of dynamic textures. *Pattern Recognition Letters*, 31(12):1627–1632, 2010.

3. John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2009.

4. Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

5. Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

6. Yoshua Bengio Ian Goodfellow and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.

7. Yen Yu Lin, Tyng Luh Liu, and Chiou Shann Fuh. Multiple kernel learning for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1147–1160, 2011.

8. Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5), 2007.

9. Yoshua Bengio and Aaron Courville. *Deep learning of representations*, pages 1–28. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

10. Yoshua Bengio et al. Deep learning of representations for unsupervised and transfer learning. *ICML Unsupervised and Transfer Learning*, 27:17–36, 2012.

11. Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001.

12. Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

13. Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

14. John P Cunningham and Zoubin Ghahramani. Linear dimensionality reduction: Survey, insights, and generalizations. *Journal of Machine Learning Research*, 2015.

15. Effrosini Kokiopoulou, Jie Chen, and Yousef Saad. Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, 18(3):565–602, 2011.

16. Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

17. Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*, volume 46. John Wiley & Sons, 2004.

18. Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.

19. Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

20. Stéphane Mallat. *A wavelet tour of signal processing*. Academic press, 1999.

21. Amir Said and William A Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, 1996.

22. David L Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

23. Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

24. Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

25. Simon Hawe, Matthias Seibert, and Martin Kleinsteuber. Separable dictionary learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 438–445. IEEE, June 2013.

26. Julien Mairal, Francis Bach, and Jean Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012.

27. Jianchao Yang, Kai Yu, Yihong Gong, and Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1794–1801. IEEE, 2009.

28. Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3360–3367. IEEE, 2010.

29. Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2169–2178. IEEE, 2006.

30. Christian Thurau and Václav Hlavác. Pose primitive based human action recognition in videos or still images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

31. Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

32. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

33. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

34. Jianchao Yang, Zhaowen Wang, Zhe Lin, Scott Cohen, and Thomas Huang. Coupled dictionary training for image super-resolution. *IEEE Transactions on Image Processing*, 21(8):3467–3478, 2012.

35. Kai Yu, Tong Zhang, and Yihong Gong. Nonlinear learning using local coordinate coding. In *Advances in neural information processing systems*, pages 2223–2231, 2009.

36. Miao Zheng, Jiajun Bu, Chun Chen, Can Wang, Lijun Zhang, Guang Qiu, and Deng Cai. Graph regularized sparse coding for image representation. *IEEE Transactions on Image Processing*, 20(5):1327–1336, 2011.

37. Shenghua Gao, Ivor Wai-Hung Tsang, and Liang-Tien Chia. Laplacian sparse coding, hypergraph laplacian sparse coding, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):92–104, 2013.

38. Bo Chen, Gungor Polatkan, Guillermo Sapiro, David Blei, David Dunson, and Lawrence Carin. Deep learning with hierarchical convolutional factor analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1887–1901, 2013.

39. Andrew Ng. Sparse autoencoder. *CS294A Lecture notes*, 72:1–19, 2011.

40. Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Discriminative learned dictionaries for local image analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

41. Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang. Sparse representation based fisher discrimination dictionary learning for image classification. *International Journal of Computer Vision*, 109(3):209–232, 2014.

42. Umamahesh Srinivas, Yuanming Suo, Minh Dao, Vishal Monga, and Trac D Tran. Structured sparse priors for image classification. *IEEE Transactions on Image Processing*, 24(6):1763–1776, 2015.

43. Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

44. Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

45. Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

46. Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.

47. Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61, 1998.

48. Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

49. Carlos Ramirez, Vladik Kreinovich, and Miguel Argaez. Why ℓ_1 is a good approximation to ℓ_0: A geometric explanation. *Journal of Uncertain Systems*, 7(3):203–207, 2013.

50. Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

51. Yurii Nesterov, Arkadii Nemirovskii, and Yinyu Ye. *Interior-point polynomial algorithms in convex programming*, volume 13. SIAM, 1994.

52. Stephen J Wright. *Primal-Dual Interior-Point Methods*, volume 54. SIAM, 1997.

53. Kwangmoo Koh, Seung-Jean Kim, and Stephen P Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Journal of Machine learning research*, 8(8):1519–1555, 2007.

54. Tony F Chan, Gene H Golub, and Pep Mulet. A nonlinear primal-dual method for total variation-based image restoration. *SIAM journal on scientific computing*, 20(6):1964–1977, 1999.

55. Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear algebra and its applications*, 284(1):193–228, 1998.

56. Joshua Trzasko and Armando Manduca. Highly undersampled magnetic resonance image reconstruction via homotopic-minimization. *IEEE Transactions on Medical imaging*, 28(1):106–121, 2009.

57. David L Donoho and Yaakov Tsaig. Fast solution of-norm minimization problems when the solution may be sparse. *IEEE Transactions on Information Theory*, 54(11):4789–4812, 2008.

58. Elaine T Hale, Wotao Yin, and Yin Zhang. A fixed-point continuation method for l1-regularized minimization with applications to compressed sensing. *CAAM TR07-07, Rice University*, 43:44, 2007.

59. Zaiwen Wen, Wotao Yin, Donald Goldfarb, and Yin Zhang. A fast algorithm for sparse reconstruction based on shrinkage, subspace optimization, and continuation. *SIAM Journal on Scientific Computing*, 32(4):1832–1857, 2010.

60. Zaiwen Wen, Wotao Yin, Hongchao Zhang, and Donald Goldfarb. On the convergence of an activeset method for ℓ_1 minimization. *Optimization Methods and Software*, 27(6):1127–1146, 2012.

61. Mário AT Figueiredo, Robert D Nowak, and Stephen J Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, 2007.

62. Ewout Van Den Berg and Michael P Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

63. Ewout Van den Berg and Michael P Friedlander. Sparse optimization with least-squares constraints. *SIAM Journal on Optimization*, 21(4):1201–1229, 2011.

64. Sangwoon Yun and Kim-Chuan Toh. A coordinate gradient descent method for ℓ_1-regularized convex minimization. *Computational Optimization and Applications*, 48(2):273–307, 2011.

65. Manya V Afonso, José M Bioucas-Dias, and Mário AT Figueiredo. Fast image recovery using variable splitting and constrained optimization. *IEEE Transactions on Image Processing*, 19(9):2345–2356, 2010.

66. Mário AT Figueiredo and Robert D Nowak. An EM algorithm for wavelet-based image restoration. *IEEE Transactions on Image Processing*, 12(8):906–916, 2003.

67. Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

68. Mário AT Figueiredo and Robert D Nowak. A bound optimization approach to wavelet-based image deconvolution. In *IEEE International Conference on Image Processing, 2005.*, volume 2, pages II–782. IEEE, 2005.

69. Michael Elad. Why simple shrinkage is still relevant for redundant representations? *IEEE Transactions on Information Theory*, 52(12):5559–5569, 2006.

70. Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

71. José M Bioucas-Dias and Mário AT Figueiredo. A new twist: two-step iterative shrinkage/thresholding algorithms for image restoration. *IEEE Transactions on Image Processing*, 16(12):2992–3004, 2007.

72. Thomas Blumensath and Mike E Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5-6):629–654, 2008.

73. Joel A Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, 50(10):2231–2242, 2004.

74. Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

75. Joel A Tropp and Anna C Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

76. Deanna Needell and Roman Vershynin. Greedy signal recovery and uncertainty principles. In *Electronic Imaging 2008*, pages 68140J–68140J. International Society for Optics and Photonics, 2008.

77. Deanna Needell and Joel A Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

78. David L Donoho, Yaakov Tsaig, Iddo Drori, and Jean-Luc Starck. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 58(2):1094–1121, 2012.

79. Tong Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. In *Advances in Neural Information Processing Systems*, pages 1921–1928, 2009.

80. Nick Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Applied and computational harmonic analysis*, 10(3):234–253, 2001.

81. Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

82. Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

83. Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265–274, 2009.

84. G Hosein Mohimani, Massoud Babaie-Zadeh, and Christian Jutten. Fast sparse representation based on smoothed ℓ_0 norm. In *Independent Component Analysis and Signal Separation*, pages 389–396. Springer, 2007.

85. Hosein Mohimani, Massoud Babaie-Zadeh, Irina Gorodnitsky, and Christian Jutten. Sparse recovery using smoothed ℓ_0 (SL0): Convergence analysis. *arXiv preprint arXiv:1001.5073*, 2010.

86. G Hosein Mohimani, Massoud Babaie-Zadeh, and Christian Jutten. Complex-valued sparse representation based on smoothed ℓ_0 norm. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3881–3884. IEEE, 2008.

87. Bhaskar D Rao and Kenneth Kreutz-Delgado. An affine scaling methodology for best basis selection. *IEEE Transactions on Signal Processing*, 47(1):187–200, 1999.

88. Bhaskar D Rao, Kjersti Engan, Shane F Cotter, Jason Palmer, and Kenneth Kreutz-Delgado. Subset selection in noise based on diversity measure minimization. *IEEE Transactions on Signal Processing*, 51(3):760–770, 2003.

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

90. BD Rao and Kenneth Kreutz-Delgado. Basis selection in the presence of noise. In *Conference Record of the Thirty-Second Asilomar Conference on Signals, Systems & Computers.*, volume 1, pages 752–756. IEEE, 1998.

91. Rick Chartrand. Fast algorithms for nonconvex compressive sensing: Mri reconstruction from very few data. In *IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2009. ISBI'09.*, pages 262–265. IEEE, 2009.

92. Rick Chartrand and Wotao Yin. Iteratively reweighted algorithms for compressive sensing. In *IEEE international conference on Acoustics, speech and signal processing, (ICASSP).*, pages 3869–3872. IEEE, 2008.

93. Rayan Saab and Özgür Yılmaz. Sparse recovery by non-convex optimization–instance optimality. *Applied and Computational Harmonic Analysis*, 29(1):30–48, 2010.

94. Rayan Saab, Rick Chartrand, and Özgür Yilmaz. Stable sparse approximations via nonconvex optimization. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3885–3888. IEEE, 2008.

95. Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

96. Kjersti Engan, Sven Ole Aase, and J Hakon Husoy. Method of optimal directions for frame design. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 5, pages 2443–2446. IEEE, 1999.

97. Michael S Lewicki and Terrence J Sejnowski. Learning overcomplete representations. *Neural computation*, 12(2):337–365, 2000.

98. Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Te-Won Lee, and Terrence J Sejnowski. Dictionary learning algorithms for sparse representation. *Neural computation*, 15(2):349–396, 2003.

99. Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

100. Kjersti Engan, Karl Skretting, and John Håkon Husøy. Family of iterative ls-based dictionary learning algorithms, ils-dla, for sparse signal representation. *Digital Signal Processing*, 17(1):32–49, 2007.

101. Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

102. P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

103. Karl Skretting and Kjersti Engan. Recursive least squares dictionary learning algorithm. *IEEE Transactions on Signal Processing*, 58(4):2121–2130, 2010.

104. Stuart P Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

105. Kjersti Engan, Sven Ole Aase, and John Håkon Husøy. Multi-frame compression: Theory and design. *Signal Processing*, 80(10):2121–2140, 2000.

106. Sven Ole Aase, JH Husoy, Karl Skretting, and Kjersti Engan. Optimized signal expansions for sparse representation. *IEEE Transactions on Signal Processing*, 49(5):1087–1096, 2001.

107. Karl Skretting. Sparse signal representation using overlapping frames. 2002.

108. Qiang Zhang and Baoxin Li. Discriminative k-svd for dictionary learning in face recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2691–2698. IEEE, 2010.

109. Zhuolin Jiang, Zhe Lin, and Larry S Davis. Label consistent K-SVD: learning a discriminative dictionary for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2651–2664, 2013.

110. Olivier Bousquet and Léon Bottou. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

111. Michal Aharon and Michael Elad. Sparse and redundant modeling of image content using an image-signature-dictionary. *SIAM Journal on Imaging Sciences*, 1(3):228–247, 2008.

112. Koray Kavukcuoglu and Yann Lecun. Fast inference in sparse coding algorithms with applications to object recognition. In *Technical report, Computational and Biological Learning Lab, Courant Institute, NYU.* Citeseer.

113. Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.

114. Gianfranco Doretto, Alessandro Chiuso, Ying Nian Wu, and Stefano Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.

115. Antoni B Chan and Nuno Vasconcelos. Layered dynamic textures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1862–1879, 2009.

116. Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

117. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

118. Fu Jie Huang, Y-Lan Boureau, Yann LeCun, et al. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

119. Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

120. Michael Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing.* Springer, 2010.

121. John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

122. Xue Mei and Haibin Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2259–2272, 2011.

123. R. Raina, A. Battle, H. Lee, B. Packer, and A.Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM, 2007.

124. Koray Kavukcuoglu, Rob Fergus, Yann LeCun, et al. Learning invariant features through topographic filter maps. In *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*, pages 1605–1612. IEEE, 2009.

125. Junbin Gao, Qinfeng Shi, and Tibério S Caetano. Dimensionality reduction via compressive sensing. *Pattern Recognition Letters*, 33(9):1163–1170, 2012.

126. Julien Mairal, Marius Leordeanu, Francis Bach, Martial Hebert, and Jean Ponce. Discriminative sparse image models for class-specific edge detection and image interpretation. In *European Conference on Computer Vision (ECCV)*, pages 43–56. Springer, 2008.

127. Florent Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1243–1256, 2008.

128. Qiang Qiu, Vishal M Patel, and Rama Chellappa. Information-theoretic dictionary learning for image classification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(11):2173–2184, 2014.

129. Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis R Bach. Supervised dictionary learning. In *Advances in neural information processing systems*, pages 1033–1040, 2009.

130. Jianchao Yang, Kai Yu, and Thomas Huang. Supervised translation-invariant sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3517–3524. IEEE, 2010.

131. Ioannis A Gkioulekas and Todd Zickler. Dimensionality reduction using the sparse linear model. In *Advances in Neural Information Processing Systems*, pages 271–279, 2011.

132. Yu-Xiang Wang and Huan Xu. Noisy sparse subspace clustering. *Journal of Machine Learning Research*, 17(12):1–41, 2016.

133. Francis Bach, Rodolphe Jenatton, Julien Mairal, and Guillaume Obozinski. Structured sparsity through convex optimization. *Statistical Science*, pages 450–468, 2012.

134. Mark Schmidt, Glenn Fung, and Rómer Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *Proceedings of the 18th European conference on Machine Learning*, pages 286–297. Springer-Verlag, 2007.

135. Andreas Argyriou. A study of convex regularizers for sparse recovery and feature selection. Technical report, Center for Visual Computing, Ecole Centrale Paris, 2010.

136. H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67(2):301–320, 2005.

137. Chunhui Chen and Olvi L Mangasarian. A class of smoothing functions for nonlinear and mixed complementarity problems. *Computational Optimization and Applications*, 5(2):97–138, 1996.

138. Mandy Lange, Dietlind Zühlke, Olaf Holz, Thomas Villmann, and Saxonia-Germany Mittweida. Applications of ℓ_p-norms and their smooth approximations for gradient based learning vector quantization. In *European Symposium on Artificial Neural Networks (ESANN)*, 2014.

139. Honglak Lee, Chaitanya Ekanadham, and Andrew Y Ng. Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880, 2008.

140. Geoffrey Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

141. JA Bagnell and David M Bradley. Differentiable sparse coding. In *Advances in Neural Information Processing Systems*, pages 113–120, 2009.

142. Daniel Gabay. Minimizing a differentiable function over a differential manifold. *Journal of Optimization Theory and Applications*, 37(2):177–219, 1982.

143. Steven T Smith. Optimization techniques on riemannian manifolds. *Fields institute communications*, 3(3):113–135, 1994.

144. Simon Hawe, Martin Kleinsteuber, and Klaus Diepold. Analysis operator learning and its application to image reconstruction. *IEEE Transactions on Image Processing*, 22(6):2138–2150, 2013.

145. Simon Alois Hawe. *Learning Sparse Data Models via Geometric Optimization with Applications to Image Processing.* PhD thesis, Universität München, 2013.

146. Xian Wei, Hao Shen, and Martin Kleinsteuber. Trace quotient meets sparsity: A method for learning low dimensional image representations. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), pages 5268–5277, 2016.

147. Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

148. David G Luenberger, Yinyu Ye, et al. *Linear and nonlinear programming*, volume 2. Springer, 1984.

149. Roger W Brockett. Differential geometry and the design of gradient algorithms. In *Proc. Symp. Pure Math., AMS*, volume 54, pages 69–92, 1993.

150. Alan Edelman, Tomás A Arias, and Steven T Smith. The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

151. P-A Absil, Christopher G Baker, and Kyle A Gallivan. Trust-region methods on riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

152. Jean-Pierre Dedieu, Pierre Priouret, and Gregorio Malajovich. Newton's method on riemannian manifolds: covariant alpha theory. *IMA Journal of Numerical Analysis*, 23(3):395–419, 2003.

153. Y. Liu and C. Storey. Efficient generalized conjugate gradient algorithms, part 1: Theory,. *Journal of Optimization Theory and Applications*, 69(1):129–137, 1991.

154. U. Helmke and J. B. Moore. *Optimization and Dynamical Systems*. Springer, Berlin, 1994.

155. Hao Shen, Klaus Diepold, and Knut Hüper. A geometric revisit to the trace quotient problem. In *Proceedings of the 19th International Symposium of Mathematical Theory of Networks and Systems (MTNS 2010)*, pages 1–7, 2010.

156. Martin Kleinsteuber and Knut Huper. An intrinsic cg algorithm for computing dominant subspaces. In *Proceedings of the* 32^{th} *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 4, pages IV–1405. IEEE, 2007.

157. Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1):185–203, 1981.

158. Gianfranco Doretto, Daniel Cremers, Paolo Favaro, and Stefano Soatto. Dynamic texture segmentation. In *Ninth IEEE International Conference on Computer Vision (ICCV)*, pages 1236–1242. IEEE, 2003.

159. Guoying Zhao and Matti Pietikainen. Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):915–928, 2007.

160. Rizwan Chaudhry, Gregory Hager, and René Vidal. Dynamic template tracking and recognition. *International journal of computer vision*, 105(1):19–48, 2013.

161. Vijay Mahadevan and Nuno Vasconcelos. Spatiotemporal saliency in dynamic scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):171–177, 2010.

162. Martin Szummer and Rosalind W Picard. Temporal texture modeling. In *International Conference on Image Processing (ICIP)*, volume 3, pages 823–826. IEEE, 1996.

163. Arno Schödl, Richard Szeliski, David H Salesin, and Irfan Essa. Video textures. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 489–498. ACM Press/Addison-Wesley Publishing Co., 2000.

164. Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics (ToG)*, 22(3):277–286, 2003.

165. Johannes Ballé, Aleksandar Stojanovic, and Jens-Rainer Ohm. Models for static and dynamic texture synthesis in image and video compression. *IEEE Journal of Selected Topics in Signal Processing*, 5(7):1353–1365, 2011.

166. Petar M Djuric, Jayesh H Kotecha, Jianqui Zhang, Yufei Huang, Tadesse Ghirmai, Mónica F Bugallo, and Joaquin Miguez. Particle filtering. *Signal Processing Magazine, IEEE*, 20(5):19–38, 2003.

167. Payam Saisan, Gianfranco Doretto, Ying Nian Wu, and Stefano Soatto. Dynamic texture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–58. IEEE, 2001.

168. Antoni B Chan and Nuno Vasconcelos. Classifying video with kernel dynamic textures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–6. IEEE, 2007.

169. Byron Boots, Geoffrey J Gordon, and Sajid M Siddiqi. A constraint generation approach to learning stable linear dynamical systems. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1329–1336, 2007.

170. Zoubin Ghahramani and Geoffrey E Hinton. Variational learning for switching state-space models. *Neural computation*, 12(4):831–864, 2000.

171. Tony Van Gestel, Johan AK Suykens, Paul Van Dooren, and Bart De Moor. Identification of stable models in subspace identification by using regularization. *IEEE Transactions on Automatic Control*, 46(9):1416–1420, 2001.

172. Torbjørn Eltoft, Taesu Kim, and Te-Won Lee. On the multivariate laplace distribution. *IEEE Signal Processing Letters*, 13(5):300–303, 2006.

173. Yuhui Quan, Yan Huang, and Hui Ji. Dynamic texture recognition via orthogonal tensor dictionary learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 73–81, 2015.

174. Dmitry Chetverikov and Sándor Fazekas. On motion periodicity of dynamic textures. In *British Machine Vision Conference (BMVC)*, pages 167–176. Citeseer, 2006.

175. Roberto Costantini, Luciano Sbaiz, and Sabine Süsstrunk. Higher order svd analysis for dynamic texture synthesis. *IEEE Transactions on Image Processing*, 17(1):42–52, 2008.

176. Renaud Péteri and Dmitry Chetverikov. Dynamic texture recognition using normal flow and texture regularity. In *Pattern Recognition and Image Analysis*, pages 223–230. Springer, 2005.

177. Zhixiang Ren, Shenghua Gao, Deepu Rajan, Liang-Tien Chia, and Yun Huang. Spatiotemporal saliency detection via sparse representation. In *2012 IEEE International Conference on Multimedia and Expo (ICME)*, pages 158–163. IEEE, 2012.

178. Bernard Ghanem and Narendra Ahuja. Sparse coding of linear dynamical systems with an application to dynamic texture recognition. In *20th International Conference on Pattern Recognition (ICPR)*, pages 987–990. IEEE, 2010.

179. Bernard Ghanem and Narendra Ahuja. Maximum margin distance learning for dynamic texture recognition. In *European Conference on Computer Vision (ECCV)*, pages 223–236. Springer, 2010.

180. Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 129–136. ACM, 1995.

181. Arunkumar Ravichandran, Rizwan Chaudhry, and René Vidal. View-invariant dynamic texture recognition using a bag of dynamical systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1651–1657. IEEE, 2009.

182. Laurens van der Maaten. Learning a parametric embedding by preserving local structure. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, (AISTATS)*, volume 5, pages 384–391, 2009.

183. Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

184. Jason Weston, Frédéric Ratle, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Proceedings of the Twenty-Fifth International Conference (ICML)*, volume 307, pages 1168–1175, 2008.

185. Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

186. Peter N Belhumeur, João P Hespanha, and David Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

187. Shuicheng Yan, Dong Xu, Benyu Zhang, Hong-Jiang Zhang, Qiang Yang, and Stephen Lin. Graph embedding and extensions: a general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40–51, 2007.

188. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

189. Deng Cai, Xiaofei He, Jiawei Han, and Hong-Jiang Zhang. Orthogonal laplacianfaces for face recognition. *IEEE Transactions on Image Processing*, 15(11):3608–3614, 2006.

190. Deng Cai, Xiaofei He, and Jiawei Han. Semi-supervised discriminant analysis. In *IEEE 11th International Conference on Computer Vision (ICCV)*, pages 1–7. IEEE, 2007.

191. Mehrtash T Harandi, Mathieu Salzmann, and Richard Hartley. From manifold to manifold: geometry-aware dimensionality reduction for spd matrices. In *European Conference on Computer Vision (ECCV)*, pages 17–32. Springer, 2014.

192. Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 720–729, 2015.

193. Zhiwu Huang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Projection metric learning on grassmann manifold with application to video based face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 140–149, 2015.

194. Effrosyni Kokiopoulou and Yousef Saad. Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2143–2156, 2007.

195. Jian Yang, David Zhang, Jing-yu Yang, and Ben Niu. Globally maximizing, locally minimizing: unsupervised discriminant projection with applications to face and palm biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):650–664, 2007.

196. Hwann-Tzong Chen, Huang-Wei Chang, and Tyng-Luh Liu. Local discriminant embedding and its variants. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 846–853. IEEE, 2005.

197. Liang Sun, Shuiwang Ji, and Jieping Ye. *Multi-label dimensionality reduction*. CRC Press, 2013.

198. Guoxian Yu, Guoji Zhang, Carlotta Domeniconi, Zhiwen Yu, and Jane You. Semi-supervised classification based on random subspace dimensionality reduction. *Pattern Recognition*, 45(3):1119–1135, 2012.

199. Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16(16):321–328, 2004.

200. Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *The Journal of Machine Learning Research*, 7:2399–2434, 2006.

201. Feiping Nie, Dong Xu, Ivor Wai-Hung Tsang, and Changshui Zhang. Flexible manifold embedding: A framework for semi-supervised and unsupervised dimension reduction. *IEEE Transactions on Image Processing*, 19(7):1921–1932, 2010.

202. Mingbo Zhao, Zhao Zhang, and Tommy WS Chow. Trace ratio criterion based generalized discriminative learning for semi-supervised dimensionality reduction. *Pattern Recognition*, 45(4):1482–1499, 2012.

203. Qiaolin Ye, Ning Ye, Chunxia Zhao, Tongming Yin, and Haofeng Zhang. Flexible orthogonal semisupervised learning for dimension reduction with image classification. *Neurocomputing*, 144:417–426, 2014.

204. Yi Huang, Dong Xu, and Feiping Nie. Semi-supervised dimension reduction using trace ratio criterion. *IEEE Transactions on Neural Networks and Learning Systems*, 23(3):519–526, 2012.

205. Jonathan J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

206. Deng Cai, Xiaofei He, and Jiawei Han. Spectral regression for efficient regularized subspace learning. In *IEEE 11th International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.

207. Athinodoros S Georghiades, Peter N Belhumeur, and David J Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

208. Haichao Zhang, Jianchao Yang, Yanning Zhang, Nasser M Nasrabadi, and Thomas S Huang. Close the loop: Joint blind image restoration and recognition with sparse representation prior. In *IEEE International Conference on Computer Vision (ICCV)*, pages 770–777. IEEE, 2011.

209. M Ranzato, Fu Jie Huang, Y-L Boureau, and Yann LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

210. Ning Zhou, Yi Shen, Jinye Peng, and Jianping Fan. Learning inter-related visual dictionary for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3490–3497. IEEE, 2012.

211. S Nayar, Sammeer A Nene, and Hiroshi Murase. Columbia object image library (coil 100). *Department of Comp. Science, Columbia University, Tech. Rep. CUCS-006-96*, 1996.

212. Will Zou, Shenghuo Zhu, Kai Yu, and Andrew Y Ng. Deep learning of invariant features via simulated fixations in video. In *Advances in Neural Information Processing Systems*, pages 3212–3220, 2012.

213. Hossein Mobahi, Ronan Collobert, and Jason Weston. Deep learning from temporal coherence in video. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 737–744. ACM, 2009.

214. Junping Zhang, Ziyu Xie, and Stan Z Li. Prime discriminant simplicial complex. *IEEE Transactions on Neural Networks and Learning Systems*, 24(1):133–144, 2013.

215. Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

216. Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

217. Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

218. David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

219. Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1469–1472. ACM, 2010.

220. Hongping Cai, Krystian Mikolajczyk, and Jiri Matas. Learning linear discriminant projections for dimensionality reduction of image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):338–352, 2011.

221. Rémi Gribonval, Rodolphe Jenatton, Francis Bach, Martin Kleinsteuber, and Matthias Seibert. Sample complexity of dictionary learning and other matrix factorizations. *IEEE Transactions on Information Theory*, 61(6):3469–3486, 2015.

222. Arunkumar Ravichandran, Rizwan Chaudhry, and Rene Vidal. Categorizing dynamic textures using a bag of dynamical systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):342–353, 2013.

223. Adeel Mumtaz, Emanuele Coviello, Gert RG Lanckriet, and Antoni B Chan. Clustering dynamic textures with the hierarchical em algorithm for modeling video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1606–1621, 2013.

224. Adeel Mumtaz, Emanuele Coviello, Gert RG Lanckriet, and Antoni B Chan. A scalable and accurate descriptor for dynamic textures using bag of system trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(4):697–712, 2015.