Multi-Label Dimensionality Reduction

by

Liang Sun

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved July 2011 by the
Graduate Supervisory Committee:

Jieping Ye, Chair
Baoxin Li
Huan Liu
Hans Mittelmann

ARIZONA STATE UNIVERSITY

August 2011

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

ABSTRACT

Multi-label learning, which deals with data associated with multiple labels simultaneously, is ubiquitous in real-world applications. To overcome the curse of dimensionality in multi-label learning, in this thesis I study multi-label dimensionality reduction, which extracts a small number of features by removing the irrelevant, redundant, and noisy information while considering the correlation among different labels in multi-label learning. Specifically, I propose Hypergraph Spectral Learning (HSL) to perform dimensionality reduction for multi-label data by exploiting correlations among different labels using a hypergraph. The regularization effect on the classical dimensionality reduction algorithm known as Canonical Correlation Analysis (CCA) is elucidated in this thesis. The relationship between CCA and Orthonormalized Partial Least Squares (OPLS) is also investigated. To perform dimensionality reduction efficiently for large-scale problems, two efficient implementations are proposed for a class of dimensionality reduction algorithms, including canonical correlation analysis, orthonormalized partial least squares, linear discriminant analysis, and hypergraph spectral learning. The first approach is a direct least squares approach which allows the use of different regularization penalties, but is applicable under a certain assumption; the second one is a two-stage approach which can be applied in the regularization setting without any assumption. Furthermore, an online implementation for the same class of dimensionality reduction algorithms is proposed when the data comes sequentially. A Matlab toolbox for multi-label dimensionality reduction has been developed and released. The proposed algorithms have been applied successfully in the *Drosophila* gene expression pattern image annotation. The experimental results on some benchmark data sets in multi-label learning also demonstrate the effectiveness and efficiency of the proposed algorithms.

To my dear parents, for your love, understanding, support and encouragement.

ACKNOWLEDGEMENTS

First and foremost, I would like to gratefully and sincerely thank my advisor, Prof. Jieping Ye, for his guidance, understanding, patience, and most importantly, his friendship during my Ph.D. studies at Arizona State University. His infectious enthusiasm and unlimited zeal have been major driving forces through my graduate career. This dissertation would have never been possible without his support. The experiences with him are my lifelong assets.

My research has also benefited tremendously from various collaborations over the years. I would particularly like to thank Prof. Huan Liu (at Arizona State University), Prof. Baoxin Li (at Arizona State University), Prof. Hans D. Mittelmann (at Arizona State University), Prof. Guoliang Xue (at Arizona State University), Dr. Bill William Pavlicek (at Mayo Clinic), Dr. Keshu Zhang (at Applied Research & Technology Center, Motorola), Dr. Haifeng Li (at Applied Research & Technology Center, Motorola) for many thoughtful conversations.

My thanks also go to all the members of the Machine Learning group at Arizona State University, particularly Betul Ceran, Jianhui Chen, Shuiwang Ji, Ji Liu, Jun Liu, Rinkal Patel, Jun Shi, Qian Sun, Shuo Xiang, Sen Yang, Lei Yuan, Jiayu Zhou. Each and every member of the group has helped me in different ways by providing useful feedback and insightful discussions. It has been a great pleasure working with them during my Ph.D. studies.

My time at Arizona State University was made enjoyable in large part due to the many friends that became a part of my life. Their support and care helped me overcome setbacks and stay focused on my research. I greatly value their friendship. I would like to acknowledge Shuai Huang, Wen Jin, Tingting Ma, Yunsong Meng, Yang Qin, Lei Tang, Guannan Wang, Zheshen Wang, Yin Yin, Hang Zhang, Zheng Zhao, Zhibin Zhou, and all other friends who have made Tempe feel like a home. In particular, special thanks go to Yin Yin and Zhibin Zhou since it was so enjoyable to live with them in the past five years.

Last, but not least, I would like to thank my parents for their love, understanding, support and encouragement in the past five years. This dissertation is dedicated to them and I am deeply indebted to them for their unwavering faith in me.

This work is supported by National Science Foundation under grants IIS-0612069 and IIS-0953662.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Introduction to Multi-Label Learning

Multi-label learning deals with data associated with multiple labels simultaneously. In traditional classification, each instance is relevant to only one label. However, in many real-world problems, one instance may belong to multiple labels. For example, in news categorization, a piece of news regarding Apple's release of new iPhone is associated with both the label *business* and the label *technology*. In other words, each instance is associated with a set of labels instead of only one label. One distinguishing difference between multi-label learning and traditional binary or multi-class learning is that the labels in multi-label learning are not mutually exclusive, leading to the fact that each instance may belong to multiple labels. Thus, one of the challenges of multi-label learning is how to effectively utilize the correlation among different labels in classification.

In our discussion, we assume that each instance in the training data set is represented by a feature vector and the associated label set. Multi-label learning is the task of predicting the label sets of unseen instances by building a classifier based on the training data set. Formally, let $\mathscr{X}$ and $\mathscr{Y}$ denote the instance space and the label space, respectively. In multi-label learning, the label space $\mathscr{Y}$ is defined as $\mathscr{Y} = \{0,1\}^k$, where k is the number of labels. Similar to traditional classification, given a training data set, the goal of multi-label learning is to learn a classifier $f : \mathscr{X} \to \mathscr{Y}$ which assigns a predicted label set for each instance $\mathbf{x} \in \mathscr{X}$. Specifically, the output of the classifier f given the instance $\mathbf{x} \in \mathscr{X}$ is

$$f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_k(\mathbf{x})]^T, \tag{1.1}$$

where $f_j(\mathbf{x})\,(j = 1, \cdots, k)$ is the either 1 or 0 indicating the association of $\mathbf{x}$ to the jth label. In the following discussion, the set of labels is denoted as $\mathscr{L} = \{C_1, \cdots, C_k\}$.

Multi-label learning has applications in many real-world problems, such as text categorization [1, 2], semantic annotation of images [3, 4], functional genomics [2, 5], 3D hand pose estimation [6], and biological literature classification [7]. We will discuss the applications of multi-label learning in more detail in Section 1.2.

Motivated by the increasing number of applications, multi-label learning has recently attracted significant attention in the literature [8–10]. Many algorithms have been proposed to solve multi-label learning, and roughly they can be divided into two categories:

1. *Problem transformation*, which transforms the multi-label learning problem into a series of single-label problems first, and then solves them using existing algorithms.

2. *Algorithm adaption*, which solves the multi-label problem directly by adapting existing algorithms for single label.

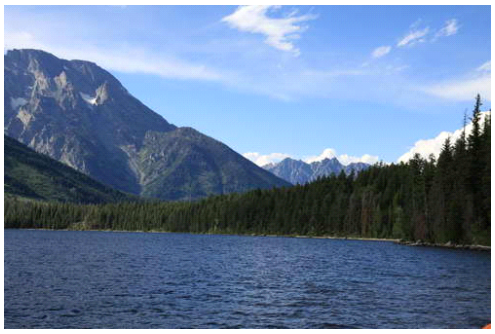
We will review existing multi-label learning algorithms in more detail in Section 1.3.

In this thesis, we focus on the *multi-label dimensionality reduction*, which extracts a small number of features by removing the irrelevant, redundant, and noisy information while considering the correlation among different labels in multi-label learning. Similar to other machine learning and data mining tasks, multi-label learning also suffers from the so-called *curse of dimensionality* [11]. Although dimensionality reduction is well studied in the literature, limited progress has been made on multi-label dimensionality reduction [12–14]. In this thesis, we develop and analyze novel methods for effectively capturing the label correlation for multi-label dimensionality reduction. In addition, we propose to develop efficient implementations of dimensionality reduction algorithms, including both existing algorithms and newly proposed ones.

In the rest of this chapter, we will briefly introduce multi-label learning and dimensionality reduction, including their applications, existing algorithms and related work. We will also highlight the main challenges of multi-label dimensionality reduction.

1.2 Applications of Multi-Label Learning

Multi-label learning has been applied successfully in many real-world applications. In this section, we introduce several applications, including scene classification, text classification, functional genomics and *Drosophila* gene expression pattern image annotation.



(A) Scene 1

(B) Scene 2

Figure 1.1: Examples of multi-label scenes. The Scene 1 is associated with the label "*lake*" and the label "*mountain*", and the Scene 2 is associated with the label "*geyser*" and the label "*mountain*".

Scene Classification

Humans are extremely proficient at perceiving natural scenes and understanding their contents. In semantic scene classification, the task is to determine the associated labels such as *mountains*, *lakes*, or *parties* for given images. Scene classification has a lot of applications in many areas, such as content-based image indexing and organization, and content-sensitive image enhancement [4]. For example, many current digital library systems support content-based image retrieval, which allows the user to search images similar to a specified query image [15]. In this case, knowing the label of the query image can reduce the search space and improve the retrieval accuracy.

A natural scene may be associated with multiple labels. Hence, scene classification can be modeled as a multi-label learning problem. For example, Figure 1.1(A) shows an image associated with the label "*lake*" and the label "*mountain*"; Figure 1.1(B) shows an image associated with the label "*geyser*" and the label "*mountain*".

Text Categorization

Text Categorization (TC) is the task of classifying text documents under one or more of a set of predefined categories or subject codes [16, 17]. Originally dated back to the early 1960s, the effectiveness of text categorization has been improved significantly in the past 20 years mainly due to the use of advanced machine learning methods [16]. Text categorization has been ap-

plied in various fields, including web page categorization under hierarchical labels, detection of text genre, classifying text (or hypertext) documents given a predefined label set, personalized information delivery and filtering unsuitable content [16, 17]. Typically, the predefined labels (or categories) in text categorization are not assumed to be mutually exclusive, thus the text categorization can naturally be modeled as a multi-label learning problem. For instance, consider labels *business, technology, entertainment*, and *politics* in news categorization, a news article about Apple's release of new iPhone may be labeled with both the label *business* and the label *technology*.

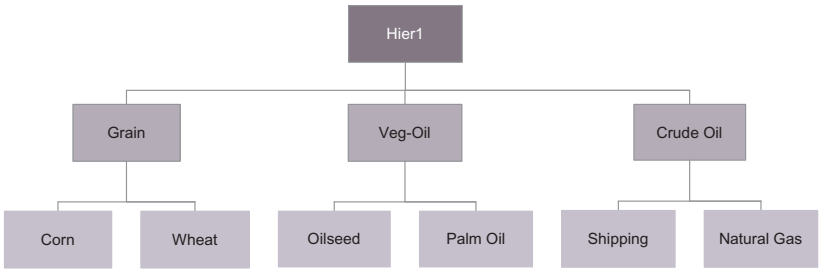
When applying multi-label learning to perform text categorization, the first step is to transform documents into some representation suitable for the learning algorithm, such as the vector space model [2] and the binary representation [18]. In the past, many multi-label learning algorithms have been proposed to perform text categorization [1, 2, 19–22]. One well-known algorithm in multi-label text categorization is BoosTexter, which extends the classical boosting algorithm AdaBoost [23] specifically to handle multi-label data. Some other algorithms include the Bayesian approach [1] using the mixture model and the EM algorithm, the Maximal Figure-of-Merit (MFoM) approach [20]. We will review existing multi-label learning algorithms in Section 1.3.

One widely used benchmark data set in multi-label text categorization is the Reuters-21578 data set[1]. The data was originally collected and labeled by Carnegie Group, Inc. and Reuters, Ltd. in the course of developing the CONSTRUE text categorization system. Reuters-21578 consists of 21,578 Reuters newswire documents that appeared in 1987. Almost all documents in Reuters collection come with title, dateline and text body, and the number of topics (labels) is 135. In particular, three widely used subsets of the Reuters-21578 data set have been extracted [18] by identifying the labels that suggest parent-child relationships, and the labels are organized in a hierarchical structure as shown in Fig 1.2. Note that the roots of the three category trees are virtual categories.

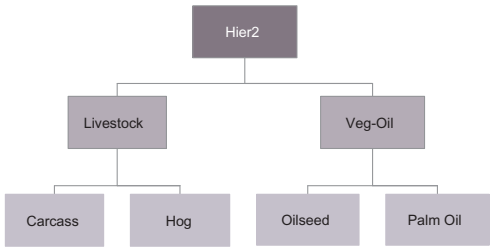
Another data set which becomes more popular for text categorization in recent years is the Reuters Corpus Volume 1 (RCV1) data set[2] [24]. The RCV1 data set consists of over

[1] http://www.research.att.com/∼lewis/reuters21578.html

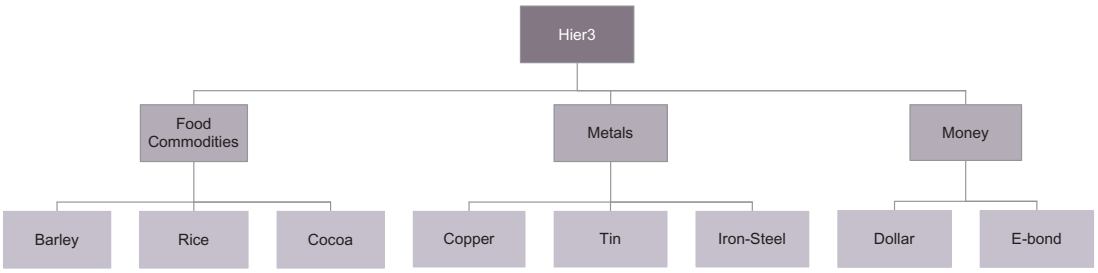
[2] http://www.daviddlewis.com/resources/testcollections/rcv1/



(A) Data set Hier1



(B) Date set Hier2



(C) Date set Hier3

Figure 1.2: The Label structures in the Reuters-21578 data set.

800,000 manually categorized newswire stories recently made available by Reuters, Ltd. for research purposes. Similar to the Reuters-21578 data set, the labels in RCV1 data set are organized in a hierarchical structure. The original data set is referred as RCV1-v1, and a corrected version called RCV1-v2 is generated and becomes more popular in text categorization research. More details on this data set can be found in [24].

Functional Genomics

Functional genomics is an important field in bioinformatics. It studies gene and protein functions on a large scale by conducting analysis on a vast amount of data collected by genomic

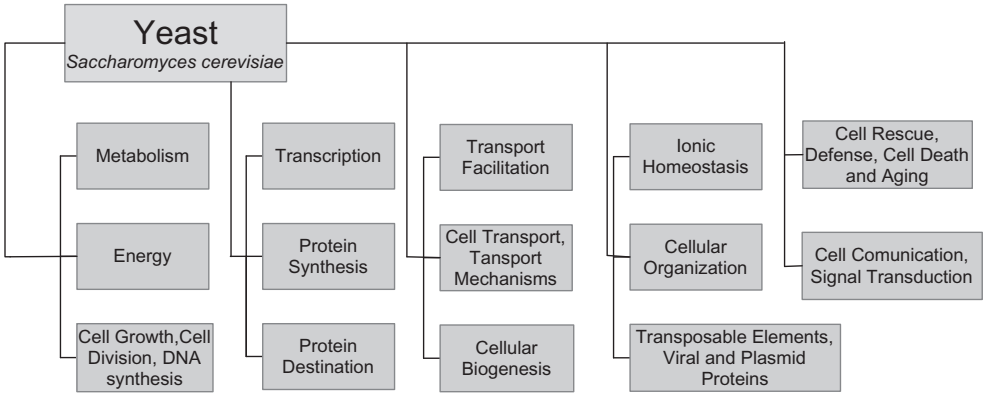


Figure 1.3: The labels selected in the Yeast data set.

projects [25, 26]. For example, the DNA microarrays allow researchers to simultaneously measure the expression levels of thousands of different genes, and overwhelming amounts of data are produced [27]. Recently, much research focuses on automatic analysis of microarray data [26]. In automated gene expression analysis, the task is to predict the functions for genes. Generally it is based on the assumption that genes with similar functions have similar expression profiles in cells [25]. Note that each gene may be associated with multiple functions in functional genomics. When the functions are considered as labels, the function prediction problem in functional genomics can be modeled as a multi-label learning problem.

A widely used benchmark data set in multi-label learning for functional genomics is the Yeast data set [28, 29]. The Yeast data set consists of microarray expression data and phylogenetic profiles from the budding yeast *Saccharomyces cerevisiae*. It contains 2417 samples, and each sample is represented by a 103-dimensional feature vector[3]. Each sample (gene) is associated with a set of functional labels whose maximum size can be potentially more than 190. The functional classes (labels) are organized in a tree structure, which is known in the literature [28, 29]. This data set is preprocessed in [28] and only the function classes in the top hierarchy are selected, resulting in a total of 14 labels, as shown in Figure 1.3.

Drosophila *Gene Expression Pattern Image Annotation*

Detailed knowledge of the expression and interaction of genes is crucial to deciphering the mechanisms underlying cell-fate specification and tissue differentiation. The fruit fly *Drosophi-*

[3]http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/datasets/multilabel.html#yeast

la melanogaster is one of the model organisms in developmental biology, and its patterns of gene expression have been studied extensively [30–33]. The comprehensive atlas of spatial patterns of gene expression during *Drosophila* embryogenesis has been created by *in situ* hybridization techniques, and the patterns are documented in the form of digital images [32, 34–36]. Comparative analysis of gene expression pattern images can potentially reveal new genetic interactions and yield insights into the complex regulatory networks governing embryonic development [32, 37–39].

To facilitate pattern comparison and searching, the images of *Drosophila* gene expression patterns are annotated with anatomical and developmental ontology terms using a controlled vocabulary [32, 34]. The annotation is performed not only for each terminally differentiated embryonic structure but also for the developmental intermediates that correspond to it. Four general classes of terms, called anlage *in statu nascendi*, anlage, primordium, and organ (ordered in terms of developmental time), are used in the annotation. Such an elaborate naming scheme describes a developing "path", starting from the cellular blastoderm stage until organs are formed, that documents the dynamic process of *Drosophila* embryogenesis. Due to the overwhelming complexity of this task, the images are currently annotated manually by human experts. However, the number of available images produced by high-throughput *in situ* hybridization is now rapidly increasing [37, 38, 40–42]. It is therefore tempting to design computational methods for the automated annotation of gene expression patterns.

The annotated terms from the controlled vocabulary can be considered as labels in multi-label classification. Since a variable number of terms from the controlled vocabulary can be assigned to a group of pattern images, the problem of gene expression pattern annotation can be modeled as a multi-label learning problem. In the current image database, annotation terms are associated with a group of patterns sharing a subset of the named structures. Thus, the difficulty of this problem is that the labels (annotation terms) are assigned to groups of patterns rather than to individual images. Figure 6.1 shows sample *Drosophila* gene expression pattern images and their corresponding annotated terms.

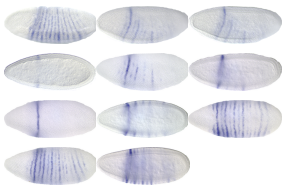
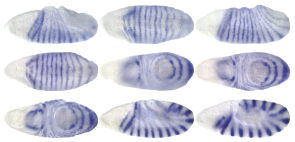
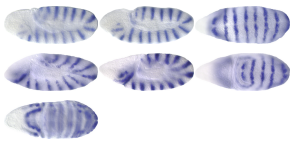
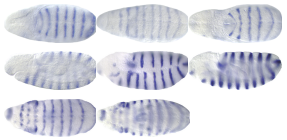
Stage range		BDGP terms
4-6		dorsal ectoderm anlage in statu nascendi mesectoderm anlage in statu nascendi segmentally repeated trunk mesoderm anlage in statu nascendi ventral ectoderm anlage in statu nascendi
7-8		dorsal ectoderm primordium hindgut anlage mesectoderm primordium procephalic ectoderm anlage trunk mesoderm primordium P2 ventral ectoderm primordium P2
9-10		inclusive hindgut primordium mesectoderm primordium procephalic ectoderm primordium trunk mesoderm primordium ventral ectoderm primordium
11-12		atrium primordium brain primordium clypeo-labral primordium dorsal epidermis primordium gnathal primordium head epidermis primordium P1 hindgut proper primordium midline primordium ventral epidermis primordium ventral nerve cord primordium

Figure 1.4: Sample image groups and their associated terms (class labels) in the BDGP database (http://www.fruitfly.org) for the segmentation gene *engrailed* in 4 stage ranges.

1.3 Prior Work on Multi-Label Learning: Challenges and Progress

In this section we give a brief review of the state-of-the-art multi-label learning algorithms in the literature.

Existing multi-label classification algorithms [1, 2, 5, 19–21, 28, 43–53] can be divided into two categories [8]: 1) problem transformation, and 2) algorithm adaption. In the first approach, the multi-label learning problem is transformed into a series of single-label classification problems so that existing single-label learning algorithms can readily be applied. The second approach, i.e., algorithm adaption, adapts existing single-label classification algorithms to handle multi-label classification directly. In [54], the algorithms are categorized based on the order of correlations among labels used in the algorithm. Specifically, the first-order approach

only decomposes the multi-label learning problem into a sequence of independent binary classification problems; the second-order approach considers the pairwise relations between labels; the higher-order approach considers the higher-order relations among labels.

Besides multi-label classification, another popular problem in multi-label learning is label ranking (LR) [55] which learns an ordering of the labels based on their relevance to a given instance. In this thesis we mainly focus on Multi-Label Classification (MLC). More details on the connections between multi-label classification and label ranking can be found in [8].

In the following, we first highlight some challenges of multi-label learning and then review some multi-label learning algorithms. We follow the categorization in [8] to survey these multi-label learning algorithms, i.e., problem transformation and algorithm adaption.

Challenges of Multi-Label Learning

In comparison with traditional binary classification and multi-class classification, the generality of multi-label classification inevitably makes it more challenging to solve. Here we list some of the most fundamental challenges which exist in the successful application of multi-label learning in real-world problems.

The first challenge lies in how to effectively exploit the label structure to improve the performance of multi-label classification. In multi-label learning, the labels are often corrected since they are not mutually exclusive. In this case, how to measure and capture the correlation in the label space for improved prediction is crucial in multi-label learning. So far, some algorithms have been proposed in the literature to capture the label correlations [8, 51, 54, 56, 57]. However, how to effectively make use of the correlation in the label space still remains an open question. Furthermore, as we discussed in Section 1.2, in some applications of text categorization and bioinformatics, the labels are organized in a hierarchical structure or in a Directed Acyclic Graph (DAG). It is clear that taking advantage of such structures can lead to improved classification performance. For example, in a general-to-specific label tree structure, an instance associated with a specific label should also be associated with its parent label. How to effectively use these complex structures in specific applications also remains a challenging research question [58].

The second challenge lies in the classification imbalance problem in multi-label learning. When each label is treated independently, it can be observed that most instances are irrelevant to a specific label. Since the number of positive instances for some label is significantly less than the number of negative instances, it is difficult to build a highly accurate classifier for these labels without considering the correlations among different labels. Thus, the research question is how to correctly classify these rare classes given their limited numbers of training instances. In fact, the classification imbalance problem becomes worse when the number of labels is relatively large.

The third challenge concerns the effectiveness and efficiency of multi-label learning for large-scale problems, especially when the data dimensionality is high and the number of labels is large. Multi-label learning also suffers from the curse of dimensionality, and many existing multi-label learning algorithms fail when the dimensionality is high since data points become sparse and far apart from each other. We will provide more details on how to deal with the high-dimensional data in multi-label learning in Section 1.4. In addition, the computational cost of multi-label learning increases as the number of labels increases. For example, the Binary Relevance (BP) and Label Power-set (LP) (both will be discussed in detail in the next subsection) can only work when the label size is comparatively small. Recently, some algorithms have been proposed [53, 59] to deal with a large number of labels. For example, the dimensionality of the label space is reduced by using a random projection in [53]. However, the computation of the mapping from the reduced space to the original label space is expensive, which limits its applicability in large-scale problems. In addition, when the number of labels is large it is more difficult to maintain a large number of prediction models in memory [8].

Problem Transformation

The most straightforward approach to handling the multi-label learning problem is to transform it into a series of single-label classification problems so that existing algorithms for single-label classification can readily be applied. The key idea is to eliminate the label overlapping in the original label space. Compared with the algorithm adaption scheme, the problem transformation scheme is more flexible since any off-the-shelf single-label classifier can be applied after problem transformation. In this subsection, we will introduce three different transformation schemes: COpy transformation (CO), Binary Relevance (BR), and Label Power-set (LP).

Copy Transformation (CO)

The instance COpy transformation (CO) converts the multi-label problem into a multi-class problem by performing a transformation on the data set. Specifically, for each example $(\mathbf{x}_i, \mathbf{y}_i)$ in the original multi-label problem, we replace it with a set of new examples $\{(\mathbf{x}_i, C_j) | \mathbf{y}_i(j) = 1\}$. In other words, a new example $(\mathbf{x}_i, C_j)$ is generated if $\mathbf{x}_i$ is associated with C_j in the original multi-label problem. Some variation of this transform also adds some weight for the new instances, e.g., $\frac{1}{\sum_j \mathbf{y}_i(j)}$ for each new example generated for the original example $(\mathbf{x}_i, \mathbf{y}_i)$. By applying the copy transformation, the original multi-label learning problem is transformed into a multi-class classification problem. The copy transformation of a toy multi-label learning problem is illustrated in Figure 1.5.

Binary Relevance (BR)

Binary Relevance (BR), or one-against-all approach, is the most well-known and most commonly used transformation approach for multi-label learning in the literature [8, 55]. It has been used widely as a baseline method to evaluate the performance of multi-label learning algorithms. Specifically, BR learns k binary classifiers for all labels in $\mathscr{L}$ independently. For each label C_j, it uses all instances associated with it as the positive training samples, and the remaining instances as negative samples. Based on the new training data set, a classifier f_j can be learned for the label C_j. After learning k classifiers $\{f_1, \cdots, f_k\}$ for all k labels, the prediction for the new instance $\mathbf{x}$ is given by $f(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_k(\mathbf{x})]^T$.

Binary Relevance is a straightforward transformation approach and can be combined with many state-of-the-art binary classification algorithms such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN). The drawback of binary relevance is that it does not consider correlations among the labels in the label transformation as all labels are treated independently. In addition, when the number of labels is large, it may not scale to large-size data sets since a binary classifier has to be constructed for each label. Binary relevance also suffers from the class imbalance problem, i.e., the number of positive instances is significantly less than the number of negative instances for some labels due to the typical sparsity of labels in multi-label data.

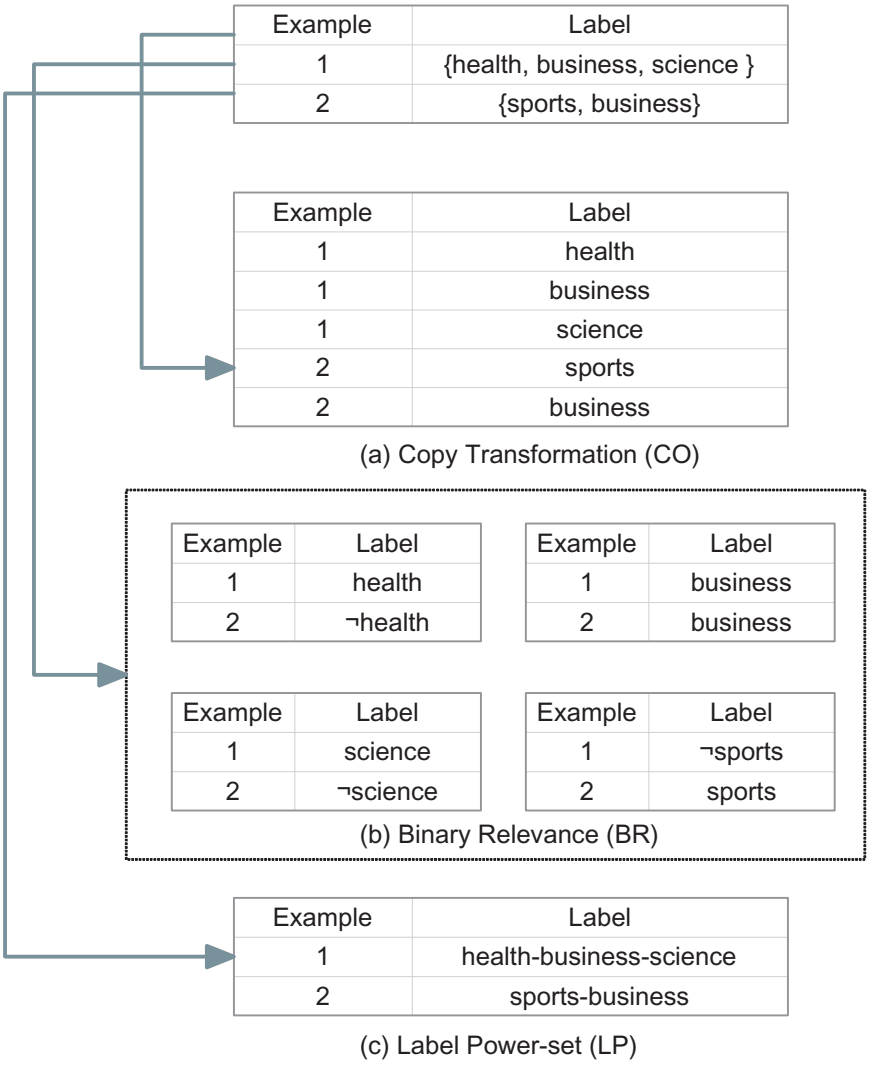


Figure 1.5: Illustration of different transformations of a toy multi-label learning problem.

The binary relevance transformation of a toy multi-label learning problem is illustrated in Figure 1.5.

Label Power-Set (LP)

Another popular approach is the Label Power-set (LP) [4], which treats each distinct label set in the training data set of the multi-label learning problem as a new label. As a result, the multi-label problem can be transformed into a series of single-label classification problems. Given a new data instance, LP will output the most probable classes, i.e., a set of labels in the original problem.

Compared with the copy transformation and the binary relevance, the label power-set transformation considers the correlations and dependencies among different labels explicitly. One drawback of the label power-set transformation is that the label set which does not appear in the training data set of the original multi-label learning problem cannot be handled properly. In addition, the label power-set transformation is computationally expensive since the number of labels is increased exponentially. In the worst case, the number of labels after LP transformation is $2^k - 1\,(k \geq 2)$, which means that the computational cost will increase exponentially after LP transformation. Furthermore, many of these new labels tend to be very sparse which leads to very severe class imbalance problem due to the fact that many of these new labels are only associated with a limited number of instances.

The label power-set transformation of a toy multi-label learning problem is illustrated in Figure 1.5.

Besides the three classical transformation approaches, some novel transformation methods which combine both binary relevance and label power-set have also been proposed in the literature recently [60].

Single-Label Classification After Transformation

After label transformation, the single-label classification algorithms can be applied readily to predict the final label(s) of new data instances. Some well-known single-label algorithms which have been applied in multi-label classification include SVM, decision tree, k-nearest neighbors, naive Bayes classifier, artificial neural network, etc. [61]. The comparison between different single-label algorithms after label transformation has been performed in [10] and [61]. As reported by [10], SVM provides the best tradeoff between predictive performance and computational complexity.

Algorithm Adaption

In this subsection, we will review some multi-label learning algorithms by adapting existing single-label classification algorithms in the literature. One key issue in algorithm adaption is how to process the label overlapping and improve prediction performance by considering the correlation among different labels in the label space.

Algorithms based on Decision Tree

In [43], the classical decision tree algorithm C4.5 is adapted to handle multi-label problems. For a given data set, the entropy is redefined to process instances associated with multiple labels. Formally, given a set of instances associated with multiple labels $\mathscr{L} = \{C_1, \ldots, C_k\}$, the entropy is defined as

$$\text{Entropy}(\mathscr{L}) \stackrel{\triangle}{=} -\sum_{i=1}^{k} \left(\mathbb{P}(C_i)\log\mathbb{P}(C_i) + (1-\mathbb{P}(C_i)\log(1-\mathbb{P}(C_i)))\right), \tag{1.2}$$

where $\mathbb{P}(C_i)$ is the probability (relative frequency) of class C_i, and $1-\mathbb{P}(C_i)$ is the probability of not being member of class C_i.

Following the rule of maximum information gain which is the difference of entropy after splitting, we can construct a decision tree to handle the multi-label data set. On the other hand, in multi-label learning we have to allow leaves of the tree to potentially be a set of labels, i.e., the outcome of a classification of an instance can be a set of labels.

In [5], the entropy is further extended to handle hierarchial multi-label classification where the labels are organized in a hierarchical structure.

Algorithms based on Probabilistic Framework

In [1] a Bayesian classification approach is proposed for multi-label classification. Specifically, a probabilistic generative model is constructed to model multiple labels associated with each document. The words in a document are produced by a mixture of word distributions, one for each topic (label). In the generative model, it first selects a set of classes (instead of only one class) associated with this document; it then produces a set of mixture weights for these selected classes. Based on these weights we can generate each word in the document as follows: we first select a class for this document using these weights, and then generate a single word based on the selected class. In classification, the Bayes rule is applied to predict the most probable class set given the document. In terms of parameter estimation in this generative model, the EM algorithm is applied to learn the distribution over mixture weights and the word distribution in each class's mixture component. Compared with single-label classification algorithms, the proposed generative model explicitly takes the correlations among different labels into account.

Two types of generative models called Parametric Mixture Models (PMMs) are also proposed in [21] for multi-label text categorization. The basic assumption of PMMs is very similar to that in [1]. Namely, the words in a document belonging to a label set can be regarded as a mixture of characteristic words related to each of these labels. For example, a document that belongs to both "*sports*" and "*music*" would consist of a mixture of characteristic words mainly related to both categories. Also efficient learning and prediction algorithms have been proposed for PMMs in [21].

Compared with other multi-label learning algorithms, both generative models proposed in [1] and [21] can only be applied for text categorization. More general models are desirable to handle the general multi-label learning problem.

Algorithms based on Support Vector Machines

In [28] an algorithm based on Support Vector Machines (SVM) called RankSVM is proposed. RankSVM tries to control the model complexity while minimizing the empirical error. In fact, it shares a lot of common properties with SVM. The key idea of the RankSVM algorithm is to use a novel ranking loss function to capture the characteristics of multi-label learning problem and then solve the resulting optimization problem accordingly.

Formally, for each label C_j, we construct a linear model:

$$f_j(\mathbf{x}_i) = \mathrm{sgn}(\mathbf{w}_j^T \mathbf{x}_i + \mathbf{b}_j).$$

Next we consider the ith sample $(\mathbf{x}_i, \mathbf{y}_i)$ in the training data set, where $\mathbf{x}_i \in \mathbb{R}^d$ and its corresponding label vector $\mathbf{y}_i \in \mathbb{R}^k$ is

$$\mathbf{y}_i = [\mathbf{y}_i(1), \mathbf{y}_i(2), \ldots, \mathbf{y}_i(k)]^T,$$

where $\mathbf{y}_i(j)$ is either 1 or 0 indicating the association of $\mathbf{x}_i$ to the jth label. Denote $\mathbf{Y}_i = \{j | \mathbf{y}_i(j) = 1\}$ and $\bar{\mathbf{Y}}_i = \{j | \mathbf{y}_i(j) = 0\}$, i.e., $\mathbf{Y}_i$ contains the label indices associated with $\mathbf{x}_i$ and $\bar{\mathbf{Y}}_i$ contains the label indices not associated with $\mathbf{x}_i$. Let the output of k linear models be $f(\mathbf{x}_i) = [f_1(\mathbf{x}_i)), f_2(\mathbf{x}_i), \ldots, f_k(\mathbf{x}_i)]^T$. The ranking loss function over $(\mathbf{x}_i, \mathbf{y}_i)$ is defined as

$$RL(f, \mathbf{x}_i, \mathbf{y}_i) = \sum_{i=1}^{n} \frac{1}{|\mathbf{Y}_i||\bar{\mathbf{Y}}_i|} \left| (q,l) \in (\mathbf{Y}_i \times \bar{\mathbf{Y}}_i) \text{ s.t. } f_q(\mathbf{x}_i) \leq f_l(\mathbf{x}_i) \right|. \tag{1.3}$$

The ranking loss defined in Eq. (1.3) measures the average fraction of pairs which are not correctly ordered.

The basic idea of RankSVM is to find a series of linear models $\{f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_k(\mathbf{x})\}$ such that the ranking loss function is minimized while a large margin is also achieved. In multi-label learning, the margin of $\mathbf{x}_i, \mathbf{y}_i$ is defined as

$$\min_{q \in \mathbf{Y}_i, l \in \bar{\mathbf{Y}}_i} \frac{(\mathbf{w}_q - \mathbf{w}_l)^T \mathbf{x}_i + \mathbf{b}_q - \mathbf{b}_l}{\left\| \mathbf{w}_q - \mathbf{w}_l \right\|}. \tag{1.4}$$

In fact, it is the signed ℓ_2 distance of $\mathbf{x}_i$ to the decision boundary. Similar to SVM, an optimization problem can be formed and solved.

In the above discussion, we assume that $f_j(\mathbf{x}) = \mathbf{w}_j^T \mathbf{x} + \mathbf{b}_j (1 \leq j \leq k)$ are linear models. These linear models can be extended to the kernel induced feature space by applying the kernel trick [62].

In [44] two methods for enhancing existing discriminative classifiers for multi-label classification are proposed. Specifically, the first one exploits correlation among different labels by combining text features and information about relationships between classes by constructing a new kernel for SVMs with heterogeneous features. The second one improves the margin of SVMs for better multi-label classification.

Algorithms based on Artificial Neural Network

In [2], a neural network algorithm called BP-MLL, i.e., BackPropagation for Multi-Label Learning, is proposed. It extends the backpropagation algorithm for multi-label learning by defining a novel error function which captures the characteristics of multi-label learning. Based on the proposed error function, the resulting minimization problem can be solved by gradient descent combined with the error backpropagation strategy [63]. Specifically, given the training data set $\{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, let the output of the neural network be $\{f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n)\}$ where $\mathbf{y}_i = [\mathbf{y}_i(1), \mathbf{y}_i(2), \ldots, \mathbf{y}_i(k)]^T$, $f(\mathbf{x}_i) = [f_1(\mathbf{x}_i)), f_2(\mathbf{x}_i), \cdots, f_k(\mathbf{x}_i)]^T$, and $\mathbf{y}_i(j)$ is either 1 or 0. Then the novel error function for multi-label learning in [2] is defined as follows:

$$E_z \stackrel{\triangle}{=} \sum_{i=1}^{n} E(\mathbf{x}_i), \tag{1.5}$$

where E_i is defined as

$$E(\mathbf{x}_i) \stackrel{\triangle}{=} \sum_{i=1}^{n} \frac{1}{|\mathbf{Y}_i||\bar{\mathbf{Y}}_i|} \sum_{q \in \in \mathbf{Y}_i, l \in \bar{\mathbf{Y}}_i} \exp\left(-(f_q(\mathbf{x}_i) - f_l(\mathbf{x}_i))\right). \tag{1.6}$$

In Eq. (1.6) $\mathbf{Y}_i = \{j | \mathbf{y}_i(j) = 1\}$ and $\bar{\mathbf{Y}}_i = \{j | \mathbf{y}_i(j) = 0\}$, i.e., $\mathbf{Y}_i$ contains the label indices associated with $\mathbf{x}_i$ and $\bar{\mathbf{Y}}_i$ contains the label indices not associated with $\mathbf{x}_i$. Thus $|\mathbf{Y}_i|$ is actually the number of labels associated with the instance $\mathbf{x}_i$. Note that in Eq. (1.6) we use the index q to denote the label index associated with $\mathbf{x}_i$ and use the index l to denote the label index not associated with $\mathbf{x}_i$. As a result, $f_q(\mathbf{x}_i) - f_l(\mathbf{x}_i)$ measures the difference between the outputs of the network on one label belonging to $\mathbf{x}_i$ and another label not belonging to it. Clearly, a larger difference between $f_q(\mathbf{x}_i)$ and $f_l(\mathbf{x}_i)$ leads to better classification performance. In Eq. (1.6) the exponential function is employed to severely penalize the difference if $f_q(\mathbf{x}_i)$ is much smaller than $f_l(\mathbf{x}_i)$ and make the error function smooth. Also note that there are $|\mathbf{Y}_i||\bar{\mathbf{Y}}_i|$ pairs of labels with one associated with $\mathbf{x}_i$ and another not associated with $\mathbf{x}_i$, thus we use $\frac{1}{|\mathbf{Y}_i||\bar{\mathbf{Y}}_i|}$ to normalize the error. The intuitive idea behind this error function is that the labels associated with an instance should be ranked higher than those not associated with that instance.

Algorithms based on k-Nearest Neighbor

In [45], a lazy learning algorithm named ML-kNN, i.e., Multi-Label k-Nearest Neighbor, which extends the k-Nearest Neighbor (kNN) to handle multi-label data, is proposed. The basic idea of ML-kNN is very similar to the traditional kNN. Specifically, the k nearest neighbors of a unseen instance are identified first in ML-kNN. Based on the statistical information gained from the label sets of these neighboring instances, the Maximum A Posterior (MAP) is applied to predict the label set for the unseen instance. Compared with traditional kNN, one advantage of ML-kNN is that the prior probabilities can be incorporated into the estimation.

In [46], a novel class balanced k-nearest neighbor approach for multi-label classification by emphasizing balanced usage of data from all classes is proposed.

Algorithms based on Ensemble Learning

BoosTexter [19] extends the popular boosting algorithm AdaBoost [23] to multi-label learning by the same authors. The basic idea of boosting is to combine many simple and less accurate classifiers called weak classifiers into a single highly accurate classifier. Generally the weak

classifiers are trained sequentially so that the instances more difficult to classify by the preceding classifiers are focused. In the simplest version of AdaBoost for single-label classification, the boosting algorithm maintains a set of important weights over training examples so that the weak classifiers can concentrate on those examples which are difficult to classify. In BoosTexter, it maintains a set of weights over both training examples and labels to handle multiple labels. In the boosting process, training examples and their corresponding labels which are difficult to classify get incrementally higher weights while examples and labels which are easy to classify get lower weights. In other words, we force the weak learning algorithm to focus on the combination of examples and labels which are difficult to predict and thus improve the overall performance of the ensemble algorithm. Specifically, two different extensions of AdaBoost have been proposed [19]. The first extension, AdaBoost.MH, focuses on the prediction of all correct labels while the second one, AdaBoost.MR, focuses on label ranking.

As we discussed in Section 1.3, the label power-set transformation explicitly considers the label correlation. Unfortunately, it suffers from the huge number of new labels and the majority of them are associated with very few positive samples. To overcome the limitations of the label power-set transformation, the RAndom k-labELsets (RAKEL) algorithm [48], an ensemble method for multi-label classification, is proposed recently.

Given the label set $\mathscr{L} = \{C_1, C_2, \ldots, C_k\}$, a set $S \subseteq \mathscr{L}$ with $|S| = q$ is called a q-labelset. The set of all distinct q-labelsets on $\mathscr{L}$ is denoted as $\mathscr{L}^q$. The RAKEL algorithm iteratively constructs a single-label weak learner from $\mathscr{X}$ to the selected q-labelset. Specifically, in the pth iteration of RAKEL, a q-labelset S_p is selected randomly from $\mathscr{L}^q$ without replacement. Then the weak learner constructs a single-label classifier $h_p : \mathscr{X} \to S_p$. Two parameters are specified by the user, e.g., the size of label set q and the number of iterations in RAKEL. It has been shown in [48] by using small q and an adequate number of iterations, RAKEL can model label correlations effectively. For the prediction of a new instance $\mathbf{x}$, we first apply all weak learners to obtain the binary decisions for all q-labelsets. Then RAKEL computes a ranking of all original labels by averaging the predictions of all weak learners. The final positive decision for a specific original label is made if the average is greater than a user-specified threshold.

RAKEL explicitly considers the correlations among different labels. One disadvantage of RAKEL is its efficiency. In the current RAKEL algorithm, the random selection is used to speed up the computation. On the other hand, parameter selection is also an issue in its practical application.

In [49], a multi-label learning algorithm called Model-Shared Subspace Boosting (MSSBoost) is proposed to reduce the information redundancy in the learning process. Specifically, it learns a number of base models across multiple labels, and each model is based on a random feature subspace and bootstrap data samples. By combining these shared subspace models, the decision function for each label can be jointly estimated.

Recently, the ensemble learning has also been applied in hierarchical multi-label learning [50].

Other Algorithms

In [51], an algorithm called Correlated Label Propagation (CLP) is proposed to explicitly model interactions between labels. CLP simultaneously propagates multiple labels from training data points to test data points. After formulating an optimization problem in CLP, an efficient algorithm based on properties of submodular functions is proposed to find its optimal solution.

In [20], a multi-label classification algorithm for text categorization is proposed. It generalizes the Maximal Figure-of-Merit (MFoM) [64] approach from binary classification to multi-label classification. In particular, all MFoM classifiers are trained simultaneously. Compared with binary MFoM classifiers which are trained independently, the resulting models are more robust, especially for the labels with a very small number of positive training examples.

In [52], an algorithm based on the Conditional Random Field (CRF) classification model which directly parameterizes label co-occurrences in multi-label classification is proposed.

In [53], an algorithm is proposed to make use of the sparsity in the label space, i.e., the majority of the data instances are only associated with a limited number of labels. The basic idea is to project the label vector onto a lower-dimensional space using some random matrix. And then the multi-label classification problem is transformed into a series of regression problem. In particular, it has been shown that the number of regression problems is only logarithmic

in the total number of possible labels. Using established theories and tools in compressed sensing [65, 66], the prediction in the regression model can be recovered to the label vector in the original label space.

In [57] a multi-label learning algorithm is proposed to solve the weak label problem. The weak label problem is quite similar to multi-instance learning. In weak label problem, the appearance of a label means that the instance is associated with this label, while the absence of a label does not imply that this label is not proper for this instance. In other words, the label information for a given instance in the training set is "partial". In the algorithm proposed in [57], a convex optimization problem is formulated to make sure that the classification boundary for each label goes across low density regions. The novel part of the proposed algorithm is that different similarities between instances are used for different labels. In addition, it is assumed these similarities can be derived from a group of low-rank base similarities.

In multi-instance learning [67], each object is described by a number of instances. Multi-instance learning has been successfully applied in many areas [68, 69]. Specifically, in multi-instance learning, the label of the object is determined by the labels of its corresponding instances. If all instances are negative, then the label of the object is negative. If there exists at least one positive instance, then the corresponding object is labeled as positive. The task of multi-instance learning is to learn a function which can predict the label for an unseen object given its relevant instances.

In [70], the multi-instance multi-label problem is studied. In multi-instance multi-label learning, each object is associated with not only multiple instances but also multiple labels. New algorithms called MIMLBOOST and MIMLSVM are proposed to solve multi-instance multi-label problems. The relationship between multi-instance multi-label learning and the traditional multi-class learning, multi-instance learning, and multi-label learning is also studied in [70].

1.4 Dimensionality Reduction for Multi-Label Learning

Introduction to Dimensionality Reduction

Recent technological innovations have allowed us to collect massive amounts of data with a large number of features, such as gene expression pattern images [71], microarray gene expression data [72], protein/gene sequences [73], text documents [74], and neuroimages [75]. The

proliferation of such data has facilitated knowledge discovery and pattern prediction/analysis using computational approaches [37, 72, 76, 77]. A common characteristic of such data is that the number of features is much larger than the sample size. The objective of this thesis is to study the theoretical and computational issues involved in high-dimensional data analysis in multi-label learning. While this goal is challenging, its pursuit will have a significant impact, as Donoho noted in [78]: "*The trend today is towards more observations but even more so, to radically larger numbers of variables .... Classical methods are simply not designed to cope with this kind of explosive growth of dimensionality of the observation vector. We can say with complete confidence that in the coming century, high-dimensional data analysis will be a very signicant activity.*"

One of the key issues in such data analysis is the so-called *curse of dimensionality* [11]. This term was coined by Bellman [11] to show that the number of samples required to estimate a function with a given level of accuracy grows exponentially with dimensions that it comprises. In other words, the curse of dimensionality means that for a given sample size, there is a maximum number of features above which the performance of an algorithm will degrade rather than improve. In fact, many data mining algorithms fail when the dimensionality is high [78,79] since data points become sparse and far apart from each other.

Dimensionality reduction, which extracts a small number of features by removing the irrelevant, redundant, and noisy information, is an effective way to mitigate the curse of dimensionality. Basically, dimensionality reduction transforms the high-dimensional data into a meaningful representation with a reduced dimensionality. Dimensionality reduction is important and widely used in many domains, such as data compression, noise removal, etc. By applying dimensionality reduction, the size of the data can be reduced significantly which requires less storage space and computational cost in the further processing while some important properties of the original data set are preserved.

Data visualization is another important application of dimensionality reduction. It is the study of the visual representation of data through graphical means and is effective in exploratory data analysis [80–83]. In data visualization, the data in the original high-dimensional space are represented as a data point in a 2- or 3-dimensional space. The task of data visualization is

to construct the new representation in 2- or 3-dimensional space in such a way that the essential properties or information in the original data space is preserved [81]. By applying dimensionality reduction, the data can be embedded into 2- or 3-dimensional space so that the analysts can explore the data in an intuitive manner [80]. For example, the nonlinear dimensionality reduction has been applied successfully in the visualization of gene expression compendia for retrieving relevant experiments [80].

Dimensionality reduction has been studied extensively in many areas (reviewed in [84–87]). These dimensionality reduction algorithms can be divided into different categories based on different criteria, e.g., supervised and unsupervised dimensionality reduction algorithms, linear and nonlinear dimensionality reduction algorithms, etc. In linear dimensionality reduction algorithms, the data are projected onto a lower-dimensional space through a linear mapping or transformation, e.g., Principal Component Analysis (PCA) [88] and Linear Discriminant Analysis (LDA) [89, 90]. Recently, some nonlinear dimensionality reduction algorithms have been proposed to reduce the data dimensionality in a nonlinear way [87, 91]. For example, in manifold learning, the goal is to find a low-dimensional nonlinear manifold embedded in the original high-dimensional space, which preserves the local properties of the data distribution. In supervised dimensionality reduction the label information is used so that the label discriminatory information can be preserved after projection, e.g., Linear Discriminant Analysis (LDA) [89, 90]. In unsupervised dimensionality reduction, no label information is used and the goal is to preserve some properties of the data, e.g., the variability in Principal Component Analysis (PCA) [88] and the local properties in Isomap [92] and LLE [93].

The commonly-used dimensionality reduction methods include supervised approaches such as Linear Discriminant Analysis (LDA) [89, 90], Canonical Correlation Analysis (CCA) [94] and Partial Least Squares (PLS) [95–98], and unsupervised approaches such as Principal Component Analysis (PCA) [88], Latent Semantic Indexing (LSI) [99], Nonnegative Matrix Factorization (NMF) [100, 101], Latent Dirichlet Allocation [102], Independent Component Analysis (ICA) [94, 103, 104], and additional spectral methods such as Multi-Dimensional Scaling (MDS) [105], Locally Linear Embedding (LLE) [93], Isomap [92], Laplacian Eigenmaps [106], Hessian Eigenmaps [107], Maximum Variance Unfolding (MVU) [108, 109], Lo-

cal Tangent Space Alignment [110], and Kernel methods [62], as well as their extensions [111–117]. Multilayer neural networks can also be used to reduce the data dimensionality [118] . Canonical Correlation Analysis (CCA) [94] and Partial Least Squares (PLS) [95–98] are classical techniques for modeling relations between multiple sets of observed variables. One popular use of CCA and PLS is for multi-label classification, in which one set of variables is derived from the data and the other set is derived from the class labels. In this thesis since we are interested in the label information in the multi-label learning, we will focus on supervised dimensionality reduction algorithms.

Linear and Nonlinear Dimensionality Reduction

Formally, dimensionality reduction studies the following problem: given the d-dimensional variable $\mathbf{x} = [x_1, \ldots, x_d]^T \in \mathbb{R}^d$, the goal is to find a lower-dimensional representation of $\mathbf{x}$, i.e., $\mathbf{s} = [s_1, \ldots, s_p]^T \in \mathbb{R}^p$ where $p \ll d$ is the new dimension, to preserve the information and structure of the original data based on some criterion. In the thesis, we mainly focus on linear dimensionality reduction. Specifically, in linear dimensionality reduction, each new feature is a linear combination of all the input features, i.e.,

$$\mathbf{s} = \mathbf{W}^T \mathbf{x}, \tag{1.7}$$

where $\mathbf{W} \in \mathbb{R}^{d \times p}$ is the so-called projection (transformation) matrix. By optimizing different criteria, we can obtain different $\mathbf{W}$'s.

In this thesis we mainly focus on supervised linear dimensionality reduction, i.e., the dimensionality reduction in the feature space with the aid of the associated label information. Note that the dimensionality reduction in the label space is also considered recently in the literature [53, 59], but we restrict our attention to the dimensionality reduction in the feature space in this thesis. A typical example of supervised linear dimensionality reduction is Linear Discriminate Analysis (LDA) [89, 90], one of the most popular dimensionality reduction algorithms. Specifically, LDA attempts to minimize the within-class variance while maximizing the between-class variance after the linear projection. As a result, after linear projection, the data points belonging to the same class tend to be close while the data points belonging to different classes tend to be far away. Note that in LDA we assume that the classes are mutually exclusive. As a result, LDA cannot be applied in multi-label dimensionality reduction directly.

The nonlinear dimensionality reduction algorithms have attracted significant attention in recent years [87]. These nonlinear algorithms are designed to deal with more complex data, especially the data which intrinsically lies in more complex lower-dimensional spaces, e.g., a lower-dimensional manifold [92]. For example, a set of images produced by the rotation of a face through different angles can be considered as data points lying along a continuous 1-dimensional curve. These nonlinear dimensionality reduction algorithms outperform traditional linear algorithms on artificial data sets, e.g., the Swiss role data set which consists of a set of data points lying on a spiral-like 2-dimensional manifold within a 3-dimensional space. Various nonlinear dimensionality reduction algorithms have been compared with linear algorithms PCA and LDA in [91] empirically. Unfortunately, these nonlinear algorithms do not outperform the linear algorithms on many real-world data sets although they perform very well on selected artificial data sets. Despite the theoretical soundness of these nonlinear algorithms, it is concluded in [91] that they may not capable of outperforming traditional linear algorithms.

Multi-Label Dimensionality Reduction

In many multi-label learning tasks, the high-dimensional data is frequently involved. Similar to many machine learning and data mining tasks, multi-label learning also suffers from the curse of dimensionality. In addition, in many applications of multi-label learning, e.g., text processing [119], data visualization is an important tool to provide insights into the data. Although numerous studies have been devoted to dimensionality reduction, most of them focus on either the unsupervised or the multi-class setting. In addition, many real-world applications can be cast as multi-label classification problems due to its generality. Some well-known examples include scene classification, text categorization, functional genomics, and *Drosophila* gene expression pattern image annotation as discussed in Section 1.2. Multi-label classification has also found applications in many other fields, including music retrieval and annotation [120] and chemical informatics [121]. In comparison with the traditional binary or multi-class classification which assumes mutually exclusive class membership and assigns each sample to a single class, multi-label classification allows different classes to overlap with each other and therefore is more challenging to perform dimensionality reduction.

Next we introduce dimensionality reduction in text processing as an example. Dimensionality reduction is widely used in text categorization and text visualization [119, 122]. In text categorization and visualization, the documents are first transformed into some suitable representation for learning algorithms. A popular representation method is the vector space model [123] where each document is represented as a vector with length $|T|$ where T is the set of terms that occur at least once in one document. In some applications, some words may appear in one document over 100 times but may not appear in any of the other documents. Note that all distinct terms are counted in this model, thus the dimensionality $|T|$ may be prohibitively high and the resulting high-dimensional space using the vector space model is inherently sparse, which are the major challenges for text processing [122]. In this case, it is necessary to reduce the dimensionality via dimensionality reduction.

Unfortunately, existing dimensionality reduction algorithms are less effective for multi-label learning problems. For example, the unsupervised dimensionality reduction algorithms, e.g., PCA, ignore the label information; the supervised dimensionality reduction algorithms, e.g., LDA, typically assume that mutually exclusive labels. One possible solution is to use the problem transformation approaches discussed in Section 1.3 so that the traditional supervised dimensionality reduction algorithms can be applied after transformation. A drawback of this approach is that the label correlations are often ignored in the transformation and meanwhile the computational cost is increased dramatically. For example, some dimensionality reduction algorithms have been proposed for text categorization and visualization in the literature [122], such as term selection methods based on document frequency, information gain, etc. However, the overlapping between labels in the label space is not considered in existing algorithms.

As we discussed in Section 1.3, one fundamental challenge in multi-label learning is how to effectively exploit the label structure for improved classification performance. This is also the case for dimensionality reduction since we expect that after projection the label discriminatory information is preserved in multi-label setting. Also note that dimensionality reduction is a preprocessing step for classification. A natural question is whether we can combine dimensionality reduction algorithms with some classification algorithms together for improved performance. Another challenge is how to build interpretable models for dimensionality reduction.

In traditional linear dimensionality reduction, since the new features are linear combinations of all the original features, it is often difficult to interpret new features. Thus, it is desirable to achieve the dimensionality reduction but also to reduce the number of explicitly used original features. Recently, some sparse feature selection algorithms have been proposed [124–126] via ℓ_1-norm minimization. It is also of great interest to explore the use of ℓ_1-norm minimization in multi-label dimensionality reduction. Finally, the scalability of dimensionality reduction algorithms is also an issue for large-scale problems.

Related Work on Multi-Label Dimensionality Reduction

So far few dimensionality reduction algorithms have been proposed for multi-label learning [12–14, 46, 127]. In this subsection we briefly review these algorithms.

In [12], an algorithm called Multi-label informed Latent Semantic Indexing (MLSI) which preserves the information of data and meanwhile captures the correlations between the multiple labels is proposed. Note that Latent Semantic Indexing (LSI) is a purely unsupervised dimensionality reduction algorithm. In order to incorporate the discriminatory information encoded in the labels, a projection is computed for both the data $\mathbf{X}$ and the corresponding label $\mathbf{Y}$. The difference between MLSI and Canonical Correlation Analysis (CCA) is that in CCA two differen projections are computed simultaneously for $\mathbf{X}$ and $\mathbf{Y}$ respectively while in MLSI the same projection is computed for both $\mathbf{X}$ and $\mathbf{Y}$. As a result, MLSI obtains a new feature space which captures both the information of the original feature space and the label space.

In [13], the sparse kernel orthonormalized partial least squares is proposed to handle multi-label data by imposing sparsity constraints on kernel orthonormalized partial least squares for feature extraction. When partial least squares is applied for supervised learning, one view corresponds to the data $\mathbf{X}$ while the other view corresponds to the associated label $\mathbf{Y}$. It has been shown that orthonormalized PLS is competitive with various PLS variants [13]. The proposed algorithm in [13] is claimed to be applicable for multi-label learning, although its applicability is not discussed in detail and also no empirical result is reported.

In [14] a dimensionality reduction algorithm called Multi-label Dimensionality reduction via Dependence Maximization (MDDM) is proposed. In MDDM, the goal is to find a projection such that the dependence between the feature (data) and the corresponding label is

maximized after projection. In particular, the Hilbert-Schmidt independence criterion [128] is adopted to measure the dependence between the feature description and the corresponding labels due to its simplicity and neat theoretical properties. Furthermore, it is shown in [14] that a closed-form solution can be obtained for the resulting optimization problem in MDDM, which leads to efficient and effective implementations of the proposed algorithm.

In [46], a dimensionality reduction technique called class balanced linear discriminant analysis is proposed. The key idea of classed balanced LDA is to define within-class scatter matrix and between-class scatter matrix for multi-label learning. Since each instance belongs to multiple labels, the traditional definition of within-class and between-class scatter matrices cannot be applied. In fact, the proposed class balanced LDA is equivalent to traditional LDA performed on the data set after applying the copy transformation.

Specifically, in [46] the multi-label within-class scatter matrix $\mathbf{S}_w$ is defined as:

$$\mathbf{S}_w = \sum_{j=1}^{k} \mathbf{S}_w^{(j)}, \text{ where } \mathbf{S}_w^{(j)} = \sum_{i=1}^{n} \mathbf{y}_i(j)(\mathbf{x}_i - \mathbf{m}_j), \tag{1.8}$$

and $\mathbf{m}_j$ is the mean of label j and it is defined as follows:

$$\mathbf{m}_j = \frac{\sum_{i=1}^{n} \mathbf{y}_i(j)\mathbf{x}_i}{\sum_{i=1}^{n} \mathbf{y}_i(j)}. \tag{1.9}$$

In other words, all instances relevant to label j are counted in the computation of the label mean $\mathbf{m}_j$. The multi-label between-class scatter matrix $\mathbf{S}_b$ is defined as:

$$\mathbf{S}_b = \sum_{j=1}^{k} \mathbf{S}_b^{(j)}, \text{ where } \mathbf{S}_b^{(j)} = \left(\sum_{i=1}^{n} \mathbf{y}_i(j) \right) (\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T, \tag{1.10}$$

where $\mathbf{m}$ is the global mean of the data set for multi-label learning, and it is defined as:

$$\mathbf{m} = \frac{\sum_{j=1}^{k} \sum_{i=1}^{n} \mathbf{y}_i(j)\mathbf{x}_i}{\sum_{j=1}^{k} \sum_{i=1}^{n} \mathbf{y}_i(j)}. \tag{1.11}$$

Note that the definition of the global mean $\mathbf{m}$ also considers the multiple counts for instances belonging to multiple labels, thus it is different from the global mean in the traditional multi-class learning as in standard LDA.

Based on the revised definitions of the within-class and between-class scatter matrices for multi-label learning, the transformation matrix $\mathbf{W}$ can be computed readily by following the

standard LDA algorithm. Namely, $\mathbf{W} \in \mathbb{R}^{d \times r}$ consists of the top r eigenvectors of the matrix $\mathbf{S}_w^{\dagger}\mathbf{S}_b$, where r is the reduced dimensionality.

Interestingly, the class balanced LDA can be considered as a special case of our Hypergraph Spectral Learning approach, which will be discussed in detail in Chapter 4.

The LDA is also extended for multi-label learning in [127] by applying the copy transformation.

1.5 Contributions

In this thesis, we study *multi-label dimensionality reduction*. Although dimensionality reduction is well studied in the literature, limited progress has been made on multi-label dimensionality reduction [12–14]. In this thesis, we focus on the following fundamental research questions: (1) How to fully exploit the class label correlation for effective dimensionality reduction in multi-label learning? (2) How to scale dimensionality reduction algorithms to large-scale multi-label problems? (3) How to derive sparse dimensionality reduction algorithms to enhance model interpretability?

Specifically, the thesis can be roughly divided into three parts:

- The design and analysis of dimensionality reduction algorithms for multi-label learning;
- Efficient implementations of dimensionality reduction algorithms, including both existing algorithms and newly proposed ones;
- The application of proposed algorithms on the *Drosophila* gene expression pattern image annotation.

In the remaining of this section, we will discuss each part in more detail.

Design and Analysis of Algorithms for Multi-Label Dimensionality Reduction

Canonical Correlation Analysis (CCA) [129] is a well-known technique for finding the correlations between two sets of multi-dimensional variables. CCA can be applied as a multi-label dimensionality reduction tool in which the two sets of variables are derived from the data and the class labels, respectively. By maximizing the correlation between the data and the associat-

ed label information, the data can be projected onto a lower-dimensional space directed by the label information. In this thesis, we study CCA and reveal some of its interesting properties. In particular, we show that the CCA projection for one set of variables is independent of the regularization on the other set of multi-dimensional variables, providing new insights on the effect of regularization on CCA [130, 131]. The relationship between CCA and Partial Least Squares (PLS) is also investigated in this thesis [130].

In this thesis, we propose Hypergraph Spectral Learning (HSL) to perform dimensionality reduction for multi-label data by exploiting correlations among different labels using a hypergraph [56]. A hypergraph [132–134] is a generalization of the traditional graph in which the edges, called hyperedges, are arbitrary non-empty subsets of the vertex set. In comparison with the traditional graph, the hyperedge can contain more than 2 vertices. As the discrete analog of the Laplace-Beltrami operator on compact Riemannian manifolds [135], the Laplacian matrix can be defined on the hypergraph [132]. Specifically, one can either define hypergraph Laplacian directly using the analogies from traditional 2-graph or expand it into a 2-graph.

In order to model multi-label learning problems using the hypergraph, we construct a hyperedge for each label and include all data points relevant to this label into the hyperedge to capture the correlation among labels. Based on the Laplacian of the constructed hypergraph, we propose the hypergraph spectral learning framework for learning a low-dimensional embedding through a linear transformation by solving an optimization problem. Specifically, the objective function in the optimization problem attempts to preserve the inherent relationship among data points captured by the hypergraph Laplacian. Intuitively, data points that share many common labels tend to be close to each other in the embedded space. HSL is a rather general dimensionality reduction algorithm since different definitions of the hypergraph Laplacian can be used in HSL. For example, it can be shown that CCA is a special case of HSL by defining a specific Laplacian matrix.

Scalable Implementations of Multi-Label Dimensionality Reduction

Both HSL and CCA can be formulated as a generalized eigenvalue problem. In addition, many popular dimensionality reduction algorithms can also be cast as the same type of generalized eigenvalue problem, such as Orthonormalized Partial Least Squares (OPLS) and Linear Dis-

criminant Analysis (LDA). It is well-known that solving large-scale generalized eigenvalue problems is much more challenging than the standard eigenvalue problems [136, 137]. In practical applications, scalability becomes a major concern when the size of data increases.

In this thesis, we propose a direct least squares formulation to efficiently solve a class of dimensionality reduction algorithms, including HSL, CCA, OPLS and LDA as special cases [138, 139]. Specifically, we show that under a mild condition[4], the generalized eigenvalue problem involved in a class of dimensionality reduction algorithms can be formulated as an equivalent least squares problem with a specific target matrix. Based on the equivalent least squares formulation, we extend these dimensionality reduction algorithms by incorporating different regularization terms into the least squares formulation. For example, sparse dimensionality reduction algorithms can be obtained by incorporating ℓ_1-norm penalty. After transforming it into the least squares formulation, we can then apply the iterative conjugate gradient algorithm to solve it efficiently.

One limitation of the direct least squares approach is that the equivalence relationship only holds when the data points are linearly independent, which may not be the case for low-dimensional data. Furthermore, the equivalence relationship between the least squares formulation and the original generalized eigenvalue problem fails when the regularization is considered. We further propose a scalable two-stage approach for the same class of dimensionality reduction algorithms. One appealing feature of the two-stage approach is that it can be applied in the regularization setting without any assumption [140].

To facilitate the use of efficient implementations for the proposed dimensionality reduction algorithms, we have developed a multi-label dimensionality reduction toolbox in Matlab. The whole package is already available online[5]. In this package, the following dimensionality reduction algorithms are included:

- Canonical Correlation Analysis (CCA);
- Partial Least Squares (PLS);

[4]It states that $\{\mathbf{x}_i\}_{i=1}^n$ are linearly independent before centering, i.e., $\text{rank}(\mathbf{X}) = n-1$ after the data is centered (of zero mean).

[5]http://www.public.asu.edu/ lsun27/Code/DMPack.html

- Hypergraph Spectral Learning (HSL);
- Linear Discriminant Analysis (LDA).

For each dimensionality reduction algorithm, we provide four different implementations:

- The implementation which solves the generalized eigenvalue problem directly;
- The direct least squares approach;
- The two-stage approach;
- The implementation based on online algorithms.

Applications of Multi-Label Dimensionality Reduction

We have applied hypergraph spectral learning to annotate the *Drosophila* gene expression patterns images, which are described in Section 1.2. Note that the annotated labels are anatomical and developmental ontology terms from a controlled vocabulary, thus they can be treated as labels and the annotation problem can be modeled as a multi-label learning problem. In the current high-throughput database, annotation terms are assigned to groups of patterns rather than to individual images. We propose to extract invariant features from images, and construct pyramid match kernels to measure the similarity between sets of patterns. To exploit the complementary information conveyed by different features and incorporate the correlation among patterns sharing common structures, we propose efficient convex formulations to integrate the kernels derived from various features. In particular, the hypergraph spectral learning is applied to predict annotated terms. The proposed framework is evaluated by comparing its annotation with that of human curators, and promising performance in terms of F1 score has been reported. We will discuss more details on this application in Chapter 6.

1.6 Notations

Throughout the thesis, all matrices are boldface uppercase, and vectors are boldface lowercase. n is the number of samples in the training data set, d is the data dimensionality, and k is the number of classes (or labels). The ith sample in the training data set is denoted as $\mathbf{x}_i \in \mathbb{R}^d$, and its corresponding label is denoted as $\mathbf{y}_i \in \mathbb{R}^k$, where $\mathbf{y}_i(j) = 1$ if $\mathbf{x}_i$ belongs to class j and

$\mathbf{y}_i(j) = 0$ otherwise. The label set is denoted as $\mathscr{L} = \{C_1, C_2, \ldots, C_k\}$. $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ represents the data matrix, and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n] \in \mathbb{R}^{k \times n}$ is the matrix representation for the label information. The training data set with n multi-label examples is denoted as $\mathscr{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$. $\mathbf{I}_p$ is the p-by-p identity matrix, and $1_p \in \mathbb{R}^p$ is the vector of all ones. Note that the subscript p may be omitted when the size is clear from the context.

1.7 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter 2 discusses Partial Least Squares (PLS) and its applications in classification.
- Chapter 3 provides in-depth study of Canonical Correlation Analysis (CCA). Its relationship between PLS is also discussed.
- Chapter 4 introduces Hypergraph Spectral Learning (HSL) for multi-label dimensionality reduction. The equivalent least squares formulation for a class of dimensionality reduction algorithms, including HSL, CCA, Orthonormalized PLS (OPLS) and Linear Discriminant Analysis (LDA), is also discussed. An efficient algorithm based on the equivalent least squares formulation is also presented in this chapter.
- Chapter 5 introduces an efficient two-stage approach for the same class of dimensionality reduction algorithms as in Chapter 4.
- Chapter 6 introduces the application of multi-label dimensionality reduction algorithms in *Drosophila* gene expression pattern image annotation.
- Chapter 7 summarizes the contributions, concludes the thesis and discusses some future work.

Chapter 2

PARTIAL LEAST SQUARES

2.1 Introduction

Partial Least Squares (PLS) originates from Herman Wold's nonlinear iterative partial least squares algorithm [95]. Similar to canonical correlation analysis, it is a family of methods for modeling the relationships between two sets of variables [98, 141, 142]. One of the appealing features of PLS is that it can analyze data with much higher dimensionality than the sample size and with massive collinearity among the variables [143, 144]. PLS is also resistant to overfitting, and it is fast and easy to implement. In comparison to Principal Component Analysis (PCA), which extracts the variance of the input data, PLS extracts latent features from data by maximizing the covariance between the two blocks of variables.

The basic assumption in PLS is that the high multi-collinearity exists among the variables, and this can be deduced by dimensionality reduction via latent variables. It is a popular tool for regression, classification, and dimensionality reduction [97, 145, 146], especially in the field of chemometrics. PLS can be applied to classification problems by encoding the class membership in an appropriate indicator matrix. There is a close relationship between PLS and linear discriminant analysis [145]. It can also be applied as a dimensionality reduction tool. After relevant latent vectors are extracted, an appropriate classifier, such as Support Vector Machines (SVM) [62] can be applied for classification [147]. PLS can also be extended to regression problems by treating each of the predictor and response variables as a block of variables. Furthermore, it has been shown that the PLS regression yields a shrinkage estimator [146].

PLS computes orthogonal score vectors (also called latent vectors) by maximizing the covariance between different sets of variables. This is commonly done through an iterative procedure or solving an eigenvalue problem. Given the latent variables, the data sets are then transformed in a process where information contained in the latent variables is subtracted. This process, often referred to as deflation, can be done in various ways, resulting in many variants of PLS, such as PLS Mode A [148], PLS2 [149], Orthonormalized PLS (OPLS) [150], PLS-SB [151], SIMPLS [152]. All these variants will be discussed in detail in Section 2.3.

PLS has recently gained a lot of attention in the analysis of high dimensional data in many fields [153–156], such as bioinformatics [157–161]. PLS has been extended to the kernel-induced feature space. This leads to kernel PLS, which has been applied in many domains [147, 156, 162–165]. In addition, sparse PLS has been developed [156–158, 162], since sparsity often leads to easy interpretation and a good generalization ability [79]. In this chapter, we will focus on the basic concepts of PLS, including many popular variants of PLS. Its applications in regression and classification will also be discussed.

The rest of this chapter is organized as follows. We introduce the basics of PLS in Section 2.2. Several PLS variants including PLS Mode A, PLS2, PLS1, PLS-SB, SIMPLS and Orthonormalized PLS (OPLS) are introduced in Section 2.3. The OPLS and its relationship with other variants are also discussed in Section 2.3. The PLS regression is discussed in Section 2.4, in which the shrinkage properties of PLS regression are studied in comparison with Ordinary Least Squares (OLS), Principal Component Regression (PCR) and ridge regression. The PLS classification is discussed in Section 2.5, where we study the connections between PLS and linear discriminant analysis [166]. In Section 2.6, the relationship between PLS and CCA is discussed.

2.2 Basic Models of Partial Least Squares

Similar to CCA, PLS models the relationship between two data sets, or two blocks of variables. We denote the two data sets as $\mathbf{X} \in \mathbb{R}^{d\times n}$ and $\mathbf{Y} \in \mathbb{R}^{k\times n}$, where n is the number of samples, and d and k are the dimensionality of $\mathbf{X}$ and $\mathbf{Y}$, respectively. Without loss of generality, we assume that both $\mathbf{X}$ and $\mathbf{Y}$ are centered, i.e., $\mathbf{X}1 = 0$, and $\mathbf{Y}1 = 0$, where 1 is a vector of all ones, and 0 is a vector of all zeros. In the general model of PLS, we assume that $\mathbf{X}$ and $\mathbf{Y}$ can be decomposed into the following form:

$$\mathbf{X} = \mathbf{P}\mathbf{T}^T + \mathbf{E} \tag{2.1}$$

$$\mathbf{Y} = \mathbf{Q}\mathbf{U}^T + \mathbf{F}, \tag{2.2}$$

where $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_p] \in \mathbb{R}^{n\times p}$ and $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_p] \in \mathbb{R}^{n\times p}$ denote the score vectors, or latent vectors of $\mathbf{X}$ and $\mathbf{Y}$, respectively, $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_p] \in \mathbb{R}^{d\times p}$ and $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_p] \in \mathbb{R}^{k\times p}$ are called loadings for $\mathbf{X}$ and $\mathbf{Y}$, respectively, $\mathbf{E} \in \mathbb{R}^{d\times n}$ and $\mathbf{F} \in \mathbb{R}^{k\times n}$ are called residuals. The assumption in PLS is that the latent variables $\mathbf{T}$ and $\mathbf{U}$ capture the underlying information

Algorithm 1 The NIPALS Algorithm

Input: $\mathbf{X}$, $\mathbf{Y}$
Output: $\mathbf{t}$, $\mathbf{u}$, $\mathbf{p}$, $\mathbf{q}$
Initialize $\mathbf{u}$
repeat
  $\mathbf{w} = \mathbf{X}\mathbf{u}$
  $\mathbf{w} = \frac{\mathbf{w}}{\|\mathbf{w}\|_2}$
  $\mathbf{t} = \mathbf{X}^T\mathbf{w}$
  $\mathbf{c} = \mathbf{Y}\mathbf{t}$
  $\mathbf{c} = \frac{\mathbf{c}}{\|\mathbf{c}\|_2}$
  $\mathbf{u} = \mathbf{Y}^T\mathbf{c}$
until convergence
$\mathbf{p} = \frac{\mathbf{X}\mathbf{t}}{\mathbf{t}^T\mathbf{t}}$, $\mathbf{q} = \frac{\mathbf{Y}\mathbf{u}}{\mathbf{u}^T\mathbf{u}}$

of the original blocks $\mathbf{X}$ and $\mathbf{Y}$. The difference between PLS and PCA is that the latent scores $\mathbf{T}$ and $\mathbf{U}$ are extracted by considering the two blocks $\mathbf{X}$ and $\mathbf{Y}$ simultaneously in PLS whereas only one block is considered in PCA.

In the past several decades, various PLS variants have been proposed [96, 97]. In the following, we discuss the classical formulation, known as the Nonlinear Iterative Partial Least Squares (NIPALS) [148].

The NIPALS algorithm

The Nonlinear Iterative Partial Least Squares (NIPALS) algorithm [148] is the classical PLS formulation, and it is considered as the basis of other PLS variants. The NIPALS algorithm is described in Algorithm 1. Note that in this framework, only one pair of $(\mathbf{t}, \mathbf{p})$ and $(\mathbf{u}, \mathbf{q})$ are computed. In order to compute a sequence of the pairs, the deflation scheme can be applied to update $\mathbf{X}$ and $\mathbf{Y}$ so that different pairs can be obtained by applying the NIPALS algorithm repeatedly.

The key of the NIPALS algorithm is that $\mathbf{w}$ and $\mathbf{c}$ are updated by making use of both blocks $\mathbf{X}$ and $\mathbf{Y}$. In fact, NIPALS tries to maximize the covariance between $\mathbf{t}$ and $\mathbf{u}$ as

$$\max_{\mathbf{t},\mathbf{u}} \mathrm{cov}(\mathbf{t}, \mathbf{u}) = \max_{\mathbf{w},\mathbf{c}} \mathrm{cov}(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}) = \max_{\mathbf{w},\mathbf{c}} \mathbf{w}^T\mathbf{X}\mathbf{Y}^T\mathbf{c}.$$

Thus, in the NIPALS algorithm, we update $\mathbf{c}$ by setting it as $\mathbf{c} = \mathbf{Y}\mathbf{X}^T\mathbf{w}$ and update $\mathbf{w}$ as $\mathbf{w} = \mathbf{X}\mathbf{Y}^T\mathbf{c}$. Based on $\mathbf{w}$ and $\mathbf{c}$, the score vectors $\mathbf{t}$ and $\mathbf{u}$ can be updated accordingly.

One important consequence of the NIPALS algorithm is that the solution of the NIPALS algorithm can be computed directly by solving some eigenvalue problems. It follows from the procedure of NIPALS in Algorithm 1 that

$$\mathbf{w} \propto \mathbf{X}\mathbf{u} \propto \mathbf{X}\mathbf{Y}^T\mathbf{c} \propto \mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{t} \propto \mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T\mathbf{w}.$$

Thus, $\mathbf{w}$ corresponds to the eigenvector of the following eigenvalue problem:

$$\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{w}. \tag{2.3}$$

Note that $\mathbf{t} = \mathbf{X}^T\mathbf{w}$, which implies that $\mathbf{t}$ corresponds to the eigenvector of the following eigenvalue problem:

$$\mathbf{X}^T\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{t} = \lambda\mathbf{t}. \tag{2.4}$$

Similarly, we can show that $\mathbf{c}$ corresponds to the eigenvector of the following eigenvalue problem:

$$\mathbf{Y}\mathbf{X}^T\mathbf{X}\mathbf{Y}^T\mathbf{c} = \lambda\mathbf{c}, \tag{2.5}$$

and $\mathbf{u}$ corresponds to the eigenvector of the following eigenvalue problem:

$$\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T\mathbf{X}\mathbf{u} = \lambda\mathbf{u}. \tag{2.6}$$

2.3 Partial Least Squares Variants

PLS is an iterative process. After the first pair of score vectors $(\mathbf{t},\mathbf{u})$ is obtained, the next step is to compute the next pair of score vectors. Various deflation schemes have been proposed to compute the score vector pairs sequentially [96–98]. Different deflation schemes define different variants of PLS, including PLS Mode A, PLS2, PLS-SB, SIMPLS, and orthonormalized PLS, which are briefly reviewed below.

PLS Mode A

The key of PLS Mode A is to deflate $\mathbf{X}$ and $\mathbf{Y}$ by subtracting the rank-one approximation using the corresponding score and loading vectors. Specifically, the loading vectors $\mathbf{p}$ and $\mathbf{q}$, which form the columns of $\mathbf{P}$ and $\mathbf{Q}$, respectively, are computed as the coefficients of regressing $\mathbf{X}$ on $\mathbf{t}$ and $\mathbf{Y}$ on $\mathbf{u}$, respectively:

$$\begin{aligned}\mathbf{p} &= \mathbf{X}\mathbf{t}/(\mathbf{t}^T\mathbf{t}),\\ \mathbf{q} &= \mathbf{Y}\mathbf{u}/(\mathbf{u}^T\mathbf{u}).\end{aligned}$$

Algorithm 2 PLS Mode A

Input: $\mathbf{X}$, $\mathbf{Y}$, p.
Output: $\mathbf{T}$, $\mathbf{U}$, $\mathbf{P}$, $\mathbf{Q}$.
Initialize $\mathbf{T}$, $\mathbf{U}$, $\mathbf{P}$, and $\mathbf{Q}$:

$$\mathbf{T} = [], \mathbf{U} = [], \mathbf{P} = [], \mathbf{Q} = []$$

for $i = 1$ to p **do**
Compute the first pair of singular vectors of $\mathbf{X}\mathbf{Y}^T$ corresponding to the largest singular value: $(\mathbf{w}, \mathbf{c})$.
Compute the score vectors as:

$$\mathbf{t} = \mathbf{X}^T\mathbf{w}, \mathbf{u} = \mathbf{Y}^T\mathbf{c}.$$

Regress $\mathbf{X}$ on $\mathbf{t}$ and $\mathbf{Y}$ on $\mathbf{u}$:

$$\mathbf{p} = \frac{\mathbf{X}\mathbf{t}}{\mathbf{t}^T\mathbf{t}}, \mathbf{q} = \frac{\mathbf{Y}\mathbf{u}}{\mathbf{u}^T\mathbf{u}}.$$

Subtract the rank-one approximation to deflate $\mathbf{X}$ and $\mathbf{Y}$:

$$\mathbf{X} \leftarrow \mathbf{X} - \mathbf{p}\mathbf{t}^T, \mathbf{Y} \leftarrow \mathbf{Y} - \mathbf{q}\mathbf{u}^T.$$

Update $\mathbf{T}$, $\mathbf{U}$, $\mathbf{P}$, and $\mathbf{Q}$ as:

$$\mathbf{T} \leftarrow [\mathbf{T}, \mathbf{t}], \mathbf{U} \leftarrow [\mathbf{U}, \mathbf{u}], \mathbf{P} \leftarrow [\mathbf{P}, \mathbf{p}], \mathbf{Q} \leftarrow [\mathbf{Q}, \mathbf{q}].$$

if $\mathbf{X} == 0$ or $\mathbf{Y} == 0$ **then**
break;
end if
end for

At each iteration of PLS Mode A, we update $\mathbf{X}$ and $\mathbf{Y}$ as follows:

$$\begin{aligned} \mathbf{X} &\leftarrow \mathbf{X} - \mathbf{p}\mathbf{t}^T, \\ \mathbf{Y} &\leftarrow \mathbf{Y} - \mathbf{q}\mathbf{u}^T. \end{aligned}$$

The full procedure of PLS Mode A is described in Algorithm 2, where p is the number of extracted pairs of score vectors.

An important property of PLS Mode A is that the extracted score vectors $\{\mathbf{t}_i\}_{i=1}^p$ and $\{\mathbf{u}_i\}_{i=1}^p$ are mutually orthogonal, which is summarized in the following theorem:

Theorem 2.1 *For $i \neq j$, the score vectors obtained at the ith and the jth steps of PLS Mode A are mutually orthogonal, that is,*

$$\boldsymbol{t}_i^T\boldsymbol{t}_j = 0, \ \mathbf{u}_i^T\mathbf{u}_j = 0.$$

Proof We use the superscript i to denote the variables at the ith iteration. For example, the data matrices $\mathbf{X}$ and $\mathbf{Y}$ at the ith iteration are denoted as $\mathbf{X}^{(i)}$ and $\mathbf{Y}^{(i)}$, respectively. It follows from Algorithm 2 that

$$\begin{aligned}
\mathbf{X}^{(i+1)} &= \mathbf{X}^{(i)} - \mathbf{p}^{(i)}\mathbf{t}^{(i)^T} \\
&= \mathbf{X}^{(i)} - \mathbf{X}^{(i)}\frac{\mathbf{t}^{(i)}\mathbf{t}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}} \\
&= \mathbf{X}^{(i)} - \mathbf{X}^{(i)}\frac{\mathbf{t}^{(i)}\mathbf{w}^{(i)^T}\mathbf{X}^{(i)}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}} \\
&= \left(\mathbf{I} - \mathbf{X}^{(i)}\frac{\mathbf{t}^{(i)}\mathbf{w}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}}\right)\mathbf{X}^{(i)} \\
&= \mathbf{B}^{(i)}\mathbf{X}^{(i)},
\end{aligned}$$

where $\mathbf{B}^{(i)} = \mathbf{I} - \mathbf{X}^{(i)}\frac{\mathbf{t}^{(i)}\mathbf{w}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}}$. Without loss of generality, we assume that $i < j$. Thus, we have

$$\begin{aligned}
\mathbf{X}^{(j)} &= \mathbf{B}^{(j-1)}\mathbf{X}^{(j-1)} \\
&= \mathbf{B}^{(j-1)}\mathbf{B}^{(j-2)}\mathbf{X}^{(j-2)} \\
&= \cdots \\
&= \mathbf{B}^{(j-1)}\cdots\mathbf{B}^{(i+1)}\mathbf{X}^{(i+1)} \\
&= \mathbf{Z}\mathbf{X}^{(i+1)},
\end{aligned}$$

where $\mathbf{Z} = \mathbf{B}^{(j-1)}\cdots\mathbf{B}^{(i+1)}$. Then the inner product between $\mathbf{t}^{(i)}$ and $\mathbf{t}^{(j)}$ for $i \neq j$ can be computed as follows:

$$\begin{aligned}
\mathbf{t}^{(i)^T}\mathbf{t}^{(j)} &= \mathbf{t}^{(i)^T}\mathbf{X}^{(j)^T}\mathbf{w}^{(j)} \\
&= \mathbf{t}^{(i)^T}\mathbf{X}^{(i+1)^T}\mathbf{Z}^T\mathbf{w}^{(j)} \\
&= \mathbf{t}^{(i)^T}\left(\mathbf{X}^{(i)} - \mathbf{p}^{(i)}\mathbf{t}^{(i)^T}\right)^T\mathbf{Z}^T\mathbf{w}^{(j)} \\
&= \mathbf{t}^{(i)^T}\left(\mathbf{X}^{(i)} - \mathbf{X}^{(i)}\frac{\mathbf{t}^{(i)}\mathbf{t}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}}\right)^T\mathbf{Z}^T\mathbf{w}^{(j)} \\
&= \mathbf{t}^{(i)^T}\left(\mathbf{I} - \frac{\mathbf{t}^{(i)}\mathbf{t}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}}\right)\mathbf{X}^{(i)^T}\mathbf{Z}^T\mathbf{w}^{(j)} \\
&= \left(\mathbf{t}^{(i)^T} - \mathbf{t}^{(i)^T}\frac{\mathbf{t}^{(i)}\mathbf{t}^{(i)^T}}{\mathbf{t}^{(i)^T}\mathbf{t}^{(i)}}\right)\mathbf{X}^{(i)^T}\mathbf{Z}^T\mathbf{w}^{(j)} \\
&= 0.
\end{aligned}$$

Similarly, we can show that $\mathbf{u}^{(i)^T}\mathbf{u}^{(j)} = 0$ for $i \neq j$. ∎

PLS2

PLS2 is one of the most popular PLS formulations for regression [149], especially in chemometrics [96]. Unlike the PLS Mode A, the relationship between $\mathbf{X}$ and $\mathbf{Y}$ in PLS2 is asymmetric, and the extracted latent vectors of $\mathbf{X}$ are assumed to be good predictors of $\mathbf{Y}$. In particular, a linear relationship between the score vectors $\mathbf{T}$ and $\mathbf{U}$ is assumed as

$$\mathbf{U} = \mathbf{TD} + \mathbf{H}, \tag{2.7}$$

where $\mathbf{D}$ is a $p \times p$ diagonal matrix and $\mathbf{H}$ denotes the matrix of residuals. As a result, we can deflate $\mathbf{Y}$ using $\mathbf{t}$ directly instead of using $\mathbf{u}$. Specifically, at each iteration, the following deflation scheme is applied on $\mathbf{X}$ and $\mathbf{Y}$:

$$\begin{aligned} \mathbf{X} &\leftarrow \mathbf{X} - \mathbf{p}\mathbf{t}^T, \\ \mathbf{Y} &\leftarrow \mathbf{Y} - \frac{\mathbf{Y}\mathbf{t}}{\mathbf{t}^T\mathbf{t}}\mathbf{t}^T. \end{aligned}$$

Similar to PLS Mode A, this deflation scheme guarantees the mutual orthogonality of the extracted score vectors $\{\mathbf{t}_i\}_{i=1}^{k}$.

PLS1

PLS1 [167–169] is a special case of PLS2 in which one of the blocks contains only a single variable. Without loss of generality, we assume that $\mathbf{Y} \in \mathbb{R}^{1\times n}$, i.e., $k = 1$. In fact, PLS1 can be considered as a regularization technique similar to ridge regression and principal component regression [96], which will be discussed in detail in Section 2.4. In particular, the deflation of $\mathbf{Y}$ is not necessary, since $\mathbf{Y}$ contains only one column [149]. In PLS1, the mutual orthogonality of $\{\mathbf{t}_i\}_{i=1}^{p}$ is preserved, and PLS1 essentially finds a direction in the space of regression coefficients that are orthogonal to all previous coefficients.

PLS-SB

PLS-SB was introduced in behavioral teratology by Sampson *e*t al. [151] (S denotes Sampson and B denotes Bookstein who proposed this PLS variant). The overall procedure of PLS-SB is similar to that of PLS Mode A. The major difference between PLS-SB and PLS Mode A is that in PLS-SB, $\mathbf{X}$ and $\mathbf{Y}$ are not updated at each iteration. Instead, we update the covariance matrix $\mathbf{X}\mathbf{Y}^T$ at each iteration. Unlike PLS Mode A, all singular vectors of the covariance matrix

$\mathbf{XY}^T$ are computed at once in PLS-SB, whereas only the singular vectors corresponding to the largest singular value are computed at each iteration in PLS Mode A. In contrast to PLS2 and PLS Mode A, the extracted score vectors $\{\mathbf{t}_i\}_{i=1}^p$ are in general not mutually orthogonal, since $\mathbf{t}_i = \mathbf{X}\mathbf{w}_i$, where $\{\mathbf{w}_i\}_{i=1}^p$ are left singular vectors of $\mathbf{XY}^T$, and are mutually orthogonal. Similarly, the orthogonality property does not hold for the score vectors $\{\mathbf{u}\}_{i=1}^p$ for the other block $\mathbf{Y}$.

SIMPLS

SIMPLS [152] was proposed to avoid the deflation steps at each iteration of PLS2 by directly computing the weight vectors $\{\mathbf{w}_i\}_{i=1}^p$ such that $\mathbf{t} = \mathbf{X}^T\mathbf{w}$. Since no deflation is performed, $\mathbf{w}$ is applied to the original matrix $\mathbf{X}$ before deflation directly. At each iteration of SIMPLS, the SVD of the sample covariance $\mathbf{XY}^T$ is computed under some constraint that the mutual orthogonality of score vectors $\{\mathbf{t}_i\}_{i=1}^p$ is preserved. It has been shown that SIMPLS is equivalent to PLS1 but differs from PLS2 when applied to the multi-dimensional matrix $\mathbf{Y}$ [152].

Orthonormalized PLS

Recall that the objective function in the NIPALS algorithm can be expressed as

$$\operatorname{cov}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2 = \operatorname{var}\left(\mathbf{X}^T\mathbf{w}\right) \operatorname{corr}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2 \operatorname{var}\left(\mathbf{Y}^T\mathbf{c}\right). \tag{2.8}$$

This can be considered as a penalized version of CCA in which only the correlation term is maximized. If the penalty term $\operatorname{var}\left(\mathbf{X}^T\mathbf{w}\right)$ is removed, we obtain the following optimization problem for orthonormalized PLS [150]:

$$\begin{aligned}(\mathbf{w}^*, \mathbf{c}^*) &= \underset{\|\mathbf{w}\|=\|\mathbf{c}\|=1}{\arg\max} \operatorname{var}\left(\mathbf{Y}^T\mathbf{c}\right) \operatorname{corr}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2 \\ &= \underset{\|\mathbf{w}\|=\|\mathbf{c}\|=1}{\arg\max} \frac{\operatorname{cov}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2}{\operatorname{var}\left(\mathbf{X}^T\mathbf{w}\right)}.\end{aligned} \tag{2.9}$$

Since the objective function is invariant to the scaling of $\mathbf{w}$ and $\mathbf{c}$, OPLS can be formulated equivalently as the following optimization problem:

$$\begin{aligned}\max_{\mathbf{w},\mathbf{c}} \quad & \mathbf{w}^T\mathbf{XY}^T\mathbf{c} \\ \text{s.t.} \quad & \mathbf{w}^T\mathbf{XX}^T\mathbf{w} = 1 \\ & \mathbf{c}^T\mathbf{c} = 1.\end{aligned} \tag{2.10}$$

Based on this constrained optimization problem, we can construct the Lagrangian function

$$L(\mathbf{w}, \lambda_x, \lambda_y) = \mathbf{w}^T \mathbf{X}\mathbf{Y}^T \mathbf{c} - \frac{\lambda_x}{2}\left(\mathbf{w}^T \mathbf{X}\mathbf{X}^T \mathbf{w} - 1\right) - \frac{\lambda_y}{2}\left(\mathbf{c}^T \mathbf{c} - 1\right),$$

where λ_x and λ_y are Lagrange multipliers. Taking the derivatives of L with respect to λ_x and λ_y, and setting them to zero, we obtain the following equations:

$$\mathbf{X}\mathbf{Y}^T \mathbf{c} - \lambda_x \mathbf{X}\mathbf{X}^T \mathbf{w} = 0 \tag{2.11}$$

$$\mathbf{Y}\mathbf{X}^T \mathbf{w} - \lambda_y \mathbf{c} = 0. \tag{2.12}$$

Hence, we have

$$\begin{aligned}
\lambda_x &= \lambda_x \left(\mathbf{w}^T \mathbf{X}\mathbf{X}^T \mathbf{w}\right) \\
&= \mathbf{w}^T \left(\lambda_x \mathbf{X}\mathbf{X}^T \mathbf{w}\right) \\
&= \mathbf{w}^T \mathbf{X}\mathbf{Y}^T \mathbf{c} \\
&= \mathbf{c}^T \mathbf{Y}\mathbf{X}^T \mathbf{w} \\
&= \mathbf{c}^T (\lambda_y \mathbf{c}) \\
&= \lambda_y,
\end{aligned}$$

where we have made use of Eqs. (2.11) and (2.12). In the following discussion, we use λ to denote both λ_x and λ_y as they are identical.

In the supervised learning, $\mathbf{X}$ typically corresponds to the data while $\mathbf{Y}$ corresponds to the label. As a result, the weight vectors for $\mathbf{X}$ is of more interest. Next we derive an optimization problem involving only the weight vector $\mathbf{w}$. It follows from Eq. (2.12) that

$$\mathbf{c} = \frac{1}{\lambda}\mathbf{Y}\mathbf{X}^T \mathbf{w}. \tag{2.13}$$

Substituting this into Eq. (2.11), we obtain the following generalized eigenvalue problem:

$$\mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}^T \mathbf{w} = \lambda^2 \mathbf{X}\mathbf{X}^T \mathbf{w}. \tag{2.14}$$

We can substitute Eq. (2.13) into the original problem in Eq. (2.10), resulting in the following optimization problem:

$$\begin{aligned}
\max_{\mathbf{w}} \quad & \mathbf{w}^T \mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}^T \mathbf{w} \\
\text{s.t.} \quad & \mathbf{w}^T \mathbf{X}\mathbf{X}^T \mathbf{w} = 1.
\end{aligned} \tag{2.15}$$

Multiple weight vectors $\{\mathbf{w}_i\}_{i=1}^{p}$ for $\mathbf{X}$ in OPLS can be computed simultaneously by solving the following optimization problem [13]:

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{Tr}\left(\mathbf{W}^T\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T\mathbf{W}\right) \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W} = \mathbf{I}_p, \end{aligned} \tag{2.16}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_p] \in \mathbb{R}^{d\times p}$ is the weight matrix and $\mathbf{I}_p \in \mathbb{R}^{p\times p}$ is the identity matrix.

OPLS can be used for multi-regression problems when $\mathbf{X}$ contains the input data and $\mathbf{Y}$ contains the response variables. Similar to other PLS variants discussed above, OPLS can also be used for dimensionality reduction in supervised classification problems when $\mathbf{Y}$ encodes the class membership information. It has been shown to be competitive with other PLS variants [13].

Relationship between OPLS and Other PLS Models

OPLS is closely related to several other PLS variants discussed in Section 2.3. It has been shown that PLS2 maximizes the same objective function as SIMPLS, but with different (and less intuitive) constraints [170]. Denote

$$\mathbf{W}^{(i-1)} = [\mathbf{w}_1, \ldots, \mathbf{w}_{i-1}]. \tag{2.17}$$

Then the ith weight vector $\mathbf{w}_i$ in PLS2 and SIMPLS can be computed by solving the following optimization problem:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \mathbf{w}^T\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T\mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T\mathbf{X}\mathbf{X}^T\mathbf{w}_j = 0, \text{ for } j = 1, \ldots, i-1, \\ & \mathbf{w}^T\mathbf{L}\mathbf{w} = 1, \end{aligned} \tag{2.18}$$

for some matrix $\mathbf{L}$. The difference between PLS2 and SIMPLS lies in the use of different matrix $\mathbf{L}$. In particular,

$$\mathbf{L} = \mathbf{I}_d - \mathbf{W}^{(i-1)}\left(\mathbf{W}^{(i-1)}\right)^{\dagger} \text{ and } \mathbf{L} = \mathbf{I}_d,$$

are used in PLS2 and SIMPLS, respectively, where $(\mathbf{W}^{(i)})^{\dagger}$ denotes the pseudoinverse of $\mathbf{W}^{(i)}$. By setting $\mathbf{L} = \mathbf{X}\mathbf{X}^T$, we obtain a PLS formulation identical to the one in Eq. (2.16) for OPLS.

We show in the following theorem that by setting $\mathbf{L} = \mathbf{X}\mathbf{X}^T$, a PLS formulation identical to the one in Eq. (2.16) can be obtained.

Theorem 2.2 *Let $\boldsymbol{W}^{(i-1)}$, $\boldsymbol{X}$, and $\boldsymbol{Y}$ be defined as above, and let $\boldsymbol{w}_i$ solves the optimization problem in Eq. (2.18) with $\mathbf{L} = \boldsymbol{X}\boldsymbol{X}^T$. Then $\boldsymbol{W}^{(p)} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_p]$ solves the optimization problem in Eq. (2.16).*

Proof Define the Lagrangian function for the optimization problem in Eq. (2.18) as:

$$\begin{aligned} L(\mathbf{w}, \lambda, \gamma) &= \mathbf{w}^T \mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}^T \mathbf{w} - \sum_{j=1}^{i-1} \lambda_j \mathbf{w}^T \mathbf{X}\mathbf{X}^T \mathbf{w}_j \\ &\quad -\gamma(\mathbf{w}^T \mathbf{L}\mathbf{w} - 1). \end{aligned} \tag{2.19}$$

Taking the derivative of L with respect to $\mathbf{w}$ and setting it to zero, we obtain

$$2\mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}^T \mathbf{w} - \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)} \Lambda - 2\gamma \mathbf{L}\mathbf{w} = 0, \tag{2.20}$$

where $\Lambda = [\lambda_1, \ldots, \lambda_i]^T$. Multiplying Eq. (2.20) by ${\mathbf{W}^{(i-1)}}^T$ on the left side, we have

$$\Lambda = 2({\mathbf{W}^{(i-1)}}^T \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)})^{-1} \left({\mathbf{W}^{(i-1)}}^T \mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}^T - \gamma {\mathbf{W}^{(i-1)}}^T \mathbf{L} \right) \mathbf{w}. \tag{2.21}$$

It follows that

$$\begin{aligned} &\left(\mathbf{I} - \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)} ({\mathbf{W}^{(i-1)}}^T \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)})^{-1} {\mathbf{W}^{(i-1)}}^T \right) \mathbf{X}\mathbf{Y}^T \mathbf{Y}\mathbf{X}\mathbf{w} \\ = \;& \gamma(\mathbf{I} - \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)} ({\mathbf{W}^{(i-1)}}^T \mathbf{X}\mathbf{X}^T \mathbf{W}^{(i-1)})^{-1} {\mathbf{W}^{(i-1)}}^T) \mathbf{L}\mathbf{w}. \end{aligned} \tag{2.22}$$

This completes the proof of the theorem. ∎

It follows from the above discussion that OPLS optimizes the same objective function as PLS2 and SIMPLS, and the orthogonality property, i.e.,

$$\mathbf{w}_i^T \mathbf{X}\mathbf{X}^T \mathbf{w}_j = 0, \text{ for } i \neq j,$$

is enforced in all of the three PLS formulations. The difference lies in how the length, denoted as

$$\|\mathbf{w}\|_{\mathbf{L}} = \sqrt{\mathbf{w}^T \mathbf{L}\mathbf{w}},$$

of each weight vector is measured based on the different choices of $\mathbf{L}$.

2.4 Partial Least Squares Regression

PLS has been used as a popular tool for regression in many fields, especially in chemometrics [171]. In this section, we discuss the variant PLS1 in the regression setting, where the response $\mathbf{Y} \in \mathbb{R}^{1\times n}$ consists of a single variable and the deflation is not performed on $\mathbf{Y}$. Similar to Principal Component Regression (PCR) and ridge regression, PLS regression yields a shrinkage estimator. PLS regression extracts components from the data $\mathbf{X}$ that have a high covariance with the response $\mathbf{Y}$. Unlike the linear estimators produced by Ordinary Least Squares (OLS), PCR, and ridge regression, the estimator computed by PLS regression depends on the response $\mathbf{Y}$ nonlinearly. As a result, it is not straightforward to obtain the shrinkage factors for PLS. In this section, we give a detailed overview of the shrinkage properties of PLS regression. In particular, we show how to derive the shrinkage factors for PLS regression. The effects of shrinkage factors for the mean squared error of estimators produced by different regression models are also discussed.

Basics of PLS Regression

In PLS regression, the latent variables of $\mathbf{X}$ are assumed to be predictive of $\mathbf{Y}$. In addition, we assume that there is a linear relationship between the latent scores $\mathbf{t}$ and $\mathbf{u}$. This linear relationship leads to a deflation scheme for both $\mathbf{X}$ and $\mathbf{Y}$. Specifically, we assume that

$$\mathbf{U} = \mathbf{TD} + \mathbf{H}. \tag{2.23}$$

Substituting Eq. (2.23) into Eq. (2.2), we obtain that

$$\mathbf{Y} = \mathbf{QD}^T\mathbf{T}^T + \left(\mathbf{QH}^T + \mathbf{F}\right). \tag{2.24}$$

As a result, we can assume a linear relationship between $\mathbf{Y}$ and $\mathbf{T}$, and $\mathbf{QH}^T + \mathbf{F}$ can be considered as the residual. In the following discussion, we primarily focus on the PLS1 regression. As we discuss in Section 2.3, PLS1 [167–169] is a special case of PLS2 when $\mathbf{Y} \in \mathbb{R}^{1\times n}$ contains a single variable. Thus, we do not perform deflation for $\mathbf{Y}$; otherwise the algorithm will terminate in one step. We give the detailed procedure of PLS1 for regression in Algorithm 3. Note that in Algorithm 3, p is the number of components extracted by PLS1. At the ith iteration, we denote the weight vectors and latent vectors as

$$\mathbf{T}^{(i)} = \left[\mathbf{t}^{(1)}, \ldots, \mathbf{t}^{(i)}\right], \ \mathbf{W}^{(i)} = \left[\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(i)}\right]. \tag{2.25}$$

Algorithm 3 The PLS1 Algorithm for Regression

Input: $\mathbf{X}$, $\mathbf{Y}$, p
Output: β
$\mathbf{T} = []$
for $i = 0$ to p **do**
  $\mathbf{w} = \mathbf{X}\mathbf{Y}^T$
  $\mathbf{t} = \mathbf{X}^T\mathbf{w}$
  $\mathbf{X} = \mathbf{X} - \frac{\mathbf{X}\mathbf{t}\mathbf{t}^T}{\mathbf{t}^T\mathbf{t}}$
  $\mathbf{T} = [\mathbf{T}, \mathbf{t}]$
end for

The weight vectors $\mathbf{W}^{(i)}$ will be used extensively in the discussion of the shrinkage properties of PLS regression. Let the compact SVD of $\mathbf{X}$ be

$$\mathbf{X} = \mathbf{U}_1 \Sigma \mathbf{V}_1^T, \tag{2.26}$$

where $\mathbf{U}_1 = [\mathbf{u}_1, \ldots, \mathbf{u}_r] \in \mathbb{R}^{d\times r}$, $\mathbf{V}_1 = [\mathbf{v}_1, \ldots, \mathbf{v}_r] \in \mathbb{R}^{n\times r}$ are matrices with orthonormal columns, $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)(\sigma_1 > \sigma_2 > \ldots > \sigma_r > 0)$, and $r = \text{rank}(\mathbf{X})$. Given $\mathbf{U}_1 = [\mathbf{u}_1, \ldots, \mathbf{u}_r] \in \mathbb{R}^{d\times r}$, there exists a matrix $\mathbf{U}_1^\perp \in \mathbb{R}^{d\times(d-r)}$ with orthonormal columns such that $[\mathbf{U}_1, \mathbf{U}_1^\perp]$ is an orthogonal matrix [172], i.e., $\mathbf{U}_1\mathbf{U}_1^T + \mathbf{U}_1^\perp{\mathbf{U}_1^\perp}^T = \mathbf{I}_d$ and ${\mathbf{U}_1^\perp}^T\mathbf{U}_1^\perp = \mathbf{I}_{d-r}$. We denote $\mathbf{U}_1^\perp$ as $\mathbf{U}_1^\perp = [\mathbf{u}_{r+1}, \ldots, \mathbf{u}_d]$ in the following discussion.

Shrinkage in Regression

The shrinkage regression methods are particularly effective when there is multicollinearity among the regressors. We assume that $\theta \in \mathbb{R}^d$ is the parameter to be estimated and our estimator is $\hat{\theta}$. We can compute the expectation error $\hat{\theta} - \theta$ at each point and average over all feasible points, leading to the mean squared error (MSE) for the estimation $\hat{\theta}$:

$$\begin{aligned}
\text{MSE}(\hat{\theta}) &= \mathbb{E}\left[\text{Tr}\left((\hat{\theta}-\theta)(\hat{\theta}-\theta)^T\right)\right] \\
&= \mathbb{E}\left[(\hat{\theta}-\theta)^T(\hat{\theta}-\theta)\right] \\
&= \mathbb{E}\left[\hat{\theta}^T\hat{\theta} - \mathbb{E}[\hat{\theta}]^T\mathbb{E}[\hat{\theta}] + \mathbb{E}[\hat{\theta}]^T\mathbb{E}[\hat{\theta}] - 2\hat{\theta}^T\theta + \theta^T\theta\right] \\
&= \mathbb{E}\left[\left(\hat{\theta}-\mathbb{E}[\hat{\theta}]\right)^T\left(\hat{\theta}-\mathbb{E}[\hat{\theta}]\right)\right] + (\mathbb{E}[\hat{\theta}]-\theta)^T(\mathbb{E}[\hat{\theta}]-\theta) \\
&= \text{var}(\hat{\theta}) + \text{bias}^2(\hat{\theta}),
\end{aligned} \tag{2.27}$$

where

$$\begin{aligned}
\text{var}(\hat{\theta}) &= \mathbb{E}\left[\left(\hat{\theta}-\mathbb{E}[\hat{\theta}]\right)^T\left(\hat{\theta}-\mathbb{E}[\hat{\theta}]\right)\right], \\
\text{bias}^2(\hat{\theta}) &= (\mathbb{E}[\hat{\theta}]-\theta)^T(\mathbb{E}[\hat{\theta}]-\theta).
\end{aligned}$$

This decomposition of MSE is called the bias-variance decomposition [79], in which the MSE is decomposed into two components: variance and squared bias. Following this decomposition, an estimator $\hat{\theta}$ is called unbiased if $\text{bias}(\hat{\theta}) = 0$ and biased otherwise.

It is well-known that Ordinary Least Squares (OLS) produces the minimum variance among all linear unbiased estimators in regression. It follows from the bias-variance decomposition that there may exist some biased estimators with small variance, which results in a smaller MSE compared to OLS. In other words, we hope to decrease $\text{var}(\hat{\theta})$ even when $\text{bias}(\hat{\theta})$ is increased. The biased estimators are very popular in statistics and machine learning [79, 173].

In linear regression, we are given the regressors $\mathbf{X} \in \mathbb{R}^{d\times n}$ and response $\mathbf{Y} \in \mathbb{R}^{1\times n}$. We assume there is a linear relationship between $\mathbf{X}$ and $\mathbf{Y}$:

$$\mathbf{Y} = \beta^T \mathbf{X} + \varepsilon, \tag{2.28}$$

where $\varepsilon \in \mathbb{R}^{1\times n}$ is the error term with $\mathbb{E}[\varepsilon] = 0$ and $\text{cov}(\varepsilon) = \sigma^2 \mathbf{I}_n$. As a result, we can compute the expectation and covariance of $\mathbf{Y}$ as

$$\begin{aligned} \mathbb{E}[\mathbf{Y}] &= \mathbb{E}[\beta^T \mathbf{X} + \varepsilon] = \beta^T \mathbf{X}, \\ \text{cov}(\mathbf{Y}) &= \mathbb{E}\left[(\mathbf{Y} - \mathbb{E}[\mathbf{Y}])^T (\mathbf{Y} - \mathbb{E}[\mathbf{Y}])\right] = \sigma^2 \mathbf{I}_n. \end{aligned}$$

We investigate the linear estimator $\hat{\theta} = \mathbf{S}\mathbf{Y}^T$. Note that throughout the discussion, we assume that $\mathbf{X}$ is fixed and the focus is $\mathbf{Y}$. Assume that $\mathbf{S}$ is independent on $\mathbf{Y}$, then the expectation and the covariance of $\hat{\theta}$ can be computed as follows:

$$\begin{aligned} \mathbb{E}[\hat{\theta}] &= \mathbb{E}[\mathbf{S}\mathbf{Y}^T] \\ &= \mathbf{S}\mathbb{E}[\mathbf{Y}^T] \\ &= \mathbf{S}\mathbf{X}^T \beta. \end{aligned} \tag{2.29}$$

$$\begin{aligned} \text{var}(\hat{\theta}) &= \mathbb{E}\left[(\hat{\theta} - \mathbb{E}[\hat{\theta}])^T (\hat{\theta} - \mathbb{E}[\hat{\theta}])\right] \\ &= \mathbb{E}\left[(\mathbf{S}\mathbf{Y}^T - \mathbf{S}\mathbf{X}^T\beta)^T (\mathbf{S}\mathbf{Y}^T - \mathbf{S}\mathbf{X}^T\beta)\right] \\ &= \mathbb{E}\left[(\mathbf{S}\varepsilon^T)^T (\mathbf{S}\varepsilon)\right] \\ &= \sigma^2 \,\text{Tr}(\mathbf{S}\mathbf{S}^T). \end{aligned} \tag{2.30}$$

In OLS, the estimator $\hat{\beta}_{OLS}$ is given as follows:

$$\hat{\beta}_{OLS} = \left(\mathbf{X}\mathbf{X}^T\right)^{\dagger}\mathbf{X}\mathbf{Y}^T = \mathbf{U}_1\Sigma^{-1}\mathbf{V}_1^T\mathbf{Y}^T = \sum_{i=1}^{r}\frac{1}{\sigma_i}\mathbf{u}_i(\mathbf{v}_i^T\mathbf{Y}^T) = \sum_{i=1}^{r}\mathbf{b}_i, \tag{2.31}$$

where $\mathbf{b}_i = \frac{1}{\sigma_i}\mathbf{u}_i(\mathbf{v}_i^T\mathbf{Y}^T) \in \mathbb{R}^d$. Defining $\mathbf{S}_{OLS} = \left(\mathbf{X}\mathbf{X}^T\right)^{\dagger}\mathbf{X}$ and using Eqs. (2.29) and (2.30), we obtain that

$$\begin{aligned}
\mathbb{E}[\hat{\beta}_{OLS}] &= \mathbf{S}_{OLS}\mathbf{X}^T\beta = \left(\mathbf{X}\mathbf{X}^T\right)^{\dagger}\mathbf{X}\mathbf{X}^T\beta = \mathbf{U}_1\mathbf{U}_1^T\beta, &(2.32)\\
\mathrm{var}(\hat{\beta}_{OLS}) &= \sigma^2\,\mathrm{Tr}(\mathbf{S}_{OLS}\mathbf{S}_{OLS}^T)\\
&= \sigma^2\,\mathrm{Tr}\left(\left(\mathbf{X}\mathbf{X}^T\right)^{\dagger}\mathbf{X}\left(\left(\mathbf{X}\mathbf{X}^T\right)^{\dagger}\mathbf{X}\right)^T\right)\\
&= \sigma^2\,\mathrm{Tr}(\mathbf{U}_1\Sigma^{-2}\mathbf{U}_1^T)\\
&= \sigma^2\sum_{i=1}^{r}\frac{1}{\sigma_i^2}. &(2.33)
\end{aligned}$$

Note that when $\beta \in \mathscr{R}(\mathbf{X}) = \mathscr{R}(\mathbf{U}_1)$, $\mathbb{E}[\hat{\beta}_{OLS}] = \mathbf{U}_1\mathbf{U}_1^T\beta = \beta$, which implies that OLS produces an unbiased estimator $\hat{\beta}_{OLS}$.

One consequence of Eq. (2.33) is that small singular values of $\mathbf{X}$ may lead to high variance of the estimator $\hat{\beta}_{OLS}$. In order to investigate the MSE of the general shrinkage estimator, we consider the shrinkage estimator $\hat{\beta}_s$ in the following form:

$$\hat{\beta}_s = \sum_{i=1}^{r} f(\sigma_i)\mathbf{b}_i, \tag{2.34}$$

where $\mathbf{b} = \frac{1}{\sigma_i}\mathbf{u}_i(\mathbf{v}_i^T\mathbf{Y}^T) \in \mathbb{R}^d$ and $f(\sigma_i)(i = 1,\ldots,r)$ are called shrinkage factors, which can be represented in the following matrix form:

$$\hat{\beta}_s = \mathbf{U}_1\Sigma^{-1}\mathbf{G}\mathbf{V}_1^T\mathbf{Y}^T, \tag{2.35}$$

where $\mathbf{G} = \mathrm{diag}\left(f(\sigma_1),\ldots,f(\sigma_r)\right) \in \mathbb{R}^{r\times r}$. Thus, we define $\mathbf{S}_s$ as follows:

$$\mathbf{S}_s = \mathbf{U}_1\Sigma^{-1}\mathbf{G}\mathbf{V}_1^T = \mathbf{U}_1\mathbf{G}\Sigma^{-1}\mathbf{V}_1^T. \tag{2.36}$$

Based on the above discussion, we can estimate the MSEs of estimators $\hat{\beta}_s$ and $\hat{\mathbf{Y}}_s$, which is summarized in the following theorem [146]:

Theorem 2.3 *Given the shrinkage estimator $\hat{\beta}_s$ defined in Eq.* (2.34)*, the MSEs of $\hat{\beta}_s$ and $\hat{\mathbf{Y}}_s$ are given as:*

$$\begin{aligned}
\mathrm{MSE}(\hat{\beta}_s) &= \sigma^2 \sum_{i=1}^{r} \frac{f(\sigma_i)^2}{\sigma_i^2} + \sum_{i=1}^{r} (f(\sigma_i)-1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2 \qquad (2.37)\\
\mathrm{MSE}(\hat{\mathbf{Y}}_s) &= \sigma^2 \sum_{i=1}^{r} f(\sigma_i)^2 + \sum_{i=1}^{r} \sigma_i^2 (f(\sigma_i)-1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2. \qquad (2.38)
\end{aligned}$$

Proof Based on the matrix representation of $\mathbf{S}_s$ in Eq. (2.36), we can simplify $\mathrm{Tr}(\mathbf{S}_s \mathbf{S}_s^T)$ as

$$\begin{aligned}
\mathrm{Tr}(\mathbf{S}_s \mathbf{S}_s^T) &= \mathrm{Tr}(\mathbf{U}_1 \Sigma^{-1} \mathbf{G} \mathbf{V}_1^T \mathbf{V}_1 \mathbf{G} \Sigma^{-1} \mathbf{U}_1^T)\\
&= \mathrm{Tr}(\mathbf{U}_1 \Sigma^{-2} \mathbf{G}^2 \mathbf{U}_1^T)\\
&= \mathrm{Tr}(\Sigma^{-2} \mathbf{G}^2)\\
&= \sum_{i=1}^{r} \frac{f(\sigma_i)^2}{\sigma_i^2}.
\end{aligned}$$

Following Eq. (2.30), the variance of $\hat{\beta}_s$ is

$$\begin{aligned}
\mathrm{var}(\hat{\beta}_s) &= \sigma^2 \mathrm{Tr}(\mathbf{S}_s \mathbf{S}_s^T)\\
&= \sigma^2 \sum_{i=1}^{r} \frac{f(\sigma_i)^2}{\sigma_i^2}.
\end{aligned}$$

Next we analyze the squared bias of the estimator $\hat{\beta}_s$:

$$\begin{aligned}
\mathrm{bias}^2(\hat{\beta}_s) &= (\mathbb{E}[\hat{\beta}_s] - \beta)^T (\mathbb{E}[\hat{\beta}_s] - \beta)\\
&= (\mathbf{S}_s \mathbf{X}^T \beta - \beta)^T (\mathbf{S}_s \mathbf{X}^T \beta - \beta)\\
&= (\mathbf{U}_1 \mathbf{G} \mathbf{U}_1^T \beta - \beta)^T (\mathbf{U}_1 \mathbf{G} \mathbf{U}_1^T \beta - \beta)\\
&= (\mathbf{U}_1 \mathbf{G} \mathbf{U}_1^T \beta - [\mathbf{U}_1, \mathbf{U}_1^{\perp}][\mathbf{U}_1, \mathbf{U}_1^{\perp}]^T \beta)^T (\mathbf{U}_1 \mathbf{G} \mathbf{U}_1^T \beta - [\mathbf{U}_1, \mathbf{U}_1^{\perp}][\mathbf{U}_1, \mathbf{U}_1^{\perp}]^T \beta)\\
&= \beta^T \mathbf{U}_1 (\mathbf{G} - \mathbf{I})^2 \mathbf{U}_1^T \beta + \beta^T \mathbf{U}_1^{\perp} \mathbf{U}_1^{\perp T} \beta\\
&= \sum_{i=1}^{r} (f(\sigma_i) - 1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2.
\end{aligned}$$

To summarize, the MSE of $\hat{\beta}_s$ is

$$\mathrm{MSE}(\hat{\beta}_s) = \sigma^2 \sum_{i=1}^{r} \frac{f(\sigma_i)^2}{\sigma_i^2} + \sum_{i=1}^{r} (f(\sigma_i) - 1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2.$$

Similarly, we can compute the variance and squared bias for the estimator $\hat{\mathbf{Y}}_s = \hat{\beta}_s^T \mathbf{X}$ as

$$\begin{aligned} \text{var}(\hat{\mathbf{Y}}_s) &= \sigma^2 \text{Tr}(\mathbf{S}_s \mathbf{S}_s^T) = \sigma^2 \sum_{i=1}^{r} \frac{f(\sigma_i)^2}{\sigma_i^2} \\ \text{bias}^2(\hat{\mathbf{Y}}_s) &= \sum_{i=1}^{r} \sigma_i^2 (f(\sigma_i) - 1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2. \end{aligned}$$

Thus, the MSE of $\hat{\mathbf{Y}}_s$ is:

$$\begin{aligned} \text{MSE}(\hat{\mathbf{Y}}_s) &= \text{var}(\hat{\mathbf{Y}}_s) + \text{bias}^2(\hat{\mathbf{Y}}_s) \\ &= \sigma^2 \sum_{i=1}^{r} f(\sigma_i)^2 + \sum_{i=1}^{r} \sigma_i^2 (f(\sigma_i) - 1)^2 (\mathbf{u}_i^T \beta)^2 + \sum_{i=r+1}^{d} (\mathbf{u}_i^T \beta)^2. \end{aligned}$$

This completes the proof of this theorem. ∎

Theorem 2.3 shows that when $f(\sigma_i) = 1$ for all i, then the bias is minimized. Note that $|f(\sigma_i) < 1|$ decreases the variance while $|f(\sigma_i) > 1|$ increases the variance. As a result, we typically set $|f(\sigma_i) < 1|$ to decrease the MSE of the estimator.

Principal Component Regression

Both Principal Component Regression (PCR) and PLS regression involve selecting a subspace onto which the response vector $\mathbf{Y}$ is projected. The major difference is that the subspace in PLS regression is determined by both $\mathbf{X}$ and $\mathbf{Y}$, while in PCR the first p components are used to approximate the data $\mathbf{X}$. Recall that in PLS1 regression in Algorithm 3, we select $\mathbf{w}$ that achieves the maximum covariance between $\mathbf{X}$ and $\mathbf{Y}$ at each step. In PCR, the approximation of $\mathbf{X}$ is given below if only the first p ($p \leq r = \text{rank}(\mathbf{X})$) principal components are used:

$$\tilde{\mathbf{X}} = \mathbf{U}_1(:, 1:p) \Sigma(1:p, 1:p) \mathbf{V}_1(:, 1:p)^T = \sum_{i=1}^{p} \sigma_i \mathbf{u}_i \mathbf{v}_i^T, \tag{2.39}$$

where $\mathbf{U}_1(:, 1:p)$ and $\mathbf{V}_1(:, 1:p)$ contain the first p columns of $\mathbf{U}_1$ and $\mathbf{V}_1$, respectively, $\Sigma(1:p, 1:p)$ contains the first p rows and first p columns of Σ, i.e., $\Sigma(1:p, 1:p) = \text{diag}(\sigma_1, \ldots, \sigma_p)$.

Based on the approximate formulation $\tilde{\mathbf{X}}$, we perform ordinary least squares by regressing $\mathbf{Y}$ on $\tilde{\mathbf{X}}$, leading to the following estimator $\hat{\beta}_{PCR}$:

$$\begin{aligned}\hat{\beta}_{PCR} &= \left(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\right)^{\dagger}\tilde{\mathbf{X}}\mathbf{Y}^T \\ &= \mathbf{U}_1(:,1:p)\Sigma(1:p,1:p)^{-1}\mathbf{V}_1(:,1:p)^T\mathbf{Y}^T \\ &= \sum_{i=1}^{p}\frac{1}{\sigma_i}\mathbf{u}_i\mathbf{v}_i^T\mathbf{Y}^T \\ &= \sum_{i=1}^{p}\mathbf{b}_i. \end{aligned} \tag{2.40}$$

It follows from Eq. (2.40) that the estimator $\hat{\beta}_{PCR}$ can be expressed equivalently in the following form:

$$\hat{\beta}_{PCR} = \sum_{i=1}^{r} f(\sigma_i)\mathbf{b}_i, \tag{2.41}$$

where the shrinkage factors $f(\sigma_i)$ $(i = 1,\ldots,r)$ in PCR are defined as

$$f(\sigma_i) = \begin{cases} 1, \text{ if } i \leq p \\ 0, \text{ otherwise,} \end{cases} \tag{2.42}$$

where $p \leq \text{rank}(\mathbf{X})$ is the number of selected principal components. Compared with OLS, the $\mathbf{b}_i$ associated with small singular values are removed in PCR. Thus, the variance of the estimator $\hat{\beta}_{PCR}$ is reduced at the cost of increased bias.

Ridge Regression

In ridge regression, the regression coefficients are shrunk by imposing a penalty on the ℓ_2-norm of the regression coefficients. Specifically, ridge regression minimizes the following loss function:

$$\hat{\beta}_{RR} = \arg\min_{\beta} \|\beta^T\mathbf{X} - \mathbf{Y}\|_2^2 + \lambda\|\beta\|_2^2, \tag{2.43}$$

where $\lambda > 0$ is called the regularization parameter, or complexity parameter, which controls the shrinkage of $\hat{\beta}_{RR}$. It is clear that $\lambda = 0$ corresponds to the OLS. It has been shown that the optimization problem in Eq. (2.43) is equivalent to the following problem [79, 173]:

$$\begin{aligned} \min_{\beta} \quad & \|\beta^T\mathbf{X} - \mathbf{Y}\| \\ \text{s.t.} \quad & \|\beta\| \leq t, \end{aligned} \tag{2.44}$$

for some parameter t which depends on λ. It has also been shown that there is a one-to-one correspondence between the complexity parameter λ in Eq. (2.43) and t in the problem in Eq. (2.44) [173].

Note that the loss function in ridge regression can be expressed as

$$\begin{aligned} L(\beta) &= \|\beta^T \mathbf{X} - \mathbf{Y}\|_2^2 + \lambda \|\beta\|_2^2 \\ &= \beta^T \mathbf{X}\mathbf{X}^T \beta - 2\beta^T \mathbf{X}\mathbf{Y}^T + \mathbf{Y}\mathbf{Y}^T + \lambda \beta^T \beta. \end{aligned}$$

Taking the derivative with respect to β and setting it to zero, we have

$$\left(\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I}\right)\beta = \mathbf{X}\mathbf{Y}^T. \tag{2.45}$$

Then the solution of ridge regression can be represented as

$$\begin{aligned} \hat{\beta}_{RR} &= (\mathbf{X}\mathbf{X}^T + \lambda \mathbf{I})^{-1} \mathbf{X}\mathbf{Y}^T \\ &= \mathbf{U}_1 (\Sigma^2 + \lambda \mathbf{I})^{-1} \Sigma \mathbf{V}_1^T \mathbf{Y}^T \\ &= \sum_{i=1}^{r} \frac{\sigma_i^2}{\sigma_i^2 + \lambda} \mathbf{u}_i (\mathbf{v}_i^T \mathbf{Y}^T) \\ &= \sum_{i=1}^{r} f(\sigma_i) \mathbf{b}_i, \end{aligned}$$

where the shrinkage factors $f(\sigma_i)$ $(i = 1, \ldots, r)$ are defined as

$$f(\sigma_i) = \frac{\sigma_i^2}{\sigma_i^2 + \lambda}. \tag{2.46}$$

Shrinkage Properties of PLS Regression

The shrinkage properties of PLS regression have been studied extensively in the literature [146, 167, 171, 174, 175]. In particular, the connection between PLS, the Lanczos algorithm and the conjugate gradient algorithm [146, 165, 171] provides some insights into the shrinkage properties of PLS regression. In fact, PLS is identical to a common implementation of the conjugate gradient algorithm for solving the normal equation [146]. In this subsection, we analyze the shrinkage properties of PLS regression by making use of the connections between PLS and the Lanczos algorithm.

In the following, the PLS estimator after the ith step is denoted as $\hat{\beta}_{PLS}^{(i)}$. Similar to OLS and ridge regression, in PLS we also assume that

$$\hat{\beta}_{PLS}^{(i)} = \mathbf{S}_{PLS}^{(i)} \mathbf{Y}^T.$$

However, $\mathbf{S}_{PLS}^{(i)}$ depends on $\mathbf{Y}$ in a complicated, nonlinear way. As a result, Theorem 2.3 does not hold since its assumption is violated. It turns out that PLS regression also yields a shrinkage estimator. Unfortunately, the shrinkage behavior of PLS regression is very complicated due to the nonlinear relationship between $\mathbf{S}_{PLS}$ and $\mathbf{Y}$.

Next we consider the PLS regression estimator $\hat{\beta}_{PLS}^{(i)}$ from the perspective of the Lanczos algorithm and the conjugate gradient algorithm. Consider the matrix $\mathbf{K}^{(i)}$ that consists of the Krylov sequence of $\mathbf{A} = \mathbf{X}\mathbf{X}^T \in \mathbb{R}^{d\times d}$ and $\mathbf{b} = \mathbf{X}\mathbf{Y}^T \in \mathbb{R}^d$:

$$\mathbf{K}^{(i)} \triangleq \left[\mathbf{A}^0\mathbf{b}, \mathbf{A}^1\mathbf{b}, \ldots, \mathbf{A}^{i-1}\mathbf{b}\right] \in \mathbb{R}^{d\times i}. \tag{2.47}$$

We denote the space spanned by the columns of $\mathbf{K}^{(i)}$ by $\mathscr{K}^{(i)}$. Define i^* as

$$i^* = \left|\{\sigma_i | \sigma_i > 0, \mathbf{v}_i^T\mathbf{Y}^T \neq 0\}\right|. \tag{2.48}$$

Following the definition of i^*, it is clear that $i^* \leq r = \text{rank}(\mathbf{X})$. This inequality is strict if there exists at least one singular vector $\mathbf{v}_i$ such that $\mathbf{v}_i^T\mathbf{Y}^T = 0$ or there is some singular value σ_i whose multiplicity is greater than 1. The quantity i^* is called the grade of $\mathbf{b}$ with respect to $\mathbf{A}$ [146]. Using the definition of the Krylov sequence, it is easy to verify the following results:

Lemma 2.1 *Let $\mathscr{K}^{(i)}$ and i^* be defined as above, then we have*

$$\dim \mathscr{K}^{(i)} = \begin{cases} i, & \text{if } i \leq i^* \\ i^*, & \text{if } i > i^*. \end{cases} \tag{2.49}$$

Using the Krylov sequence $\mathbf{K}^{(i)}$, it can be shown that the estimator $\beta_{PLS}^{(i)}$ after the ith step of PLS can be expressed in the following form [169]:

Theorem 2.4 *The PLS estimator after the ith step can be expressed in the following form:*

$$\beta_{PLS}^{(i)} = \boldsymbol{K}^{(i)}\left[\left(\boldsymbol{K}^{(i)}\right)^T \mathbf{A}\boldsymbol{K}^{(i)}\right]^{\dagger}\left(\boldsymbol{K}^{(i)}\right)^T \boldsymbol{X}\boldsymbol{Y}^T. \tag{2.50}$$

The proof of this theorem is rather involved, and thus it is skipped. The full proof can be found in [169]. Note that any matrix $\mathbf{M}$ can be used in place of $\mathbf{K}^{(i)}$ as long as $\mathbf{M}$ forms a basis for the Krylov space $\mathscr{K}^{(i)}$ [171]. Due to the equivalence between the space spanned by the columns

of $\mathbf{W}^{(i)}$ and the Krylov space $\mathscr{K}^{(i)}$ [169, 171], $\mathbf{W}^{(i)}$ can be considered as an orthogonal basis of $\mathscr{K}^{(i)}$. In fact, $\mathbf{W}^{(i)}$ is widely used in the discussion of the shrinkage properties of PLS regression as ${\mathbf{W}^{(i)}}^T\mathbf{A}\mathbf{W}^{(i)}$ possesses a lot of appealing properties [146]. An equivalent formulation for $\hat{\beta}_{PLS}^{(i)}$ is given in the following theorem:

Theorem 2.5 *The PLS estimator $\hat{\beta}_{PLS}^{(i)}$ after the ith step can be obtained by solving the following problem:*

$$\begin{aligned} \max_{\beta} \quad & \|\boldsymbol{Y}-\beta^T\boldsymbol{X}\|_2 \\ \text{s.t.} \quad & \beta \in \mathscr{K}^{(i)}. \end{aligned} \tag{2.51}$$

A consequence of this theorem is that $\hat{\beta}_{PLS}^{(i)} = \hat{\beta}_{OLS}$ if $i \geq i^*$. Thus, in the following we focus on the shrinkage property of PLS regression when $i < i^*$.

Define $\mathbf{D}^{(i)}$ as

$$\mathbf{D}^{(i)} \triangleq {\mathbf{W}^{(i)}}^T\mathbf{A}\mathbf{W}^{(i)} \in \mathbb{R}^{i\times i}. \tag{2.52}$$

It is clear that $\mathbf{D}^{(i)}$ is symmetric and positive semidefinite. It can be further shown that $\mathbf{D}^{(i)}$ is a tridiagonal matrix [146]. Many appealing properties of $\mathbf{D}^{(i)}$ are summarized in [146]. For example, [146] shows that all eigenvalues of $\mathbf{D}^{(i)}$ are distinct under a mild assumption. In particular, we are interested in the eigenvalues of $\mathbf{D}^{(i)}$. We denote the eigenvalues of $\mathbf{D}^{(i)}$ by

$$\mu_1^{(i)} \geq \mu_2^{(i)} \geq \ldots \mu_i^{(i)} \geq 0. \tag{2.53}$$

It turns out that more interesting properties hold for the eigenvalues of $\mathbf{D}^{(i)}$ when $i < i^*$. In the following, we discuss one important property regarding the eigenvalues of $\mathbf{D}^{(i)}$ [146]:

Lemma 2.2 *If $i < i^*$, then all eigenvalues of $\boldsymbol{D}^{(i)}$ are positive and distinct. Thus, we have*

$$\mu_1^{(i)} > \mu_2^{(i)} > \ldots \mu_i^{(i)} > 0. \tag{2.54}$$

The proof of this lemma can be found in [146].

Define the polynomial $f^{(i)}(\lambda)$ of degree i as follows:

$$f^{(i)}(\lambda) \triangleq 1 - \prod_{j=1}^{i}\left(1-\frac{\lambda}{\mu_j^i}\right) = 1 - \frac{g^{(i)}(\lambda)}{g^{(i)}(0)}, \tag{2.55}$$

where the auxiliary function $g^{(i)}$ is defined as

$$g^{(i)}(\lambda) \stackrel{\triangle}{=} \sum_{j=1}^{i} \left(\mu_j^{(i)} - \lambda\right).$$

In fact, $g^{(i)}$ is the characteristic polynomial of the matrix $\mathbf{D}^{(i)}$. It is clear that $f^{(i)}(0) = 0$, which implies that the polynomial $f^{(i)}$ can be represented as

$$f^{(i)}(\lambda) = \lambda \pi^{(i)}(\lambda),$$

where $\pi^{(i)}$ is a polynomial of degree $i-1$. Also note that for $j = 1, \ldots, i$, we have

$$f^{(i)}(\mu_j^{(i)}) = 1,$$

which implies the following result:

$$\mu_j^{(i)} \pi(\mu_j^{(i)}) = 1 \Leftrightarrow \pi^{(i)}(\mu_j^{(i)}) = \frac{1}{\mu_j^{(i)}}. \tag{2.56}$$

It turns out that the shrinkage factors of PLS can be represented using the polynomial $\pi^{(i)}$. The results are summarized the following theorem [146, 171]:

Theorem 2.6 *The PLS estimator after the ith step for $i < i^*$ can be represented in the following form:*

$$\hat{\beta}_{PLS}^{(i)} = \pi^{(i)}(\mathbf{A})\boldsymbol{X}\boldsymbol{Y}^T. \tag{2.57}$$

Proof Let the SVD of $\mathbf{D}^{(i)} \in \mathbb{R}^{i \times i}$ be $\mathbf{D}^{(i)} = \mathbf{U}_{di}\Sigma_{di}\mathbf{U}_{di}^T$, where $\mathbf{U}_{di} \in \mathbb{R}^{i \times i}$ is an orthogonal matrix and $\Sigma_{di} \in \mathbb{R}^{i \times i}$ is a diagonal matrix. Following the function defined for the matrix, we have

$$\begin{aligned}
\mathbf{D}^{(i)^{-1}} &= \mathbf{U}_{di}\,\mathrm{diag}\,(\mu_1^{(i)}, \ldots, \mu_i^{(i)})^{-1}\mathbf{U}_{di}^T \\
&= \mathbf{U}_{di}\,\mathrm{diag}\,(\mu_1^{(i)^{-1}}, \ldots, \mu_i^{(i)^{-1}})\mathbf{U}_{di}^T \\
&= \mathbf{U}_{di}\,\mathrm{diag}\,(\pi^{(i)}(\mu_1^{(i)}), \ldots, \pi^{(i)}(\mu_i^{(i)}))\mathbf{U}_{di}^T \\
&= \pi^{(i)}(\mathbf{D}^{(i)}),
\end{aligned} \tag{2.58}$$

where in the third step we use the equation in Eq. (2.56).

Recall that $\mathbf{W}^{(i)}$ forms an orthonormal basis of $\mathscr{K}^{(i)}$. Then we can set $\mathbf{K}^{(i)} = \mathbf{W}^{(i)}$. Substituting Eq. (2.58) into Eq. (2.50), we obtain

$$\begin{aligned}
\beta_{PLS}^{(i)} &= \mathbf{K}^{(i)} \left[\left(\mathbf{K}^{(i)} \right)^T \mathbf{A} \mathbf{K}^{(i)} \right]^{-1} \left(\mathbf{K}^{(i)} \right)^T \mathbf{b} \\
&= \mathbf{W}^{(i)} \left[\left(\mathbf{W}^{(i)} \right)^T \mathbf{A} \mathbf{W}^{(i)} \right]^{-1} \left(\mathbf{W}^{(i)} \right)^T \mathbf{b} \\
&= \mathbf{W}^{(i)} {\mathbf{D}^{(i)}}^{-1} \left(\mathbf{W}^{(i)} \right)^T \mathbf{b} \\
&= \mathbf{W}^{(i)} \pi^{(i)}(\mathbf{D}^{(i)}) \left(\mathbf{W}^{(i)} \right)^T \mathbf{b}. \qquad (2.59)
\end{aligned}$$

Note that $\mathbf{W}^{(i)} {\mathbf{W}^{(i)}}^T$ is an orthogonal projection onto the space $\mathscr{K}^{(i)}$, the space spanned by $\{\mathbf{A}^0\mathbf{b}, \mathbf{A}\mathbf{b}, \ldots, \mathbf{A}^{i-1}\mathbf{b}\}$, thus we have

$$\mathbf{W}^{(i)} {\mathbf{W}^{(i)}}^T \mathbf{A}^j \mathbf{b} = \mathbf{A}^j \mathbf{b},\ j = 0, 1, \ldots, i-1. \qquad (2.60)$$

Recall that $\pi^{(i)}(\lambda)$ is a polynomial of degree $i-1$. We denote the polynomial $\pi^{(i)}(\lambda)$ as

$$\pi^{(i)}(\lambda) = \alpha_0 + \alpha_1 \lambda + \ldots + \alpha_{i-1} \lambda^{i-1} = \sum_i \alpha_i \lambda^i.$$

By applying Eq. (2.60) for j times, we can obtain the following result for $j < i$:

$$\begin{aligned}
& \mathbf{W}^{(i)} \left(\alpha_j {\mathbf{D}^{(i)}}^j \right) \left(\mathbf{W}^{(i)} \right)^T \mathbf{b} \\
= \ & \alpha_j \mathbf{W}^{(i)} \left(\left(\mathbf{W}^{(i)} \right)^T \mathbf{A} \mathbf{W}^{(i)} \right)^j \left(\mathbf{W}^{(i)} \right)^T \mathbf{b} \\
= \ & \alpha_j \mathbf{A}^j \mathbf{b}.
\end{aligned}$$

Thus Eq. (2.59) can be simplified as follows:

$$\begin{aligned}
\beta_{PLS}^{(i)} &= \mathbf{W}^{(i)} \pi^{(i)}(\mathbf{D}^{(i)}) \left(\mathbf{W}^{(i)} \right)^T \mathbf{b} \\
&= \pi^{(i)}(\mathbf{A}) \mathbf{b} = \pi^{(i)}(\mathbf{A}) \mathbf{X} \mathbf{Y}^T. \qquad (2.61)
\end{aligned}$$

This completes the proof of this theorem. ∎

Corollary 2.1 *If* $\dim(\boldsymbol{K}^{(i)}) = i$, *then*

$$\hat{\beta}_{PLS}^{(i)} = \sum_{j=1}^{r} f^{(i)}(\sigma_j^2) \boldsymbol{b}_j, \qquad (2.62)$$

where $f^{(i)}$ *is defined in Eq.* (2.55).

Proof It follows from the SVD of $\mathbf{X}$ in Eq. (2.26) and Theorem 2.6 that

$$\begin{aligned}
\hat{\beta}_{PLS}^{(i)} &= \pi^{(i)}(\mathbf{A})\mathbf{b} \\
&= \pi^{(i)}(\mathbf{X}\mathbf{X}^T)\mathbf{X}\mathbf{Y}^T \\
&= \pi^{(i)}(\mathbf{U}_1\Sigma^2\mathbf{U}_1^T)\mathbf{X}\mathbf{Y}^T \\
&= \mathbf{U}_1\pi^{(i)}(\Sigma^2)\mathbf{U}_1^T\mathbf{X}\mathbf{Y}^T \\
&= \mathbf{U}_1\,\mathrm{diag}\,(\pi^{(i)}(\sigma_1^2),\ldots,\pi^{(i)}(\sigma_r^2))\mathbf{U}_1^T\mathbf{U}_1\Sigma\mathbf{V}_1^T\mathbf{Y}^T \\
&= \mathbf{U}_1\,\mathrm{diag}\,(\pi^{(i)}(\sigma_1^2),\ldots,\pi^{(i)}(\sigma_r^2))\Sigma\mathbf{V}_1^T\mathbf{Y}^T \\
&= \sum_{j=1}^{r}\pi^{(i)}(\sigma_j^2)\sigma_j\mathbf{u}_j(\mathbf{v}_j^T\mathbf{Y}^T) \\
&= \sum_{j=1}^{r}\pi^{(i)}(\sigma_j^2)\sigma_j^2\mathbf{b}_j \\
&= \sum_{j=1}^{r}f^{(i)}(\sigma_j^2)\mathbf{b}_j,
\end{aligned}$$

where the last equality follows from $f^{(i)}(\lambda)=\pi^{(i)}(\lambda)\lambda$ for any $\lambda\in\mathbb{R}$. ∎

The shrinkage factors for PLS can be defined based on Corollary 2.1. By comparing the PLS regression estimator $\hat{\beta}_{PLS}$ with the estimators obtained by OLS, PCR, and ridge regression, we can observe that the main difference lies in the definitions for the shrinkage factors. Note that all shrinkage factors in OLS, PCR, and ridge regression are not greater than 1. However, this is not true for PLS regression, and some counterexamples are given in [146, 176]. Furthermore, unlike OLS, PCR, and ridge regression, the shrinkage factors in PLS regression also depends on the response $\mathbf{Y}$ nonlinearly. When $\mathbf{S}_s$ is independent of $\mathbf{Y}$, Theorem 2.3 implies that it is desirable to impose the constraint that $|f^(i)(\sigma_i)|<1$ to decrease the overall MSE. However, it is not clear how to control the shrinkage factors in PLS regression, since $\mathbf{S}_s$ depends on $\mathbf{Y}$. [167] and [146] proposes to upper bound the absolute value of shrinkage factors by 1 in PLS regression. As reported in [146], bounding the absolute value of the PLS shrinkage factors by 1 seems to yield a lower MSE empirically.

In the last part of this section, we give a result concerning the ℓ_2-norm of the estimator $\hat{\beta}_{PLS}^{(i)}$ in the PLS regression:

Theorem 2.7

$$\|\hat{\beta}_{PLS}^{(1)}\|_2 \leq \|\hat{\beta}_{PLS}^{(2)}\|_2 \leq \ldots \leq \|\hat{\beta}_{PLS}^{(i*)}\|_2 = \|\hat{\beta}_{OLS}\|_2. \tag{2.63}$$

The algebraic proof of this theorem is given in [174]. Another geometric proof is also given in [177], which shows that $\|\hat{\beta}_{PLS}^{(i)}\|_2 \leq \|\hat{\beta}_{OLS}\|_2$. Two alternative proofs can also be found in [171].

2.5 Partial Least Squares Classification

In this section we discuss the application of PLS for classification. In classification, $\mathbf{X}$ is typically the data while $\mathbf{Y}$ is called the indicator matrix that encodes the class membership for the corresponding data in $\mathbf{X}$. Linear Discriminant Analysis (LDA) [166] is a popular discriminative tool [166], and it is well-known that CCA is equivalent to LDA under certain circumstances, which was first recognized in 1938 by Bartlett [178]. Since PLS and CCA share many common features, PLS is also used widely in many applications as a classification tool [145, 179], although it is not inherently designed for this purpose. Some typical applications of PLS in classification include Alzheimer's disease discrimination [180], classification between Arabica and Robusta coffee beans [181], water pollution classification [182], separation between active and inactive compounds in a quantitative structure-activity relationship study [183], hard red wheat classification [184], microarray classification [160], and other applications in a variety of other areas [185–188].

In the following, we consider the use of PLS for multi-class classification and discuss the connections between PLS and LDA [166]. We denote the ith class as $\mathscr{C}_i$, the sample size of the ith class as $n_i = |\mathscr{C}_i|$, the mean of the ith class as $\bar{\mathbf{x}}_i = \frac{1}{n_i}\sum_{j\in\mathscr{C}_i}\mathbf{x}_j$, and the mean of all samples as $\bar{\mathbf{x}} = \frac{1}{n}\mathbf{X}1$. Since we assume $\mathbf{X}$ are centered, then $\bar{\mathbf{x}} = 0$. Without loss of generality, we assume that the data matrix $\mathbf{X}$ is organized according to k classes as $\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_k]$, where $\mathbf{X}_i \in \mathbb{R}^{d\times n_i}$ contains all samples belonging to the ith class.

In LDA, the between-class covariance matrix $\mathbf{S}_b$ and the within-class covariance matrix $\mathbf{S}_w$ are defined as follows:

$$\mathbf{S}_b = \sum_{i=1}^{k} n_i(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T, \tag{2.64}$$

$$\mathbf{S}_w = \sum_{i=1}^{k}\sum_{\mathbf{x}_j\in\mathscr{C}_i}(\mathbf{x}_j - \bar{\mathbf{x}}_i)(\mathbf{x}_j - \bar{\mathbf{x}}_i)^T. \tag{2.65}$$

Based on the between-class covariance matrix $\mathbf{S}_b$ and the within-class covariance matrix $\mathbf{S}_w$, the total covariance matrix $\mathbf{S}_t$ can be decomposed as the sum of $\mathbf{S}_b$ and $\mathbf{S}_w$:

$$\mathbf{S}_t = \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T = \mathbf{S}_b + \mathbf{S}_w. \tag{2.66}$$

The objective of LDA is to maximize the distance between instances belonging to different classes while minimizing the distance between instances belonging to the same class after projection. Formally, LDA solves the following optimization problem:

$$\max J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}, \tag{2.67}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the linear projection. It has been shown that LDA reduces to computing the eigenvectors corresponding the largest nonzero eigenvalues of the matrix $\mathbf{S}_t^{\dagger}\mathbf{S}_b$.

When PLS is applied for classification, one block of variables corresponds to the data matrix $\mathbf{X}$ while the other block encodes the class membership information. The commonly used scheme to encode the class membership information is the 1-of-k binary coding scheme [173]:

$$\mathbf{Y}_0 = \begin{bmatrix} 1_{n_1} & 0 & \dots & 0 \\ 0 & 1_{n_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1_{n_k}. \end{bmatrix}^T \in \mathbb{R}^{k \times n}. \tag{2.68}$$

Note that in our discussion of PLS, we typically assume the columns of $\mathbf{Y}$ is centered, i.e., $\mathbf{Y}1_n = 0$. It is equivalent to performing the centering operation for the matrix $\mathbf{Y}_0$ as

$$\mathbf{Y} = \mathbf{Y}_0 \mathbf{C}, \tag{2.69}$$

where $\mathbf{C} = \mathbf{I} - \frac{1}{n} 1 1^T$ is called the centering matrix. Note that $\mathbf{C} = \mathbf{C}^T$ and $\mathbf{C}^2 = \mathbf{C}$, which implies that $\mathbf{C}$ is an orthogonal projection [172].

An alternative form of the indicator matrix as proposed in [145] is defined as follows:

$$\mathbf{Z}_0 = \begin{bmatrix} 1_{n_1} & 0 & \dots & 0 \\ 0 & 1_{n_2} & \dots 0 & \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1_{n_{k-1}} \\ 0 & 0 & \dots & 0 \end{bmatrix}^T \in \mathbb{R}^{(k-1) \times n}. \tag{2.70}$$

The difference between $\mathbf{Z}_0$ and $\mathbf{Y}_0$ is that the last column in $\mathbf{Y}_0$ is deleted in $\mathbf{Z}_0$. Note that in this section we focus on multi-class classification, where the classes are mutually exclusive. As a result, $\mathbf{Z}_0$ still conveys the full membership information by deleting the last column of $\mathbf{Y}_0$. Similarly, we also perform the centering operation for $\mathbf{Z}_0$ in the discussion of PLS:

$$\mathbf{Z} = \mathbf{Z}_0\mathbf{C}.$$

Note that for the sample covariance matrix $\mathbf{S}_y = \mathbf{Y}\mathbf{Y}^T \in \mathbb{R}^{k\times k}$, where the indicator matrix $\mathbf{Y}$ is given in Eq. (2.68), we have $\text{rank}(\mathbf{S}_y) = k-1$ in multi-class classification and $\text{rank}(\mathbf{S}_y) = k$ in multi-label classification. Similarly, for multi-classification, we have $\mathbf{S}_z = \mathbf{Z}\mathbf{Z}^T \in \mathbb{R}^{(k-1)\times(k-1)}$ and $\text{rank}(\mathbf{S}_z) = k-1$. As discussed in Section 2.2, the projection vector $\mathbf{w}$ is the eigenvector of $\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T$, and $\mathbf{c}$ is the eigenvector of $\mathbf{Y}\mathbf{X}^T\mathbf{X}\mathbf{Y}^T$. In classification, we are more interested in the projection for the data matrix $\mathbf{X}$. The following results, established in [145], elucidate the connection between PLS and LDA.

Theorem 2.8 *Given the data matrix $\boldsymbol{X}$, and indicator matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$ as defined in Eqs.* (2.68) *and* (2.70)*, and assume that $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ are all centered. Then we have*

$$\boldsymbol{X}\boldsymbol{Y}^T\boldsymbol{Y}\boldsymbol{X}^T = \boldsymbol{H}^*, \text{ where } \boldsymbol{H}^* = \sum_{i=1}^{k} n_i^2(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T, \tag{2.71}$$

$$\boldsymbol{X}\boldsymbol{Z}^T\boldsymbol{Z}\boldsymbol{X}^T = \boldsymbol{H}^{**}, \text{ where } \boldsymbol{H}^{**} = \sum_{i=1}^{k-1} n_i^2(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T. \tag{2.72}$$

Proof Since $\mathbf{X}$ has already been centered, we have $\mathbf{X}\mathbf{C} = \mathbf{X}$. It follows that

$$\begin{aligned}
\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T &= \mathbf{X}\mathbf{C}^T\mathbf{Y}_0^T\mathbf{Y}_0\mathbf{C}\mathbf{X}^T \\
&= (\mathbf{X}\mathbf{C})\,\mathbf{Y}_0^T\mathbf{Y}_0\,(\mathbf{X}\mathbf{C})^T \\
&= \mathbf{X}\mathbf{Y}_0^T\mathbf{Y}_0\mathbf{X} \\
&= \begin{bmatrix}\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k\end{bmatrix}
\begin{bmatrix}
1_{n_1}1_{n_1}^T & 0 & \ldots & 0 \\
0 & 1_{n_2}1_{n_2}^T & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & 1_{n_k}1_{n_k}^T
\end{bmatrix}
\begin{bmatrix}\mathbf{X}_1^T \\ \mathbf{X}_2^T \\ \vdots \\ \mathbf{X}_k^T\end{bmatrix} \\
&= \sum_{i=1}^{k} \mathbf{X}_i 1_{n_i} 1_{n_i}^T \mathbf{X}_i^T
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{k} (n_i \bar{\mathbf{x}}_i)(n_i \bar{\mathbf{x}}_i)^T \\
&= \sum_{i=1}^{k} n_i^2 \bar{\mathbf{x}}_i (\bar{\mathbf{x}}_i)^T \\
&= \sum_{i=1}^{k} n_i^2 (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T \\
&= \mathbf{H}^*.
\end{aligned}
$$

In the above derivation we made use of the fact that $\mathbf{x}_i = \frac{1}{n_i}\mathbf{X}_i 1_{n_i}$ and $\bar{\mathbf{x}} = 0$ as $\mathbf{X}$ is centered. Similarly, we can show that

$$
\mathbf{X}\mathbf{Z}^T\mathbf{Z}\mathbf{X}^T = \sum_{i=1}^{k-1} n_i^2 (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T = \mathbf{H}^{**}.
$$

This completes the proof of this theorem. ∎

Theorem 2.8 shows that the projection vector for $\mathbf{X}$, i.e., $\mathbf{w}$, is the eigenvector of $\mathbf{H}^*$ if the indicator matrix is defined in Eq. (2.68). Note that $\mathbf{H}^*$ is similar to the between-class covariance matrix $\mathbf{S}_b$, while the weights are different. This connection to LDA reveals the success of PLS in classification. The argument also holds for $\mathbf{H}^{**}$ if the indicator matrix is defined in Eq. (2.70). Although the eigenstructures of the matrix $\mathbf{H}^*$ and $\mathbf{H}^{**}$ are computed in PLS, they are still different from the between-class covariance matrix $\mathbf{S}_b$.

Theorem 2.9 *Given the data matrix $\boldsymbol{X}$, and the class indicator matrix $\mathbf{Z}$ as defined in Eq.* (2.70)*, and assume that $\boldsymbol{X}$ and $\mathbf{Z}$ are centered. Then we have*

$$
\boldsymbol{X}\mathbf{Z}^T(\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\boldsymbol{X}^T = \mathbf{S}_b = \sum_{i=1}^{k} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^T. \tag{2.73}
$$

Proof Denote $\mathbf{S}_Z = \mathbf{Z}\mathbf{Z}^T$. Then we can simplify $\mathbf{S}_Z$ as follows:

$$
\begin{aligned}
\mathbf{S}_Z &= \mathbf{Z}\mathbf{Z}^T \\
&= \mathbf{Z}_0\mathbf{C}\mathbf{C}^T\mathbf{Z}_0^T \\
&= \mathbf{Z}_0\mathbf{C}\mathbf{Z}_0 \\
&= \mathbf{Z}_0\left(\mathbf{I} - \frac{1}{n}1_n 1_n^T\right)\mathbf{Z}_0^T \\
&= \mathbf{Z}_0\mathbf{Z}_0^T - \frac{1}{n}\mathbf{Z}_0 1_n 1_n^T \mathbf{Z}_0^T
\end{aligned}
$$

$$
\begin{aligned}
&= \begin{bmatrix} n_1 & 0 & \dots & 0 \\ 0 & n_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & n_{k-1} \end{bmatrix} - \frac{1}{n} \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_{k-1} \end{bmatrix} \begin{bmatrix} n_1 & n_2 & \dots & n_{k-1} \end{bmatrix} \\
&= \mathbf{D} - \frac{1}{n}\mathbf{v}\mathbf{v}^T,
\end{aligned}
$$

where $\mathbf{D} = \text{diag}\,(n_1, n_2, \dots, n_{k-1}) \in \mathbb{R}^{(k-1)\times(k-1)}$ and $\mathbf{v} = [n_1, n_2, \dots, n_{k-1}]^T \in \mathbb{R}^{k-1}$. Using the Sherman-Morrison formula [172], we can compute the inverse of $\mathbf{S}_Z$ as

$$
\begin{aligned}
\mathbf{S}_Z^{-1} &= \left(\mathbf{D} - \frac{1}{n}\mathbf{v}\mathbf{v}^T\right)^{-1} \\
&= \mathbf{D}^{-1} + \frac{1}{n - \mathbf{v}^T\mathbf{D}^{-1}\mathbf{v}} \left(\mathbf{D}^{-1}\mathbf{v}\mathbf{v}^T\mathbf{D}^{-1}\right) \\
&= \mathbf{D}^{-1} + \frac{1}{n - \sum_{i=1}^{k-1} n_i} \left(1_{k-1}1_{k-1}^T\right) \\
&= \mathbf{D}^{-1} + \frac{1}{n_k} \left(1_{k-1}1_{k-1}^T\right).
\end{aligned}
$$

It also follows that

$$
\begin{aligned}
\mathbf{X}\mathbf{Z}_0^T &= [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k]\,\mathbf{Z}_0 \\
&= \left[\mathbf{X}_1 1_{n_1}, \mathbf{X}_2 1_{n_2}, \dots, \mathbf{X}_{k-1} 1_{n_{k-1}}\right] \\
&= [n_1\bar{\mathbf{x}}_1, n_2\bar{\mathbf{x}}_2, \dots, n_{k-1}\bar{\mathbf{x}}_{k-1}].
\end{aligned}
$$

Therefore, we can rewrite $\mathbf{X}\mathbf{Z}(\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\mathbf{X}^T$ as follows:

$$
\begin{aligned}
\mathbf{X}\mathbf{Z}^T(\mathbf{Z}\mathbf{Z}^T)^{-1}\mathbf{Z}\mathbf{X}^T &= \mathbf{X}\mathbf{C}^T\mathbf{Z}_0^T \left(\mathbf{D}^{-1} + \frac{1}{n_k}\left(1_{k-1}1_{k-1}^T\right)\right)\mathbf{Z}_0\mathbf{C}\mathbf{X}^T \\
&= \mathbf{X}\mathbf{Z}_0^T \left(\mathbf{D}^{-1} + \frac{1}{n_k}\left(1_{k-1}1_{k-1}^T\right)\right)\mathbf{Z}_0\mathbf{X}^T \\
&= \mathbf{X}\mathbf{Z}_0^T\mathbf{D}^{-1}\mathbf{Z}_0\mathbf{X}^T + \frac{1}{n_k}\mathbf{X}\mathbf{Z}_0^T 1_{k-1}1_{k-1}^T\mathbf{Z}_0\mathbf{X}^T \\
&= \sum_{i=1}^{k-1} n_i\bar{\mathbf{x}}_i\bar{\mathbf{x}}_i^T + \frac{1}{n_k}\sum_{i=1}^{k-1} n_i\bar{\mathbf{x}}_i \left(\sum_{i=1}^{k-1} n_i\bar{\mathbf{x}}_i\right)^T \\
&= \sum_{i=1}^{k-1} n_i\bar{\mathbf{x}}_i\bar{\mathbf{x}}_i^T + \frac{1}{n_k}(-n_k\bar{\mathbf{x}}_k)\left((-n_k\bar{\mathbf{x}}_k)\right)^T
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{i=1}^{k} n_i \bar{\mathbf{x}}_i \bar{\mathbf{x}}_i^T \\
&= \sum_{i=1}^{k} n_i (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T \\
&= \mathbf{S}_b
\end{aligned}
$$

In the above derivation we have made use of the fact that $\sum_{i=1}^{k} n_i \bar{\mathbf{x}}_i = \sum_{i=1}^{n} \mathbf{x}_i = 0$, which implies that $\sum_{i=1}^{k-1} n_i \bar{\mathbf{x}}_i = -n_k \bar{\mathbf{x}}_k$. This completes the proof of this theorem. ∎

Recall that in PLS we try to maximize the covariance between two blocks of variables after projection. In contrast, CCA tries to maximize the correlation. Note that the objective function of PLS can be expressed as

$$
\operatorname{cov}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2 = \operatorname{var}\left(\mathbf{X}^T\mathbf{w}\right) \operatorname{corr}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2 \operatorname{var}\left(\mathbf{Y}^T\mathbf{c}\right),
$$

which can be considered as penalized version of CCA, with PCA for $\mathbf{X}$ and $\mathbf{Y}$ simultaneously. In classification tasks, it is not necessary to analyze the variance of the indicator matrix $\mathbf{Y}$. Thus, we can consider the following objective function:

$$
\frac{\operatorname{cov}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2}{\operatorname{var}\left(\mathbf{Y}^T\mathbf{c}\right)} = \operatorname{var}\left(\mathbf{X}^T\mathbf{w}\right) \operatorname{corr}\left(\mathbf{X}^T\mathbf{w}, \mathbf{Y}^T\mathbf{c}\right)^2. \tag{2.74}
$$

The optimal weight vector $\mathbf{w}^*$ can be obtained by solving the following optimization problem:

$$
\begin{aligned}
\max_{\mathbf{w},\mathbf{c}} \quad & \mathbf{w}^T\mathbf{X}\mathbf{Y}^T\mathbf{c} \\
\text{s.t.} \quad & \mathbf{w}^T\mathbf{w} = 1 \\
& \mathbf{c}^T\mathbf{Y}\mathbf{Y}^T\mathbf{c} = 1.
\end{aligned} \tag{2.75}
$$

Using the Lagrange dual function in optimization theory, it can be shown that $\mathbf{w}^*$ is the eigenvector corresponding to the largest eigenvalue of the following eigenvalue problem:

$$
\mathbf{X}\mathbf{Y}^T\left(\mathbf{Y}\mathbf{Y}^T\right)^{-1}\mathbf{Y}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{w}. \tag{2.76}
$$

It can also be shown that multiple weight vectors $\{\mathbf{w}_i\}_{i=1}^{\ell} (\ell < k)$ under the orthogonality constraint $\mathbf{w}_i^T\mathbf{w}_j = \delta_{ij}$ can be obtained by computing eigenvectors corresponding to the largest ℓ eigenvalues of $\mathbf{X}\mathbf{Y}^T\left(\mathbf{Y}\mathbf{Y}^T\right)^{-1}\mathbf{Y}\mathbf{X}^T$, where $\delta_{ij} = 0$ if $i \neq j$ and $\delta_{ij} = 1$ otherwise.

Using the modified objective function in Eq. (2.74) and the new indicator matrix in Eq. (2.70), we can show that the revised PLS amounts to solving the eigenvector of the between-class covariance matrix $\mathbf{S}_b$ in LDA. The main results are summarized in the following theorem:

Theorem 2.10 *If the modified objective function given in Eq.* (2.74) *is used in PLS classification, and the indicator matrix is defined as* $\mathbf{Z}_0$ *in Eq.* (2.70)*, then the weight vector for* $\boldsymbol{X}$ *is the eigenvector of the between-class covariance matrix* $\mathbf{S}_b$ *associated with the largest eigenvalue.*

Proof As discussed above, the weight vector $\mathbf{w}$ for $\mathbf{X}$ is the eigenvector of the following eigenvalue problem associated with the largest eigenvalue:

$$\mathbf{X}\mathbf{Z}^T\left(\mathbf{Z}\mathbf{Z}^T\right)^{-1}\mathbf{Z}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{w}, \tag{2.77}$$

where $\mathbf{Z} = \mathbf{Z}_0\mathbf{C}$. It follows from Theorem 2.9 that $\mathbf{X}\mathbf{Z}^T\left(\mathbf{Z}\mathbf{Z}^T\right)^{-1}\mathbf{Z}\mathbf{X}^T = \mathbf{S}_b$. This completes the proof of this theorem. ∎

2.6 Relationship between PLS and CCA

In this section we discuss the relationship between PLS and CCA by establishing a unified framework using the canonical ridge analysis [189]. The equivalence relationship between CCA and orthonormalized PLS will be discussed in detail in Chapter 3.

CCA and PLS share many common features. They both model the relationship between two blocks of variables by seeking the optimal projections. However, CCA finds the optimal projection by maximizing the correlation coefficient after projection while PLS maximizes the covariance after the projection. CCA reduces to a generalized eigenvalue problem while PLS involves an iterative procedure. At each iteration of PLS, it also reduces to eigenvalue problems in Eqs. (2.3) and (2.6). However, different deflation schemes are applied in PLS, resulting in different variants of PLS.

Table 2.1: A list of algorithms derived from different combinations of γ_x and γ_y in the unified framework.

Algorithm	γ_x	γ_y
Canonical Correlation Analysis	0	0
Regularized CCA	$\frac{\lambda}{1+\lambda}$	0
Orthonormalized PLS	0	1
Regularized OPLS	$\frac{\lambda}{1+\lambda}$	1
Partial Least Squares	1	1

The similarities between PLS and CCA have been observed for long time [96, 97, 190]. For example, a unified framework for PCA, PLS, CCA and Multiple Linear Regression (MLR)

is proposed in [190], where each technique can be obtained by selecting a different parameter value. In this section, we focus on the unified framework established in [189] using the canonical ridge analysis, which considers the following problem:

$$\max_{|\mathbf{w}_x|=|\mathbf{w}_y|=1} \frac{\operatorname{cov}(\mathbf{w}_x^T\mathbf{X}, \mathbf{w}_y^T\mathbf{Y})}{\sqrt{\left([1-\gamma_x]\operatorname{var}(\mathbf{w}_x^T\mathbf{X})+\gamma_x\right)\left([1-\gamma_y]\operatorname{var}(\mathbf{w}_y^T\mathbf{Y})+\gamma_y\right)}}, \tag{2.78}$$

where $0 \leq \gamma_x,\ \gamma_y \leq 1$ are the regularization parameters, and $\mathbf{w}_x \in \mathbb{R}^d$, $\mathbf{w}_y \in \mathbb{R}^k$ are projection vectors (or weight vectors) for $\mathbf{X}$ and $\mathbf{Y}$, respectively. It can be verified that the optimization problem in Eq. (2.78) reduces to the following eigenvalue problem:

$$\left([1-\gamma_x]\mathbf{X}\mathbf{X}^T+\gamma_x\mathbf{I}\right)^{-1}\mathbf{X}\mathbf{Y}^T\left([1-\gamma_x]\mathbf{Y}\mathbf{Y}^T+\gamma_y\mathbf{I}\right)^{-1}\mathbf{Y}\mathbf{X}^T\mathbf{w}_x=\lambda\mathbf{w}_x. \tag{2.79}$$

By choosing different values for γ_x and γ_y, the unified framework encompasses PLS, OPLS and CCA, as summarized in Table 2.1. From the unified framework, PLS can be considered as a special case of "regularized OPLS" in which $\lambda \to +\infty$.

Chapter 3

CANONICAL CORRELATION ANALYSIS

3.1 Introduction

Canonical Correlation Analysis (CCA) is a classical technique for finding the correlations between two sets of multi-dimensional variables [191]. Being considered as a classical technique, CCA has been studied and applied extensively in a variety of domains such as computational biology and information retrieval recently. CCA makes use of two views of the same set of objects and projects them onto lower-dimensional spaces in which they are maximally correlated. It is becoming a powerful tool to analyze the so-called paired data $(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X}$ and $\mathbf{Y}$ are two different representations of the same set of objects [192]. Such scenario arises in many real-world applications. In parallel corpus [193], there are texts in two languages that are similar in content, which can be considered as the paired data set. Specifically, the texts in different languages correspond to different views of the same semantics. In the content-based image retrieval [15], the image and the associated texts can be considered as two different views of the same semantic representation. CCA extracts features from both views such that the underlying semantics of different representations can be captured. In [15, 194] CCA is applied to combine image and text data together to enable the retrieval of images from a text query without reference to the label information associated with images and promising results have been achieved. CCA has been used effectively to combine multiple assays, including the DNA copy number (or comparative genomic hybridization, CGH), gene expression, and Single Nucleotide Polymorphism (SNP) data, on the same set of patient samples [195]. In supervised learning, the data and labels can be considered as the paired views for the underlying objects.

Various extensions of CCA have been proposed, such as kernel CCA [194], and sparse CCA [138, 196–198]. In addition, theoretical properties of CCA have been established. For example, the probabilistic interpretation of CCA has been proposed [199], and the connection between CCA and the least squares formulation has been established in the general setting [138]. The sparse CCA formulations have been used to identify genes whose expression is correlated with other data. For example, in [198, 200, 201] sparse CCA identifies gene with expression correlated with regions of DNA copy number change; In [202] sparse CCA identifies genes

that have expression correlated with SNPs; In [196] sparse CCA identifies sets of genes on two different microarray platforms that have correlated expression.

When CCA is used in supervised dimensionality reduction, one view is derived from the data and the other view is derived from the class labels. In this setting, the projection of the data onto the lower-dimensional space is directed by the label information, leading to the extraction of discriminative features. In contrast to multi-class classification, where the classes are mutually exclusive, the classes in multi-label learning are overlapped and correlated, making it challenging to predict all relevant labels for a given instance. One appealing feature of CCA in multi-label learning is that the label correlations can be captured, resulting in informative projection for the data [138, 197, 203].

CCA reduces to a generalized eigenvalue problem, which is computationally expensive to solve. Many popular dimensionality reduction techniques in machine learning, including partial least squares [98], hypergraph spectral learning [56], and linear discriminant analysis [166], can be formulated as a generalized eigenvalue problem [204]. In this chapter, we will investigate a class of generalized eigenvalue problems, which include all these formulations as special cases. In addition, we will discuss its relationship with the generalized Rayleigh quotient cost function and some numerical algorithms such as the incomplete Cholesky decomposition.

The rest of this chapter is organized as follows. The classical CCA is introduced in Section 3.2. The recent developments of sparse CCA are summarized in Section 3.3. The relationship between CCA and orthonormalized PLS is discussed in Section 3.4. We study a family of generalized eigenvalue problems in Section 3.6. Also in the discussion of CCA, we assume that both $\{\mathbf{x}_i\}_1^n$ and $\{\mathbf{y}_i\}_1^n$ are centered, i.e., $\sum_{i=1}^n \mathbf{x}_i = \mathbf{0}$, and $\sum_{i=1}^n \mathbf{y}_i = \mathbf{0}$.

3.2 Classical Canonical Correlation Analysis

CCA is a classical technique to assess the relationship between two sets of variables. Before discussing the CCA formulation, we first introduce the linear correlation coefficient, which measures the linear dependence between two sets of variables.

Linear Correlation Coefficient

The linear correlation coefficient between two variables x and y ($x, y \in \mathbb{R}$) is defined as the covariance of the two variables divided by the product of their standard deviations:

$$\rho = \frac{\text{cov}(x,y)}{\text{std}(x)\,\text{std}(y)} = \frac{\mathbb{E}[(x-\bar{x})(y-\bar{y})]}{\text{std}(x)\,\text{std}(y)}, \tag{3.1}$$

where $\mathbb{E}[z]$ denotes the expectation of variable z, $\text{std}(x)$ and $\text{std}(y)$ are the standard deviations of x and y, and $\bar{x}$ and $\bar{y}$ are the means of x and y, respectively. Given the samples $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$, the sample correlation coefficient, or the Pearson's product-moment correlation coefficient between x and y, is defined as

$$\rho = \frac{\sum_{i=1}^n (x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^n (x_i-\bar{x})^2}\sqrt{\sum_{i=1}^n (y_i-\bar{y})^2}}. \tag{3.2}$$

Note that when $\bar{x} = \bar{y} = 0$, the sample correlation coefficient can be simplified as:

$$\rho = \frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2}\sqrt{\sum_{i=1}^n y_i^2}}, \tag{3.3}$$

which can be expressed in the following vector form:

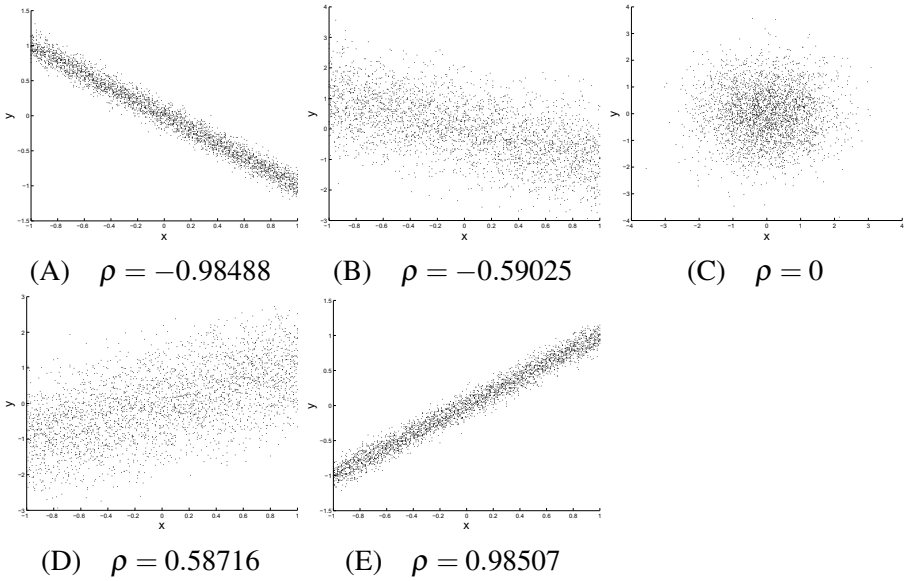
$$\rho = \frac{\mathbf{x}^T\mathbf{y}}{\|\mathbf{x}\|_2\|\mathbf{y}\|_2}, \tag{3.4}$$

where $\mathbf{x} = [x_1, \ldots, x_n]^T$ and $\mathbf{y} = [y_1, \ldots, y_n]^T$.

The linear correlation coefficient measures the strength of the linear association between two variables x and y. In fact, the sample correlation coefficient is the standardized version of the sample covariance. It can be verified easily that the correlation coefficient is between -1 and $+1$ using Eq. (3.4). The sign of ρ shows the direction of the association. A negative value of ρ indicates a negative correlation while a positive value of ρ implies a positive association between x and y. In particular, the value of -1 implies a perfect negative correlation while the value of $+1$ shows a perfect positive correlation. The value of 0 indicates a lack of linear association. Figure 3.1 gives some sample distributions and the corresponding linear correlation coefficients.

Intuitively, the correlation coefficient measures the extent to which the two variables can be fitted by a straight line. The line is also called the regression line or the least squares

line, since the sum of squared distances from all data points to the line is minimized. This definition of correlation coefficients also reveals some interesting connections between correlation coefficient and linear regression. Note that the correlation coefficient can only capture the linear association between two variables. Thus, nonlinear association cannot be revealed by the correlation coefficient.



(A) $\rho = -0.98488$ (B) $\rho = -0.59025$ (C) $\rho = 0$

(D) $\rho = 0.58716$ (E) $\rho = 0.98507$

Figure 3.1: Illustration of the linear correlation coefficient.

The Maximum Correlation Formulation for CCA

Canonical Correlation Analysis (CCA) [129] is a classical dimensionality reduction technique developed in statistics [191] and machine learning [194]. The motivation of CCA is that the linear relationship between variables may not be obvious in the original space, even though they are highly correlated. In order to capture the underlying linear association, CCA tries to find basis vectors for the two sets of variables such that they are maximally correlated after they are projected onto the basis vectors.

Let $\mathbf{X} \in \mathbb{R}^{d \times n}$ and $\mathbf{Y} \in \mathbb{R}^{k \times n}$ be the matrix representations for two different sets of variables, where $\mathbf{x}_i$ and $\mathbf{y}_i$ correspond to the ith sample. CCA computes two projection vectors, $\mathbf{w}_x \in \mathbb{R}^d$ and $\mathbf{w}_y \in \mathbb{R}^k$, such that the correlation coefficient

$$\rho = \frac{\mathbf{w}_x^T \mathbf{X}\mathbf{Y}^T \mathbf{w}_y}{\|\mathbf{w}_x^T \mathbf{X}\| \|\mathbf{w}_y^T \mathbf{Y}\|} \tag{3.5}$$

is maximized. Here we apply Eq. (3.4) to compute the correlation coefficient between $\mathbf{w}_x^T\mathbf{X}$ and $\mathbf{w}_y^T\mathbf{Y}$, since we assume that both $\{\mathbf{x}_i\}_{i=1}^n$ and $\{\mathbf{y}_i\}_{i=1}^n$ are centered. Note that $(\mathbf{w}_x^T\mathbf{X}, \mathbf{w}_y^T\mathbf{Y})$ is often called the pair of canonical variables, or canonical variate pair [191].

Since ρ is invariant to the scaling of $\mathbf{w}_x$ and $\mathbf{w}_y$, CCA can be formulated equivalently as the following optimization problem:

$$\begin{aligned} \max_{\mathbf{w}_x, \mathbf{w}_y} \quad & \mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y \\ \text{s.t.} \quad & \mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x = 1, \\ & \mathbf{w}_y^T\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y = 1. \end{aligned} \tag{3.6}$$

The Lagrangian L associated with the problem in (3.6) is

$$L(\mathbf{w}_x, \mathbf{w}_y, \lambda_x, \lambda_y) = \mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y - \frac{\lambda_x}{2}(\mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x - 1) - \frac{\lambda_y}{2}(\mathbf{w}_y^T\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y - 1). \tag{3.7}$$

We compute the derivatives of L with respect to $\mathbf{w}_x$ and $\mathbf{w}_y$ and set them to zero. This leads to the following equations:

$$\mathbf{X}\mathbf{Y}^T\mathbf{w}_y - \lambda_x\mathbf{X}\mathbf{X}^T\mathbf{w}_x = 0, \tag{3.8}$$

$$\mathbf{Y}\mathbf{X}^T\mathbf{w}_x - \lambda_y\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y = 0. \tag{3.9}$$

As a result, we have

$$\begin{aligned} \lambda_x &= \lambda_x\mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x \\ &= \mathbf{w}_x^T(\lambda_x\mathbf{X}\mathbf{X}^T\mathbf{w}_x) \\ &= \mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y \\ &= \mathbf{w}_y^T\mathbf{Y}\mathbf{X}^T\mathbf{w}_x \\ &= \mathbf{w}_y^T(\lambda_y\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y) \\ &= \lambda_y(\mathbf{w}_y^T\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y) \\ &= \lambda_y \end{aligned}$$

In the following discussion, we hence use λ to denote both λ_x and λ_y as they are always identical. The equalities in Eqs. (3.8) and (3.9) can be formulated equivalently in the following

form:

$$\begin{bmatrix} \mathbf{X}\mathbf{X}^T & \mathbf{X}\mathbf{Y}^T \\ \mathbf{Y}\mathbf{X}^T & \mathbf{Y}\mathbf{Y}^T \end{bmatrix} \begin{bmatrix} \mathbf{w}_x \\ \mathbf{w}_y \end{bmatrix} = (\lambda + 1) \begin{bmatrix} \mathbf{X}\mathbf{X}^T & 0 \\ 0 & \mathbf{Y}\mathbf{Y}^T \end{bmatrix} \begin{bmatrix} \mathbf{w}_x \\ \mathbf{w}_y \end{bmatrix}. \tag{3.10}$$

Thus, the projections $\mathbf{w}_x$ and $\mathbf{w}_y$ can be computed simultaneously by solving the generalized eigenvalue problem in Eq. (3.10). This formulation also suggests the generalization of CCA to multiple sets of variables, which will be discussed in detail in Section 3.2.

Since we focus on the use of CCA in multi-label learning, where $\mathbf{X}$ corresponds to the data and $\mathbf{Y}$ corresponds to the associated label information, the projection of $\mathbf{X}$ is of more interests. Assume that $\mathbf{Y}\mathbf{Y}^T$ is nonsingular. It follows from Eq. (3.9) that

$$\mathbf{w}_y = \frac{1}{\lambda} \left(\mathbf{Y}\mathbf{Y}^T\right)^{-1} \mathbf{Y}\mathbf{X}^T \mathbf{w}_x. \tag{3.11}$$

Substituting Eq. (3.11) into Eq. (3.8), we obtain the following genearalized eigenvalue problem:

$$\mathbf{X}\mathbf{Y}^T \left(\mathbf{Y}\mathbf{Y}^T\right)^{-1} \mathbf{Y}\mathbf{X}^T \mathbf{w}_x = \lambda^2 \mathbf{X}\mathbf{X}^T \mathbf{w}_x. \tag{3.12}$$

It is clear that $\mathbf{w}_x$ can also be obtained by solving the following optimization problem:

$$\begin{aligned} \max_{\mathbf{w}_x} \quad & \mathbf{w}_x^T \mathbf{X}\mathbf{Y}^T \left(\mathbf{Y}\mathbf{Y}^T\right)^{-1} \mathbf{Y}\mathbf{X}^T \mathbf{w}_x \\ \text{s.t.} \quad & \mathbf{w}_x^T \mathbf{X}\mathbf{X}^T \mathbf{w}_x = 1. \end{aligned} \tag{3.13}$$

After determining the first pair of $(\mathbf{w}_x, \mathbf{w}_y)$, we next focus on the second pair. We denote the ith pair of projection vectors as $(\mathbf{w}_x^{(i)}, \mathbf{w}_y^{(i)})$. We impose the constraint that the ith pair of canonical variables, $(\mathbf{w}_x^{(i)T}\mathbf{X}, \mathbf{w}_y^{(i)T}\mathbf{Y})$, is the pair of linear combinations having unit variance and maximizing the correlation among all choices that are uncorrelated with the previous $i-1$ pairs of canonical variables. Formally, we solve the following optimization problem in order to obtain the second pair of projection vectors $(\mathbf{w}_x^{(2)}, \mathbf{w}_y^{(2)})$:

$$\begin{aligned} \max_{\mathbf{w}_x, \mathbf{w}_y} \quad & \mathbf{w}_x^{(2)T} \mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} \\ \text{s.t.} \quad & \mathbf{w}_x^{(2)T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(2)} = 1, \\ & \mathbf{w}_y^{(2)T} \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(2)} = 1, \\ & \mathbf{w}_x^{(2)T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} = 0, \\ & \mathbf{w}_y^{(2)T} \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(1)} = 0. \end{aligned} \tag{3.14}$$

Similarly, we can construct the Lagrangian:

$$\begin{aligned} & L(\mathbf{w}_x^{(2)}, \mathbf{w}_y^{(2)}, \lambda_x, \lambda_y, \alpha_x, \alpha_y) \\ = \quad & \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \frac{\lambda_x}{2}(\mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(2)} - 1) - \frac{\lambda_y}{2}(\mathbf{w}_y^{(2)^T} \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(2)} - 1) \\ & -\alpha_x \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} - \alpha_y \mathbf{w}_y^{(2)^T} \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(1)}. \end{aligned}$$

Taking the derivatives of L with respect to $\mathbf{w}_x^{(2)}$ and $\mathbf{w}_y^{(2)}$ and setting them to zero, we have

$$\mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \lambda_x \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(2)} - \alpha_x \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} = 0 \tag{3.15}$$

$$\mathbf{Y}\mathbf{X}^T \mathbf{w}_x^{(2)} - \lambda_y \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \alpha_y \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(1)} = 0. \tag{3.16}$$

It follows from Eq. (3.15) that

$$\begin{aligned} \lambda_x &= \lambda_x \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(2)} \\ &= \mathbf{w}_x^{(2)^T} \left(\mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \alpha_x \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} \right) \\ &= \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \alpha_x \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} \\ &= \mathbf{w}_x^{(2)^T} \mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)}, \end{aligned}$$

and

$$\begin{aligned} \alpha_x &= \alpha_x \mathbf{w}_x^{(1)^T} \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(1)} \\ &= \mathbf{w}_x^{(1)^T} \left(\mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} - \lambda_x \mathbf{X}\mathbf{X}^T \mathbf{w}_x^{(2)} \right) \\ &= \mathbf{w}_x^{(1)^T} \mathbf{X}\mathbf{Y}^T \mathbf{w}_y^{(2)} \\ &= \lambda \mathbf{w}_y^{(1)} \mathbf{Y}\mathbf{Y}^T \mathbf{w}_y^{(2)} \\ &= 0. \end{aligned}$$

In the above derivation we use the fact that $\mathbf{w}_x^{(1)^T} \mathbf{X}\mathbf{Y}^T = \lambda \mathbf{w}_y^{(1)} \mathbf{Y}\mathbf{Y}^T$ based on Eq. (3.9). Similarly, it follows from Eq. (3.16) that

$$\lambda_y = \mathbf{w}_y^{(2)^T} \mathbf{Y}\mathbf{X}^T \mathbf{w}_x^{(2)}, \text{ and } \alpha_y = 0.$$

Thus, we have $\lambda_x = \lambda_y$ and denote it as λ as before. Using similar arguments as in the discussion of the first pair of projection vectors, we can show that $[\mathbf{w}_x^{(2)}; \mathbf{w}_y^{(2)}]$ is the second eigenvector of the eigenvalue problem in Eq. (3.10), and $\mathbf{w}_x^{(2)}$ is the second eigenvector of the generalized eigenvalue problem in Eq. (3.12).

Following the previous discussions, we can show that multiple projection vectors under the orthonormality constraints can be computed simultaneously by solving the generalized eigenvalue problem in Eq. (3.12), which is equivalent to the following optimization problem [194]:

$$\begin{aligned} \max_{\mathbf{W}\in\mathbb{R}^{d\times\ell}} \quad & \mathrm{Tr}(\mathbf{W}^T\mathbf{X}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}\mathbf{Y}\mathbf{X}^T\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W} = \mathbf{I}_\ell, \end{aligned} \tag{3.17}$$

where $\mathbf{W} \in \mathbb{R}^{d\times\ell}$ is the projection matrix for $\mathbf{X}$, ℓ is the number of projection vectors, and $\mathbf{I}_\ell$ is the identity matrix. The solution to the optimization problem in Eq. (3.17) consists of the top ℓ eigenvectors of the generalized eigenvalue problem in Eq. (3.12).

The Distance Minimization Formulation for CCA

The optimization problem in Eq. (3.6) can be expressed as a distance minimization problem, where we try to minimize the distance between two views $\mathbf{X}$ and $\mathbf{Y}$ after they are projected onto the lower-dimensional space:

$$\begin{aligned} \min_{\mathbf{w}_x,\mathbf{w}_y} \quad & \|\mathbf{w}_x^T\mathbf{X} - \mathbf{w}_y^T\mathbf{Y}\|_2 \\ \text{s.t.} \quad & \mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x = 1, \\ & \mathbf{w}_y^T\mathbf{X}\mathbf{X}^T\mathbf{w}_y = 1. \end{aligned} \tag{3.18}$$

The equivalence relationship between the optimization problems in Eqs. (3.6) and (3.18) can be verified as follows:

$$\begin{aligned} \|\mathbf{w}_x^T\mathbf{X} - \mathbf{w}_y^T\mathbf{Y}\|_2^2 &= \mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x + \mathbf{w}_y^T\mathbf{Y}\mathbf{Y}^T\mathbf{w}_y - 2\mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y \\ &= 2 - 2\mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y, \end{aligned}$$

where we use the constraints that $\mathbf{w}_x^T\mathbf{X}\mathbf{X}^T\mathbf{w}_x = \mathbf{w}_y^T\mathbf{X}\mathbf{X}^T\mathbf{w}_y = 1$. Thus, maximizing $\mathbf{w}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{w}_y$ is equivalent to minimizing $\|\mathbf{w}_x^T\mathbf{X} - \mathbf{w}_y^T\mathbf{Y}\|_2^2$ or $\|\mathbf{w}_x^T\mathbf{X} - \mathbf{w}_y^T\mathbf{Y}\|_2$.

When multiple projection vectors are required, it can be shown that the optimization problem in Eq. (3.17) can also be formulated as a distance minimization problem in which the

Frobenius norm is used instead:

$$\begin{aligned} \min_{\mathbf{W}_x,\mathbf{W}_y} \quad & \|\mathbf{W}_x^T\mathbf{X} - \mathbf{W}_y^T\mathbf{Y}\|_F \\ \text{s.t.} \quad & \mathbf{W}_x^T\mathbf{X}\mathbf{X}^T\mathbf{W}_x = \mathbf{I} \\ & \mathbf{W}_y^T\mathbf{X}\mathbf{X}^T\mathbf{W}_y = \mathbf{I}. \end{aligned} \tag{3.19}$$

Similarly, we can verify the equivalence relationship between the distance minimization formulation and the original optimization problem in Eq. (3.17) as follows:

$$\begin{aligned} & \|\mathbf{W}_x^T\mathbf{X} - \mathbf{W}_y^T\mathbf{Y}\|_F^2 \\ = \; & \mathrm{Tr}\left((\mathbf{W}_x^T\mathbf{X} - \mathbf{W}_y^T\mathbf{Y})(\mathbf{W}_x^T\mathbf{X} - \mathbf{W}_y^T\mathbf{Y})^T\right) \\ = \; & \mathrm{Tr}(\mathbf{W}_x^T\mathbf{X}\mathbf{X}^T\mathbf{W}_x) + \mathrm{Tr}(\mathbf{W}_y^T\mathbf{Y}\mathbf{Y}^T\mathbf{W}_y) - 2\,\mathrm{Tr}(\mathbf{W}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{W}^T) \\ = \; & 2\,\mathrm{Tr}(\mathbf{I}) - 2\,\mathrm{Tr}(\mathbf{W}_x^T\mathbf{X}\mathbf{Y}^T\mathbf{W}^T). \end{aligned}$$

The distance minimization formulation possesses a number of appealing properties, which make its connections to other classical problems, such as linear regression, apparent. Thus, different variants of CCA can be obtained by incorporating some penalty terms [201], including the ℓ_2-norm regularization, the ℓ_1-norm regularization [124], and the elastic net regularization [126].

Regularized CCA

In practice, the performance of CCA critically depends on the regularization, especially when the sample size is relatively small as compared to the data dimensionality. Note that the generalized eigenvalue problem in Eq. (3.12) can be transformed into an eigenvalue problem by multiplying $(\mathbf{X}\mathbf{X}^T)^{-1}$ on both sides, where $\mathbf{X}\mathbf{X}^T$ is the sample covariance matrix. The solution to the eigenvalue problem may not be reliable if $\mathbf{X}\mathbf{X}^T$ is close to be singular. To improve the robustness and prevent overfitting, one common approach is to add a scaled identity matrix to the sample covariance matrices $\mathbf{X}\mathbf{X}^T$ and $\mathbf{Y}\mathbf{Y}^T$ [103,204], resulting in the following generalized eigenvalue problem:

$$\mathbf{X}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T + \gamma_y\mathbf{I})^{-1}\mathbf{Y}\mathbf{X}^T\mathbf{w}_x = \lambda^2(\mathbf{X}\mathbf{X}^T + \gamma_x\mathbf{I})\mathbf{w}_x, \tag{3.20}$$

where $\gamma_x > 0$ and $\gamma_y > 0$ are the regularization parameters. The regularization employed in CCA can also be interpreted from the Bayesian perspective [199].

In the above discussions, the regularization terms on both $\mathbf{X}$ and $\mathbf{Y}$ are considered. In fact, only the regularization on $\mathbf{X}$ affects the computation of $\mathbf{W}_x$. We will discuss the effects of regularization on $\mathbf{X}$ and $\mathbf{Y}$ in detail in Section 3.4.

CCA for Multiple Sets of Variables

In traditional CCA, only two sets of variables are considered. In fact, CCA can be generalized to deal with problems with more than two sets of variables. The CCA formulation for multiple sets of variables can compute linear projections for multiple sets of variables simultaneously [205]. We will discuss CCA for multiple sets of variables based on the minimum distance formation in the following.

Let $\mathbf{X}^{(i)} \in \mathbb{R}^{d_i \times n}$ be the ith view, for $i = 1, \ldots, K$, where K is the number of views. CCA for multiple sets of variables involves the following optimization problem [205, 206]:

$$\begin{aligned} \min_{\mathbf{W}^{(1)},\ldots,\mathbf{W}^{(K)}} \quad & \sum_{i,j=1,i\neq j}^{K} \|\mathbf{W}^{(i)^T}\mathbf{X}^{(i)} - \mathbf{W}^{(j)^T}\mathbf{X}^{(i)}\|_F \\ \text{s.t.} \quad & \mathbf{W}^{(i)^T}\mathbf{X}^{(i)}\mathbf{X}^{(i)^T}\mathbf{W}^{(i)} = \mathbf{I},\ i = 1, \ldots, K, \end{aligned} \tag{3.21}$$

where $\mathbf{W}^{(i)}$ is the projection matrix for the ith view $\mathbf{X}^{(i)}$. The main idea of the generalized CCA for multiple sets of variables is that the sum of all pairwise distances between different views in the projected subspace is minimized. Similar to the traditional CCA discussed above, it can be shown that the optimization problem in Eq. (3.21) reduces to the following generalized eigenvalue problem [192]:

$$\begin{bmatrix} \mathbf{C}_{11} & \mathbf{C}_{12} & \cdots & \mathbf{C}_{1K} \\ \mathbf{C}_{21} & \mathbf{C}_{22} & \cdots & \mathbf{C}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_{K1} & \mathbf{C}_{K2} & \cdots & \mathbf{C}_{KK} \end{bmatrix} \begin{bmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \\ \vdots \\ \mathbf{w}^{(K)} \end{bmatrix} = (\lambda + 1) \begin{bmatrix} \mathbf{C}_{11} & 0 & \cdots & 0 \\ 0 & \mathbf{C}_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{C}_{KK} \end{bmatrix} \begin{bmatrix} \mathbf{w}^{(1)} \\ \mathbf{w}^{(2)} \\ \vdots \\ \mathbf{w}^{(K)} \end{bmatrix}$$

where $\mathbf{C}_{ij} = \mathbf{X}^{(i)}\mathbf{X}^{(j)^T}$ is the sample covariance matrix between the ith set of variables and the jth set of variables. It reduces to the generalized eigenvalue problem in Eq. (3.10) for traditional CCA when $K = 2$.

3.3 Sparse Canonical Correlation Analysis

Recently, many extensions of CCA have been proposed as new problems arise in various applications. For example, as multiple assays on the same set of patients become more popular, it

requires effective tools to integrate the data and select important features in genomic data analysis [207]. One challenge on the analysis of genomic data is that the dimensionality is much larger than the sample size. To circumvent this problem, various sparse CCA formulations have been proposed in the past.

Sparsity attracts a lot of attention in statistics and machine learning recently [79, 124, 208]. Sparsity often leads to easy interpretation, and it has been incorporated into many formulations such as multivariate linear regression [124], principal component analysis [125], and CCA [138, 196–198, 200–202, 207, 209–211]. Sparse CCA computes pairs of linear projections such that the maximal correlation is achieved in the reduced subspace with a small number of variables being involved in each projection. Indeed, it has been shown that a significant proportion of the correlations can be captured using a relatively small number of variables [209].

The existing sparse CCA formulations can be roughly categorized into three classes:

- The first approach formulates CCA as an equivalent least squares problem and then applies the ℓ_1-norm regularization [124] to the resulting least squares formulation.
- The second approach employs the iterative greedy scheme to compute the projection vectors sequentially.
- The third approach extends the CCA formulation based on the probabilistic interpretation of CCA from the Bayesian perspective.

Sparse CCA via Linear Regression

It has been shown in [138] that CCA is equivalent to the least squares formulation given the target matrix

$$\tilde{\mathbf{T}} = \mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-\frac{1}{2}} \in \mathbb{R}^{n\times k}. \tag{3.22}$$

The detailed proof of this equivalence relationship will be given in Chapter 4. Based on the equivalence relationship between CCA and the least squares formulation, sparsity in CCA can be achieved by applying an appropriate regularization to the least squares formulation. It is well-known that the ℓ_1-norm regularization can induce sparse solutions [124, 208], and it has been introduced into the least squares formulation, resulting in the so-called lasso [124]. Based

Algorithm 4 The Greedy Iterative Sparse CCA Algorithm

Input: $\mathbf{X}$, $\mathbf{Y}$, ℓ, and other control parameters.
Output: $\mathbf{W}_x$, $\mathbf{W}_y$.
for $i = 1$ to ℓ **do**
  Compute $\mathbf{w}_x^{(i)}$ and $\mathbf{w}_y^{(i)}$ using the greedy algorithm.
  Update $\mathbf{X}$ and $\mathbf{Y}$ using $\mathbf{w}_x^{(i)}$ and $\mathbf{w}_y^{(i)}$.
end for
Set $\mathbf{W}_x = [\mathbf{w}_x^{(1)}, \ldots, \mathbf{w}_x^{(\ell)}]$ and $\mathbf{W}_y = [\mathbf{w}_y^{(1)}, \ldots, \mathbf{w}_y^{(\ell)}]$.

on the equivalence relationship between CCA and least squares, the ℓ_1-norm regularized least squares CCA formulation, called "LS-CCA$_1$", has been developed, which minimizes the following objective function:

$$L_1(\mathbf{W}, \lambda) = \sum_{j=1}^{k} \left(\sum_{i=1}^{n} (\mathbf{x}_i^T \mathbf{w}_j - \tilde{\mathbf{T}}_{ij})^2 + \lambda \|\mathbf{w}_j\|_1 \right). \tag{3.23}$$

LS-CCA$_1$ can be solved efficiently using existing solvers for the ℓ_1 regularized linear regression [212–214].

Sparse CCA via Iterative Greedy Algorithms

The second approach employs greedy schemes to compute sparse projections iteratively [196, 198, 200, 202, 207, 209, 210]. The overall structure of this type of algorithms is summarized in Algorithm 4, where ℓ is the number of pairs of projection vectors $(\mathbf{w}_x, \mathbf{w}_y)$. This approach computes the projection vectors in a sequential fashion, and at each step the projection pair $(\mathbf{w}_x, \mathbf{w}_y)$ is computed using a greedy algorithm. The major difference among different sparse CCA algorithms lies in the different greedy algorithms used to compute $(\mathbf{w}_x, \mathbf{w}_y)$.

In [209], the greedy algorithm selects the pairs of features from $\mathbf{X}$ and $\mathbf{Y}$ such that the correlation between $\mathbf{X}$ and $\mathbf{Y}$ after projection is maximized. At each stage, this greedy algorithm selects two index sets I and J using a forward and stepwise subset selection method, where I and J correspond to the chosen indices in $\mathbf{X}$ and $\mathbf{Y}$, respectively. Initially, I and J are set to be the empty set. At each iteration, the algorithm selects one index from $\mathbf{X}$ or $\mathbf{Y}$ such that the resulting correlation after projection is maximized. The procedure is terminated until the required number of features have been selected at each stage.

In [200, 201, 210], the projection vectors $(\mathbf{w}_x, \mathbf{w}_y)$ are computed based on the minimum distance formulation of CCA using an alternating algorithm. At each iteration, one of $\mathbf{w}_x$ and

$\mathbf{w}_y$ is fixed while the other one is optimized. Note that in this case, the minimum distance formulation reduces to the least squares problem. Sparse solutions can be obtained by incorporating the ℓ_1-norm regularization. This procedure is repeated until convergence, resulting in the projection pair $(\mathbf{w}_x, \mathbf{w}_y)$ at that stage.

In [198, 202, 207], an iterative sparse algorithm is developed by estimating the singular vectors of $\mathbf{X}\mathbf{Y}^T$. At each step, the singular vectors of $\mathbf{X}\mathbf{Y}^T$ are estimated using soft-thresholding. In [207], the sparse CCA is further extended to analyze multiple data sets simultaneously, and a technique called sparse supervised CCA is proposed to integrate different views as well as some label information. Note that many other extensions of CCA can be derived based on Algorithm 4. For example, the nonnegative CCA algorithms have been proposed by imposing the nonnegativity constraint in the greedy framework to compute the projection pairs $(\mathbf{w}_x, \mathbf{w}_y)$ [207, 215].

Sparse CCA via Bayesian Learning

The third approach is based on the probabilistic interpretation of CCA [197, 199, 211]. Following the probabilistic interpretation of CCA [199], a nonparametric, fully Bayesian framework is proposed to capture the sparsity underlying the projections [197]. The sparse CCA framework proposed in [197] can automatically select the number of correlation components using the Indian Buffet Process (IBP) [216]. In addition, this framework can be applied for semi-supervised dimensionality reduction by making use of partial labels.

3.4 Relationship between CCA and Orthonormalized PLS

CCA is closely related orthonormalized PLS (OPLS). We show that the difference between the CCA solution and the OPLS solution is a mere orthogonal transformation. Unlike the discussions in [145], which focuses on discrimination only, the analysis here can be applied to regression and dimensionality reduction as well. We further extend the equivalence relationship to the case when regularization is applied for both sets of variables.

In the following discussion, we use the subscript *cca* and *pls* to distinguish the variables associated with CCA and OPLS, respectively. We first define two key matrices as follows:

$$\mathbf{H}_{cca} = \mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-\frac{1}{2}} \in \mathbb{R}^{n\times k}, \tag{3.24}$$

$$\mathbf{H}_{pls} = \mathbf{Y}^T \in \mathbb{R}^{n\times k}. \tag{3.25}$$

We assume that $\mathbf{Y}$ has full row rank, i.e., $\text{rank}(\mathbf{Y}) = k$. Thus, $(\mathbf{Y}\mathbf{Y}^T)^{-\frac{1}{2}}$ is well-defined. It follows from Section 2.2 and Section 3.2 that the solutions to both CCA and OPLS can be expressed as the eigenvectors corresponding to the top eigenvalues of the following matrix:

$$(\mathbf{X}\mathbf{X}^T)^{\dagger}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T), \tag{3.26}$$

where $\mathbf{H} = \mathbf{H}_{cca}$ for CCA and $\mathbf{H} = \mathbf{H}_{pls}$ for OPLS. We next derive the solution to this eigenvalue problem.

First we derive the principal eigenvectors of the generalized eigenvalue problem in Eq. (3.26). Let the SVD of $\mathbf{X}$ be

$$\mathbf{X} \quad = \quad \mathbf{U}\Sigma\mathbf{V}^T = \mathbf{U}_1\Sigma_1\mathbf{V}_1^T, \tag{3.27}$$

where $\mathbf{U} \in \mathbb{R}^{d\times d}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are orthogonal, $\mathbf{U}_1 \in \mathbb{R}^{d\times r}$ and $\mathbf{V}_1 \in \mathbb{R}^{n\times r}$ have orthonormal columns, $\Sigma \in \mathbb{R}^{d\times n}$ and $\Sigma_1 \in \mathbb{R}^{r\times r}$ are diagonal, and $r = \text{rank}(\mathbf{X})$. Denote

$$\mathbf{A} = \Sigma_1^{-1}\mathbf{U}_1^T\mathbf{X}\mathbf{H} = \Sigma_1^{-1}\mathbf{U}_1^T\mathbf{U}_1\Sigma_1\mathbf{V}_1^T\mathbf{H} = \mathbf{V}_1^T\mathbf{H} \in \mathbb{R}^{r\times k}, \tag{3.28}$$

and let the SVD of $\mathbf{A}$ be $\mathbf{A} = \mathbf{P}\Sigma_A\mathbf{Q}^T$, where $\mathbf{P} \in \mathbb{R}^{r\times r}$ and $\mathbf{Q} \in \mathbb{R}^{k\times k}$ are orthogonal and $\Sigma_A \in \mathbb{R}^{r\times k}$ is diagonal. Then we have

$$\mathbf{A}\mathbf{A}^T = \mathbf{P}\Sigma_A\Sigma_A^T\mathbf{P}^T. \tag{3.29}$$

Lemma 3.1 *The eigenvectors corresponding to the top ℓ eigenvalues of $(\boldsymbol{X}\boldsymbol{X}^T)^{\dagger}(\boldsymbol{X}\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{X}^T)$ are given by*

$$\boldsymbol{W} = \boldsymbol{U}_1\Sigma_1^{-1}\boldsymbol{P}_\ell, \tag{3.30}$$

where $\boldsymbol{P}_\ell$ consists of the first ℓ ($\ell \leq \text{rank}(\mathbf{A})$) columns of $\boldsymbol{P}$.

Proof We can decompose $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ as follows:

$$\begin{aligned}
(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T) &= \mathbf{U}_1\Sigma_1^{-2}\mathbf{U}_1^T\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T \\
&= \mathbf{U}_1\Sigma_1^{-1}A\mathbf{H}^T\mathbf{V}_1\Sigma_1\mathbf{U}_1^T \\
&= \mathbf{U}\begin{bmatrix}\mathbf{I}_r \\ 0\end{bmatrix}\Sigma_1^{-1}\mathbf{A}\mathbf{A}^T\Sigma_1\begin{bmatrix}\mathbf{I}_r & 0\end{bmatrix}\mathbf{U}^T \\
&= \mathbf{U}\begin{bmatrix}\Sigma_1^{-1}\mathbf{A}\mathbf{A}^T\Sigma_1 & 0 \\ 0 & 0\end{bmatrix}\mathbf{U}^T \\
&= \mathbf{U}\begin{bmatrix}\Sigma_1^{-1}\mathbf{P} & 0 \\ 0 & \mathbf{I}\end{bmatrix}\begin{bmatrix}\Sigma_A\Sigma_A^T & 0 \\ 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{P}^T\Sigma_1 & 0 \\ 0 & I\end{bmatrix}\mathbf{U}^T,
\end{aligned}$$

where the last equality follows from Eq. (3.29). It is clear that the eigenvectors corresponding to the top ℓ eigenvalues of $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ are given by

$$\mathbf{W} = \mathbf{U}_1\Sigma_1^{-1}\mathbf{P}_\ell.$$

This completes the proof of the lemma. ∎

It follows from Lemma 3.1 that $\mathbf{U}_1$ and Σ_1 are determined by $\mathbf{X}$. Thus the only difference between the projections computed by CCA and OPLS lies in $\mathbf{P}_\ell$. The following lemma reveals the property of $\mathbf{P}_\ell$:

Lemma 3.2 *Let $\boldsymbol{H}_{cca}$ and $\boldsymbol{H}_{pls}$ be defined in Eqs. (3.24) and (3.25). Let $\mathbf{A}_{cca} = \mathbf{V}_1^T\boldsymbol{H}_{cca}$ and $\mathbf{A}_{pls} = \mathbf{V}_1^T\boldsymbol{H}_{pls}$ be the matrix $\mathbf{A}$ defined in Eq. (5.12) for CCA and OPLS, respectively. Then the range spaces of $\mathbf{A}_{cca}$ and $\mathbf{A}_{pls}$ are the same.*

Proof Let the SVD of $\mathbf{Y}$ be

$$\mathbf{Y} = \mathbf{U}_y\Sigma_y\mathbf{V}_y^T, \tag{3.31}$$

where $\mathbf{U}_y \in \mathbb{R}^{k\times k}$, $\mathbf{V}_y \in \mathbb{R}^{n\times k}$, and $\Sigma_y \in \mathbb{R}^{k\times k}$ is diagonal. Since $\mathbf{Y}$ is assumed to have full column rank, all the diagonal elements of Σ_y are positive. Thus,

$$\begin{aligned}
\mathbf{A}_{cca} &= \mathbf{V}_1^T\mathbf{H}_{cca} = \mathbf{V}_1^T\mathbf{V}_y\mathbf{U}_y^T \\
\mathbf{A}_{pls} &= \mathbf{V}_1^T\mathbf{H}_{pls} = \mathbf{V}_1^T\mathbf{V}_y\Sigma_y\mathbf{U}_y^T.
\end{aligned}$$

It follows that $\mathbf{A}_{cca} = \mathbf{A}_{pls}\mathbf{U}_y\Sigma_y^{-1}\mathbf{U}_y^T$ and $\mathbf{A}_{pls} = \mathbf{A}_{cca}\mathbf{U}_y\Sigma_y\mathbf{U}_y^T$. Thus, the range spaces of $\mathbf{A}_{cca}$ and $\mathbf{A}_{pls}$ are the same. ∎

With this lemma, we can explicate the relationship between the projections computed by CCA and OPLS, as summarized in the following theorem:

Theorem 3.1 *Let the SVD of* $\mathbf{A}_{cca}$ *and* $\mathbf{A}_{pls}$ *be*

$$\begin{aligned} \mathbf{A}_{cca} &= \boldsymbol{P}_{cca}\Sigma_{\mathbf{A}_{cca}}\boldsymbol{Q}_{cca}^T, \\ \mathbf{A}_{pls} &= \boldsymbol{P}_{pls}\Sigma_{\mathbf{A}_{pls}}\boldsymbol{Q}_{pls}^T, \end{aligned}$$

where $\boldsymbol{P}_{cca}, \boldsymbol{P}_{pls} \in \mathbb{R}^{r\times r_A}$, *and* $r_A = \text{rank}(\mathbf{A}_{cca}) = \text{rank}(\mathbf{A}_{pls})$. *Then there exists an orthogonal matrix* $\boldsymbol{R} \in \mathbb{R}^{r_A\times r_A}$ *such that* $\boldsymbol{P}_{cca} = \boldsymbol{P}_{pls}R$.

Proof It is clear that $\mathbf{P}_{cca}\mathbf{P}_{cca}^T$ and $\mathbf{P}_{pls}\mathbf{P}_{pls}^T$ are the orthogonal projections onto the range spaces of $\mathbf{A}_{cca}$ and $\mathbf{A}_{pls}$, respectively. It follows from lemma 3.2 that both $\mathbf{P}_{cca}\mathbf{P}_{cca}^T$ and $\mathbf{P}_{pls}\mathbf{P}_{pls}^T$ are orthogonal projections onto the same subspace. Since the orthogonal projection onto a subspace is unique [172], we have

$$\mathbf{P}_{cca}\mathbf{P}_{cca}^T = \mathbf{P}_{pls}\mathbf{P}_{pls}^T. \tag{3.32}$$

Therefore,

$$\mathbf{P}_{cca} = \mathbf{P}_{cca}\mathbf{P}_{cca}^T\mathbf{P}_{cca} = \mathbf{P}_{pls}\mathbf{P}_{pls}^T\mathbf{P}_{cca} = \mathbf{P}_{pls}\mathbf{R},$$

where $\mathbf{R} = \mathbf{P}_{pls}^T\mathbf{P}_{cca} \in \mathbb{R}^{r_A\times r_A}$. It is easy to verify that $\mathbf{R}$ is orthogonal. ∎

If we retain all the eigenvectors corresponding to the nonzero eigenvalues, i.e., $\ell = r_A$, the difference between CCA and OPLS lies in the orthogonal transformation $\mathbf{R} \in \mathbb{R}^{r_A\times r_A}$. In this case, CCA and OPLS are essentially equivalent, since an orthogonal transformation preserves all pairwise distances.

The Equivalence Relationship with Regularization

The equivalence relationship still holds when the regularization is employed. We consider the regularization on $\mathbf{X}$ and $\mathbf{Y}$ separately.

Regularization on $\mathbf{X}$

Recall that regularized CCA (rCCA) and regularized OPLS (rOPLS) compute the principal eigenvectors of the following matrix:

$$(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T). \tag{3.33}$$

Lemma 3.3 *Define the matrix* $\boldsymbol{B} \in \mathbb{R}^{r\times k}$ *as*

$$\boldsymbol{B} = (\Sigma_1^2 + \gamma\boldsymbol{I})^{-1/2}\Sigma_1\mathbf{V}_1^T\boldsymbol{H} \tag{3.34}$$

and denote its SVD as $\boldsymbol{B} = \boldsymbol{P}_B\Sigma_B\boldsymbol{Q}_B^T$*, where* $\boldsymbol{P}_B \in \mathbb{R}^{r\times r}$ *and* $\boldsymbol{Q}_B \in \mathbb{R}^{k\times k}$ *are orthogonal, and* $\Sigma_B \in \mathbb{R}^{r\times k}$ *is diagonal. Then the eigenvectors corresponding to the top* ℓ *eigenvalues of matrix* $(\boldsymbol{X}\boldsymbol{X}^T + \gamma\boldsymbol{I})^{-1}(\boldsymbol{X}\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{X}^T)$ *are given by*

$$\boldsymbol{W} = \boldsymbol{U}_1(\Sigma_1^2 + \gamma\boldsymbol{I})^{-1/2}\boldsymbol{P}_{B\ell}, \tag{3.35}$$

where $\boldsymbol{P}_{B\ell}$ *consists of the first* ℓ $(\ell \le rank(\boldsymbol{B}))$ *columns of* $\boldsymbol{P}_B$*.*

Proof We can decompose $(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ as follows:

$$\begin{aligned}
& (\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T) \\
= \;& \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1}\Sigma_1\mathbf{V}_1^T\mathbf{H}\mathbf{H}^T\mathbf{V}_1\Sigma_1\mathbf{U}_1^T \\
= \;& \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\Sigma_1\mathbf{V}_1^T\mathbf{H}\mathbf{H}^T \\
& \mathbf{V}_1\Sigma_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}(\Sigma_1^2 + \gamma\mathbf{I})^{1/2}\mathbf{U}_1^T \\
= \;& \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}\mathbf{B}^T(\Sigma_1^2 + \gamma\mathbf{I})^{1/2}\mathbf{U}_1^T
\end{aligned}$$

$$\begin{aligned}
= \;& \mathbf{U}\begin{bmatrix}\mathbf{I}_r \\ 0\end{bmatrix}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}\mathbf{B}^T(\Sigma_1^2 + \gamma\mathbf{I})^{1/2}\begin{bmatrix}\mathbf{I}_r & 0\end{bmatrix}\mathbf{U}^T \\
= \;& \mathbf{U}\begin{bmatrix}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}\mathbf{B}^T(\Sigma_1^2 + \gamma\mathbf{I})^{1/2} & 0 \\ 0 & 0\end{bmatrix}\mathbf{U}^T \\
= \;& \mathbf{U}\begin{bmatrix}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{P}_B & 0 \\ 0 & \mathbf{I}\end{bmatrix}\begin{bmatrix}\Sigma_B\Sigma_B^T & 0 \\ 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{P}_B^T(\Sigma_1^2 + \gamma\mathbf{I})^{1/2} & 0 \\ 0 & \mathbf{I}\end{bmatrix}\mathbf{U}^T.
\end{aligned}$$

Thus, the eigenvectors corresponding to the top ℓ eigenvalues of

$$(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$$

are given by $\mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{P}_{B\ell}$. ∎

Using Lemma 3.3 we can show that the equivalence relationship between CCA and OPLS still holds when the regularization on $\mathbf{X}$ is applied. The main results are summarized in Lemma 3.4 and Theorem 3.2 below (proofs are similar to the ones in Lemma 3.2 and Theorem 3.1 and are omitted).

Lemma 3.4 *Let $\boldsymbol{B}_{cca}$ and $\boldsymbol{B}_{pls}$ be defined as follows:*

$$\begin{aligned}\boldsymbol{B}_{cca} &= (\Sigma_1^2 + \gamma\boldsymbol{I})^{-1/2}\Sigma_1\mathbf{V}_1^T\boldsymbol{H}_{cca} \\ \boldsymbol{B}_{pls} &= (\Sigma_1^2 + \gamma\boldsymbol{I})^{-1/2}\Sigma_1\mathbf{V}_1^T\boldsymbol{H}_{pls}.\end{aligned}$$

Then the range spaces of $\boldsymbol{B}_{cca}$ and $\boldsymbol{B}_{pls}$ are the same.

Theorem 3.2 *Let the SVD of $\boldsymbol{B}_{cca}$ and $\boldsymbol{B}_{pls}$ be*

$$\begin{aligned}\boldsymbol{B}_{cca} &= \boldsymbol{P}_{cca}^B\Sigma_{cca}^B(\boldsymbol{Q}_{cca}^B)^T, \\ \boldsymbol{B}_{pls} &= \boldsymbol{P}_{pls}^B\Sigma_{pls}^B(\boldsymbol{Q}_{Bpls}^B)^T,\end{aligned}$$

where $\boldsymbol{P}_{cca}^B, \boldsymbol{P}_{pls}^B \in \mathbb{R}^{r\times r_B}$, and $r_B = \text{rank}(\boldsymbol{B}_{cca}) = \text{rank}(\boldsymbol{B}_{pls})$. Then there exists an orthogonal matrix $\boldsymbol{R}^B \in \mathbb{R}^{r_B\times r_B}$ such that $\boldsymbol{P}_{cca}^B = \boldsymbol{P}_{pls}^B\boldsymbol{R}^B$.

Regularization on $\mathbf{Y}$

With the regularization applied on $\mathbf{Y}\mathbf{Y}^T$, we consider the eigendecomposition of following matrix:

$$(\mathbf{X}\mathbf{X}^T)^\dagger\mathbf{X}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T + \gamma\mathbf{I})^{-1}\mathbf{Y}\mathbf{X}^T. \tag{3.36}$$

The above formulation corresponds to a new matrix $\mathbf{H}_{rcca}$ for rCCA:

$$\mathbf{H}_{rcca} = \mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T + \gamma\mathbf{I})^{-1/2}. \tag{3.37}$$

We establish the equivalence relationship between CCA and OPLS when the regularization is applied on $\mathbf{Y}$.

Lemma 3.5 *Let $\boldsymbol{H}_{cca}, \boldsymbol{H}_{pls}$, and $\boldsymbol{H}_{rcca}$ be defined as in Eqs. (3.24), (3.25), and (3.37), respectively. Then the range spaces of $\boldsymbol{H}_{cca}$, $\boldsymbol{H}_{pls}$, and $\boldsymbol{H}_{rcca}$ are the same.*

Proof The proof follows directly from the definitions. ∎

Lemma 3.5 shows that the regularization on $\mathbf{Y}$ does not change the range space of $\mathbf{A}_{cca}$. Thus, the equivalence relationship between CCA and OPLS still holds. Similarly, the regularization on $\mathbf{Y}$ does not change the range space of $\mathbf{B}_{cca}$ when a regularization on $\mathbf{X}$ is applied. Therefore, the established equivalence relationship holds when the regularization is applied on both $\mathbf{X}$ and $\mathbf{Y}$.

Analysis of Regularization on CCA

Regularization is commonly employed to control the complexity of a learning model and it has been applied in various machine learning algorithms such as Support Vector Machines (SVM) [62]. In particular, regularization is crucial to kernel CCA [194] in order to avoid the trivial solution. Moreover, the use of regularization in CCA has natural statistical interpretations [103].

The established equivalence relationship between CCA and OPLS provides novel insights into the effect of regularization in CCA. In addition, it leads to a significant reduction in the computations involved in CCA. Regularization is commonly applied on both views in CCA [192, 194], since it is commonly believed that the CCA solution is dependent on both regularization terms. It follows from Lemma 3.5 that the range space of $\mathbf{H}_{rcca}$ is invariant to the regularization parameter γ_y. Thus, the range spaces of $\mathbf{A}_{cca}$ and $\mathbf{A}_{pls}$ are the same and the projection for $\mathbf{X}$ computed by rCCA is independent of γ_y. Similarly, we can show that the projection for $\mathbf{Y}$ is independent of the regularization on $\mathbf{X}$. Therefore, an important consequence from the equivalence relationship between CCA and OPLS is that the projection of CCA for one view is independent of the regularization on the other view.

Recall that the CCA formulation reduces to a generalized eigenvalue problem. In this formulation, we need to compute the inverse of the matrix $\mathbf{Y}\mathbf{Y}^T \in \mathbb{R}^{k \times k}$, which may cause numerical problems. Moreover, the dimensionality k of the data in $\mathbf{Y}$ may be large, and thus computing the inverse can be computationally expensive. For example, in the content-based

image retrieval [15], the two views correspond to text and image data that are both of high-dimensionality. Another important consequence of the established equivalence relationship between CCA and OPLS is that if only the projection for one view is required and the other view is only used to guide the projection on this view, then the inverse of a possibly large matrix can be effectively avoided.

3.5 Applications of Canonical Correlation Analysis

Canonical correlation analysis and its extensions have been used widely in a variety of areas [15, 194, 207, 215]. Here we describe some new applications of CCA.

As a classical tool to learn the underlying semantics from different views [193], CCA can be used to learn semantics of multimedia content by combining image and text data together. In [15, 194], the image data and text data are considered as two views, and the underlying semantics is learned to enable the retrieval of images from a text query but without reference to any labeling associated with the image. It is shown that kernel CCA can successfully retrieves the appropriate images given the text input [15, 194].

Canonical correlation analysis and many of its variants have gained popularity in the areas of bioinformatics and medical research [196, 198, 200–202, 207]. In particular, CCA and its various extensions are widely used in the genomic data analysis, in which multiple assays on the same set of patient samples becomes more common. Some typical examples include DNA copy number (or comparative genomic hybridization, CGH), gene expression, and single nucleotide polymorphism (SNP) data. Thus, CCA is an ideal tool to perform such task. So far, some variants of sparse CCA have been used to identify genes with expression correlated with other data. For example, [198, 200, 201] identifies gene with expression correlated with regions of DNA copy number change, [202] identifies genes that have expression correlated with SNPs, and [196] identifies sets of genes on two different microarray platforms that have correlated expression.

3.6 The Generalized Eigenvalue Problem

Recall that CCA reduces to a generalized eigenvalue problem as in Eq. (3.12). In fact, several other algorithms in dimensionality reduction can also be formulated as generalized eigenvalue problems [204], including linear discriminant analysis, partial least squares, and hypergraph

spectral learning, etc. In this section, we investigate a class of generalized eigenvalue problems encompassing these algorithms. In addition, we discuss its relationship with some cost functions.

Formally, we are interested in the generalized eigenvalue problem in the following form:

$$\mathbf{A}\mathbf{w} = \lambda \mathbf{B}\mathbf{w}, \tag{3.38}$$

where $\mathbf{A} \in \mathbb{R}^{d \times d}$ and $\mathbf{B} \in \mathbb{R}^{d \times d}$ are symmetric matrices. The generalized eigenvalue problem arises in many scenarios [137], and many algorithms have been proposed to solve it in the literature [137, 172]. In CCA, we have $\mathbf{A} = \mathbf{X}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}\mathbf{Y}\mathbf{X}^T$ and $\mathbf{B} = \mathbf{X}\mathbf{X}^T$ for computing the projection on $\mathbf{X}$.

The Generalized Rayleigh Quotient Cost Function

An important family of cost functions in machine learning and pattern recognition is the so-called generalized Rayleigh quotient [204]:

$$f(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{A}\mathbf{w}}{\mathbf{w}^T\mathbf{B}\mathbf{w}}. \tag{3.39}$$

For example, the objective function in linear discriminant analysis [166] is in the form of the generalized Rayleigh quotient. It turns out that the generalized eigenvalue problems in Eq. (3.38) provides an efficient way to optimize the generalized Rayleigh quotient cost function.

In the following, we analyze the connection between the generalized eigenvalue problem in Eq. (3.38) and the cost function $f(\mathbf{w})$ in Eq. (3.39). It is clear that the scaling of $\mathbf{w}$ does not affect the objective function $f(\mathbf{w})$. Hence, the optimization problem can be expressed in the following form:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \mathbf{w}^T\mathbf{A}\mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T\mathbf{B}\mathbf{w} = 1. \end{aligned} \tag{3.40}$$

Applying the Lagrange multiplier technique gives rise to

$$L = \mathbf{w}^T\mathbf{A}\mathbf{w} - \frac{\lambda}{2}(\mathbf{w}^T\mathbf{B}\mathbf{w} - 1).$$

Taking the derivative with respect to $\mathbf{w}$ and setting it to zero, we obtain

$$\mathbf{A}\mathbf{w} - \lambda\mathbf{B}\mathbf{w} = 0$$

$$\Leftrightarrow \quad \mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}.$$

Note that the maximum value obtained by the original optimization problem in Eq. (3.39) is the largest eigenvalue λ in the generalized eigenvalue problem in Eq. (3.38).

The discussion above can be extended to the more general case when more than one eigenpairs of the generalized eigenvalue problem in Eq. (3.38) are of interest. In this case, these eigenpairs correspond to the following cost function:

$$f(\mathbf{W}) = \frac{\mathrm{Tr}(\mathbf{W}^T\mathbf{A}\mathbf{W})}{\mathrm{Tr}(\mathbf{W}^T\mathbf{B}\mathbf{W})}, \tag{3.41}$$

where each column of $\mathbf{W}$ corresponds to an eigenvector of the generalized eigenvalue problem in Eq. (3.38).

Properties of the Generalized Eigenvalue Problem

It is well-known that the eigenvalues of the symmetric real eigenvalue problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$ are real, and the resulting eigenvectors are real and orthogonal. However, the orthogonality of eigenvectors generally does not hold for the generalized eigenvalue problem in Eq. (3.38) even when both $\mathbf{A}$ and $\mathbf{B}$ are real and symmetric. Interestingly, the so-called generalized orthogonality properties hold, which are summarized in the following theorem:

Theorem 3.3 *Let the ith eigenpair of the generalized eigenvalue problem* $\mathbf{A}\boldsymbol{w} = \lambda\boldsymbol{B}\boldsymbol{w}$ *be* $(\lambda_i, \boldsymbol{w}_i)$. *If* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *is symmetric and and* $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ *is symmetric and positive definite, then the following generalized orthogonality properties hold:*

$$\boldsymbol{w}_i^T\mathbf{A}\boldsymbol{w}_j = \delta_{ij}\lambda_i \tag{3.42}$$

$$\boldsymbol{w}_i^T\boldsymbol{B}\boldsymbol{w}_j = \delta_{ij}, \tag{3.43}$$

where $\delta_{ij} = 1$ *if* $i = j$ *and* 0 *otherwise.*

Proof Since $\mathbf{B} \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, then $\mathbf{B}^{1/2}$ is well-defined and non-singular. Define $\mathbf{u} = \mathbf{B}^{1/2}\mathbf{w}$, then the generalized eigenvalue problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$ can be trans-

formed into a symmetric eigenvalue problem:

$$\mathbf{B}^{-1/2}\mathbf{A}\mathbf{B}^{-1/2}\mathbf{u} = \lambda\mathbf{u}.$$

Note that the ith eigenpair of this symmetric eigenvalue problem can be denoted as $(\lambda_i, \mathbf{u}_i)$ where $\mathbf{u}_i = \mathbf{B}^{1/2}\mathbf{w}_i$. It follows from the properties of symmetric eigenvalue problem that

$$\mathbf{u}_i^T\mathbf{u}_j = \delta_{ij}.$$

Therefore, for $i \neq j$, we have

$$0 = \mathbf{u}_i^T\mathbf{u}_j = \mathbf{w}_i^T\mathbf{B}^{1/2}\mathbf{B}^{1/2}\mathbf{w}_j = \mathbf{w}_i^T\mathbf{B}\mathbf{w}_j = \frac{1}{\lambda_j}\mathbf{w}_i^T\mathbf{A}\mathbf{w}_j.$$

Next we consider the case that $i = j$:

$$\lambda_i = \lambda_i\mathbf{u}_i^T\mathbf{u}_i = \lambda_i\mathbf{w}_i^T\mathbf{B}^{1/2}\mathbf{B}^{1/2}\mathbf{w}_i = \lambda_i\mathbf{w}_i^T\mathbf{B}\mathbf{w}_i = \mathbf{w}_i^T\mathbf{A}\mathbf{w}_i.$$

This completes the proof. ∎

In the case that $\mathbf{B}$ is singular, an alternative formulation used frequently in machine learning is

$$\mathbf{B}^\dagger\mathbf{A}\mathbf{w} = \lambda\mathbf{w}, \tag{3.44}$$

where $\mathbf{B}^\dagger$ is the pseudoinverse of $\mathbf{B}$. The connection between the eigenvalue problem in Eq. (3.44) and the generalized eigenvalue problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$ is summarized in the following theorem:

Theorem 3.4 *Let $\mathbf{A} \in \mathbb{S}^{d\times d}$ and $\boldsymbol{B} \in \mathbb{S}^{d\times d}$, and assume that $\mathscr{R}(\mathbf{A}) \subseteq \mathscr{R}(\boldsymbol{B})$, where $\mathscr{R}(\mathbf{A})$ and $\mathscr{R}(\boldsymbol{B})$ are range spaces of $\mathbf{A}$ and $\boldsymbol{B}$, respectively. If $(\lambda, \boldsymbol{w})\,(\lambda \neq 0)$ is an eigenpair of the eigenvalue problem in Eq.* (3.44)*, then $(\lambda, \boldsymbol{w})$ is also the eigenpair of the generalized eigenvalue problem $\mathbf{A}\boldsymbol{w} = \lambda\boldsymbol{B}\boldsymbol{w}$. Conversely, if $(\lambda, \boldsymbol{w})\,(\lambda \neq 0)$ is an eigenpair of the generalized eigenvalue problem $\mathbf{A}\boldsymbol{w} = \lambda\boldsymbol{B}\boldsymbol{w}$, then $(\lambda, \boldsymbol{B}\boldsymbol{B}^\dagger\boldsymbol{w})$ is also the eigenpair of the eigenvalue problem in Eq.* (3.44)*.*

Proof First, we show that $\mathbf{A} = \mathbf{B}\mathbf{B}^\dagger\mathbf{A}$. Let the compact SVD of $\mathbf{B}$ be $\mathbf{B} = \mathbf{U}_B\Sigma_B\mathbf{V}_B$, where $\mathbf{U}_B \in \mathbb{R}^{d\times r_B}$, $\mathbf{V}_B \in \mathbb{R}^{d\times r_B}$, $\Sigma_B \in \mathbb{R}^{r_B\times r_B}$ is diagonal and $r_B = \text{rank}(\mathbf{B})$. Thus, we have $\mathscr{R}(\mathbf{B}) =$

$\mathscr{R}(\mathbf{U}_B)$. Since $\mathscr{R}(\mathbf{A}) \subseteq \mathscr{R}(\mathbf{B}) = \mathscr{R}(\mathbf{U}_B)$, there exists a matrix $\mathbf{R} \in \mathbb{R}^{r_B \times d}$ such that $\mathbf{A} = \mathbf{U}_B\mathbf{R}$. Thus, we have

$$\begin{aligned}
\mathbf{B}\mathbf{B}^\dagger\mathbf{A} &= \mathbf{B}\mathbf{B}^\dagger\mathbf{U}_B\mathbf{R} \\
&= \mathbf{U}_B\Sigma_B\mathbf{V}_B^T\mathbf{V}_B\Sigma_B^{-1}\mathbf{U}_B^T\mathbf{U}_B\mathbf{R} \\
&= \mathbf{U}_B\mathbf{R} \\
&= \mathbf{A}.
\end{aligned}$$

Note that we assume that $\mathbf{A}$ and $\mathbf{B}$ are symmetric, then we have $\mathbf{A} = \mathbf{B}\mathbf{B}^\dagger\mathbf{A} = \mathbf{A}\mathbf{B}^\dagger\mathbf{B}$.

Suppose $(\lambda, \mathbf{w})(\lambda \neq 0)$ is an eigenpair of the eigenvalue problem $\mathbf{B}^\dagger\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$, then we have

$$\begin{aligned}
& \mathbf{B}^\dagger\mathbf{A}\mathbf{w} = \lambda\mathbf{w} \\
\Leftrightarrow\ & \mathbf{V}_B\Sigma_B^{-1}\mathbf{U}_B^T\mathbf{U}_B\mathbf{R}\mathbf{w} = \lambda\mathbf{w} \\
\Rightarrow\ & \mathbf{R}\mathbf{w} = \lambda\Sigma_B\mathbf{V}_B^T\mathbf{w} \\
\Rightarrow\ & \mathbf{U}_B\mathbf{R}\mathbf{w} = \lambda\mathbf{U}_B\Sigma_B\mathbf{V}^T\mathbf{w} \\
\Leftrightarrow\ & \mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w},
\end{aligned}$$

which implies that $(\lambda, \mathbf{w})$ is also the eigenpair of the generalized eigenvalue problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$.

Next we prove the second part of this theorem. Suppose $(\lambda, \mathbf{w})(\lambda \neq 0)$ is an eigenpair of the generalized eigenvalue problem $\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w}$. Then we have

$$\begin{aligned}
& \mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w} \\
\Rightarrow\ & \mathbf{A}\mathbf{B}^\dagger\mathbf{B}\mathbf{w} = \lambda\mathbf{B}\mathbf{w} \\
\Rightarrow\ & \mathbf{B}^\dagger\mathbf{A}\left(\mathbf{B}^\dagger\mathbf{B}\mathbf{w}\right) = \lambda\mathbf{B}^\dagger\mathbf{B}\mathbf{w},
\end{aligned}$$

which implies that $(\lambda, \mathbf{B}\mathbf{B}^\dagger\mathbf{w})$ is also the eigenpair of the eigenvalue problem $\mathbf{B}^\dagger\mathbf{A}\mathbf{w} = \lambda\mathbf{w}$. Note that in the derivation we use the fact that $\mathbf{A} = \mathbf{A}\mathbf{B}^\dagger\mathbf{B}$. This completes the proof of this theorem. ∎

Algorithms for the Generalized Eigenvalue Problem

Many algorithms have been developed to solve the generalized eigenvalue problem in Eq. (3.38) [217]. A natural method to solve the generalized eigenvalue problem when $\mathbf{B}$ is nonsingular is to transform it into the following eigenvalue problem:

$$\mathbf{B}^{-1}\mathbf{A}\mathbf{w} = \lambda\mathbf{w}. \tag{3.45}$$

However, this approach suffers from several limitations. First, the eigenvalues computed by Eq. (3.45) could deviate from true eigenvalues significantly when $\mathbf{B}^{-1}\mathbf{A}$ is ill-conditioned. In addition, $\mathbf{B}^{-1}\mathbf{A}$ is usually not symmetric, thus many efficient methods for symmetric eigenvalue problems cannot be applied [172]. In addition, when $\mathbf{A}$ and $\mathbf{B}$ are sparse, such sparsity is not preserved in $\mathbf{B}^{-1}\mathbf{A}$, since the inverse of a sparse matrix may not be sparse.

In numerical linear algebra, the Cholesky decomposition is commonly applied to solve the symmetric generalized eigenvalue problem by transforming it into a symmetric eigenvalue problem [137, 217]. Assume that both $\mathbf{A}$ and $\mathbf{B}$ are symmetric and $\mathbf{B}$ is positive definite. In this case, the pair $(\mathbf{A}, \mathbf{B})$ is called a symmetric pair. As a result, we can obtain the Cholesky decomposition of $\mathbf{B}$ as

$$\mathbf{B} = \mathbf{L}\mathbf{L}^T, \tag{3.46}$$

where $\mathbf{L}$ is a lower triangular matrix and all its diagonal entries are positive. Denote $\mathbf{v} = \mathbf{L}^T\mathbf{w}$. Then $\mathbf{w} = \mathbf{L}^{-T}\mathbf{v}$. The generalized eigenvalue problem can be transformed into the following eigenvalue problem:

$$\mathbf{A}\mathbf{w} = \lambda\mathbf{B}\mathbf{w} \Leftrightarrow \mathbf{A}\mathbf{L}^{-T}\mathbf{v} = \lambda\mathbf{L}\mathbf{v} \Leftrightarrow \mathbf{L}^{-1}\mathbf{A}\mathbf{L}^{-T}\mathbf{v} = \lambda\mathbf{v}. \tag{3.47}$$

One benefit of the Cholesky decomposition is that the resulting eigenvalue problem in Eq. (3.47) is symmetric. Thus, efficient algorithms for symmetric eigenvalue problem, such as the Lanczos method [172, 217], can be applied. In addition, it is shown that the eigenvalues for the original generalized eigenvalue problem are real based on Eq. (3.47). However, the eigenvalues may not be real when $\mathbf{B}$ is not positive definite [137].

In order to further improve the efficiency, some algorithms based on the incomplete Cholesky decomposition have been proposed to solve CCA and kernel CCA recently [103, 194].

The key idea of the incomplete decomposition is to find a lower triangular matrix $\tilde{\mathbf{L}} \in \mathbb{R}^{d \times \tilde{d}} (\tilde{d} < d)$ for small $\tilde{d}$ such that

$$\mathbf{B} \approx \tilde{\mathbf{L}} \tilde{\mathbf{L}}^T. \tag{3.48}$$

In other words, the norm of the difference $\mathbf{B} - \tilde{\mathbf{L}} \tilde{\mathbf{L}}^T$ is less than some threshold. In terms of the implementation, the pivots below a certain threshold are skipped in comparison with the standard Cholesky decomposition.

Although the incomplete Cholesky decomposition provides an efficient method for solving the generalized eigenvalue problem, it cannot scale to large-size data sets and only approximate solution is obtained. It turns out that a class of dimensionality reduction techniques, including CCA, in the form of the generalized eigenvalue problem in Eq. (3.38), can be transformed equivalently into a least squares problem [138, 139]. As a result, efficient algorithms for solving the least squares, such as the conjugate gradient algorithm [172], can be applied. We postpone the discussion of the equivalence relationship and the resulting efficient implementations in Chapter 4 an Chapter 5.

Chapter 4

HYPERGRAPH SPECTRAL LEARNING

4.1 Introduction

One of the challenges of multi-label learning is how to effectively utilize the correlation among different labels in classification. In this chapter, we employ hypergraph [132] to capture the correlation among different labels for improved classification performance. A hypergraph is a generalization of the traditional graph in which the edges are arbitrary non-empty subsets of the vertex set. It has been applied for domains where higher-order relations such as co-authorship exist [132, 218]. We employ hypergraphs to exploit the higher-order relations among multiple instances sharing the same label in multi-label learning. Specifically, we construct a hyperedge for each label, and include all instances annotated with a common label into one hyperedge, thus capturing their joint similarity. Following the spectral graph embedding theory [135], we compute the lower-dimensional embedding through a linear transformation, which preserves the instance-label relations captured by the hypergraph. The projection is guided by the label information encoded in the hypergraph.

The resulting hypergraph learning formulation amounts to solving a generalized eigenvalue problem, which may be computationally expensive for large-scale problems. It turns out that a class of dimensionality reduction algorithms in machine learning can be formulated as generalized eigenvalue problems. Such algorithms include Canonical Correlation Analysis (CCA) [129], Orthonormalized Partial Least Squares (OPLS) [97], and Linear Discriminant Analysis (LDA) [166]. Although well-established algorithms in numerical linear algebra have been developed to solve generalized eigenvalue problems, they are in general computationally expensive and hence may not scale to large-scale machine learning problems. In addition, it is challenging to directly incorporate the sparsity constraint into the mathematical formulation of these algorithms. Sparsity often leads to easy interpretation and a good generalization ability. It has been used successfully in several algorithms including linear regression [124], and Principal Component Analysis (PCA) [125].

Multivariate Linear Regression (MLR) that minimizes the sum-of-squares error function, called least squares, is a classical technique for regression problems. It can also be applied

for classification problems by defining an appropriate class indicator matrix [173]. The solution to the least squares problem can be obtained by solving a linear system of equations, and a number of algorithms, including the conjugate gradient algorithm, can be applied to solve it efficiently [172]. Furthermore, the least squares formulation can be readily extended using the regularization technique. For example, the ℓ_1-norm regularization can be incorporated into the least squares formulation to improve sparsity and control model complexity [173].

Motivated by the mathematical and numerical properties of the generalized eigenvalue problem and the least squares formulation, several researchers have attempted to connect these two approaches. In particular, it has been shown that there is close relationship between LDA, CCA, and least squares [79, 138, 173, 219, 220]. However, the intrinsic relationship between least squares and other algorithms involving generalized eigenvalue problems mentioned above remains unclear. In this chapter, we also study the relationship between the least squares formulation and a class of dimensionality reduction algorithms in machine learning. In particular, we establish the equivalence relationship between these two formulations under a mild condition. As a result, various regularization techniques such as the ℓ_1-norm and ℓ_2-norm regularization can be readily incorporated into the formulation to improve model sparsity and generalization ability. In addition, this equivalence relationship leads to efficient and scalable implementations for these generalized eigenvalue problems based on the iterative conjugate gradient algorithms such as LSQR [221]. An incremental implementation for this class of dimensionality reduction algorithms is also proposed based on the equivalent least squares formulation.

We have conducted experiments using some benchmark data sets, and our results demonstrate the effectiveness of the proposed hypergraph spectral learning formulation for multi-label classification. The experiments on benchmark data sets confirm the equivalence relationship between the generalized eigenvalue problem and the corresponding least squares formulation for all dimensionality reduction algorithms addressed in this chapter under the given assumption. Our results show that even when the assumption does not hold, the performance of these two models is still very close. Results also demonstrate the efficiency and effectiveness of the equivalent least squares formulation and its extensions.

The rest of this chapter is organized as follows. We review the basics of hypergraph and hypergraph Laplacian in Section 4.2, and we present our multi-label learning formulation based on hypergraph in Section 4.3. The class of dimensionality reduction algorithms are reviewed in Section 4.4, and the equivalence relationship between the generalized eigenvalue problem and the corresponding least squares formulation is established in Section 4.5. The extensions based on the least squares formulation are discussed in Section 4.6. We present the efficient implementation for the class of dimensionality reduction algorithms in Section 4.7. The incremental implementation based on the least squares formation is discussed in Section 4.8. The experimental results are reported in Section 4.9, and this chapter concludes in Section 4.10.

4.2 Backgrounds

We review the basics of hypergraph in this section. In particular, we discuss three different approaches to defining the Laplacian for the hypergraph, including the star expansion, the clique expansion and some direct definitions of hypergraph Laplacian.

Basics of Hypergraph

A hypergraph [132–134] is a generalization of the traditional graph in which the edges, called hyperedges, are arbitrary non-empty subsets of the vertex set. In a hypergraph $G = (V, E)$, V is the vertex set and E is the edge set, where each $e \in E$ is a subset of V. A hyperedge e is said to be incident with a vertex v when $v \in e$. The degree of a hyperedge e, denoted as $\delta(e)$, is the number of vertices in e, i.e.,

$$\delta(e) = |e|. \tag{4.1}$$

The degree of every edge in a traditional graph is 2, and it is therefore called a "2-graph". The degree of a vertex $v \in V$, $d(v)$, is defined as

$$d(v) = \sum_{v \in e, e \in E} w(e), \tag{4.2}$$

where $w(e)$ is the weight associated with the hyperedge $e \in E$. The degree of an edge $\delta(e)$ is defined as

$$\delta(e) = |e|. \tag{4.3}$$

The diagonal matrix forms for $\delta(e)$, $d(v)$, $w(e)$ are denoted as $\mathbf{D}_e$, $\mathbf{D}_v$, $\mathbf{W}_H$, respectively. The vertex-edge incidence matrix $\mathbf{J} \in \mathbb{R}^{|V|\times|E|}$ is defined as

$$\mathbf{J}(v,e) = \begin{cases} 1 & \text{if } v \in e \\ 0 & \text{otherwise.} \end{cases} \tag{4.4}$$

Based on the above definitions, the following equations can be verified:

$$d(v) = \sum_{e\in E} w(e)\mathbf{J}(v,e), \tag{4.5}$$

$$\delta(e) = \sum_{v\in V} \mathbf{J}(v,e). \tag{4.6}$$

The adjacency matrix $\mathbf{A}$ is defined as

$$\mathbf{A} = \mathbf{J}\mathbf{W}_H\mathbf{J}^T - \mathbf{D}_v. \tag{4.7}$$

As the discrete analog of the Laplace-Beltrami operator on compact Riemannian manifolds [135], the graph Laplacian is extensively used in machine learning and data mining, especially in semi-supervised learning [222] and spectral clustering [223, 224]. Since the graph Laplacian matrix plays an important role in learning, a key issue of hypergraph is how to define the graph Laplacian matrix for the hypergraph.

The graph Laplacian has already been extended to higher-order structures [225]. Similar to traditional 2-graph, the Laplacian can be defined for hypergraph using the analogies from the Laplace-Beltrami operator on Riemannian manifolds. Formally, a function is called p-chain if it is defined for simplices with size p. For example, in traditional 2-graph $f : V \to \mathbb{R}$ is 0-chain, and $f : E \to \mathbb{R}$ is 1-chain. The general definition of the pth Laplacian operator on $p-$chains in hypergraph can be defined as:

$$L_p = \partial_{p+1}\partial_{p+1}^T + \partial_p^T\partial_p, \tag{4.8}$$

where ∂_p is the boundary operator that maps on p-chains to $(p-1)$-chains, and ∂_p^T is the coboundary operator that maps on $(p-1)$-chains to p-chains. In fact, L_p measures the change of functions defined on p-sized subsets of the vertex set, i.e. p-chains. Note that this definition is the same as the Laplace operator on $p-$forms on a Riemannian manifold [226]. When $p = 0$, the Laplacian defined in Eq. (4.8) reduces to the normally used Laplacian matrix in machine learning [222].

One drawback of the definition in Eq. (4.8) is that it cannot capture useful information in the learning process. It follows from the definition in Eq. (4.8) that L_p only considers p-chains and $p-1$ chains. However, in machine learning one fundamental goal is the learning of the vertex function, i.e., 0-chain. For example, in classification we are interested in the class label for each vertex, which is typically a 0-chain. Thus, $L_p(p>1)$ cannot give information for 0-chain, and may not be useful in practice.

In order to capture the underlying information from hypergraph for learning 0-chains, many hypergraph Laplacian matrices are proposed [132]. Specifically, one can either define hypergraph Laplacian directly using the analogies from traditional 2-graph or expand it into a 2-graph. It has been shown that the Laplacians defined in both ways are similar [132]. In addition, the eigenvectors of these Laplacians have been shown to be useful for learning higher-order relations, and that there is a close relationship between their hypergraph Laplacians and the structure of the hypergraph. In this section, we discuss two commonly used expansions as well as some hypergraph Laplacian defined directly.

The Clique Expansion

In graph theory, a clique in an undirected graph G is a set of vertices C such that for every two vertices in C, there exists an edge connecting the two. This is equivalent to saying that the subgraph induced by C is a complete graph. In clique expansion, each hyperedge is expanded into a clique. Denote by $G_c=(V_c,E_c)$ the 2-graph expanded from hypergraph $G=(V,E)$ using the clique expansion. We have $V_c=V$ and $E_c=\{(u,v):u\in e,v\in e,e\in E\}$. The edge weight $w_c(u,v)$ of G_c is given by

$$w_c(u,v)=\sum_{u,v\in e,e\in E} w(e). \tag{4.9}$$

Thus, the adjacency matrix in the clique expansion can be written in the following matrix form:

$$\mathbf{A}_c=\mathbf{J}\mathbf{W}_H\mathbf{J}^T. \tag{4.10}$$

The vertex degree in the expanded 2-graph $G_c=(V_c,E_c)$ can be expressed as

$$d_c(u)=\sum_{v\in V} w_c(u,v)=\sum_{e\in E}\mathbf{J}(u,e)(\delta(e)-1)w(e). \tag{4.11}$$

Let $\mathbf{D}_c$ be the diagonal matrix form for $d_c(u)$. Then the normalized Laplacian of G_c is given by

$$\mathscr{L}_c = \mathbf{D}_c^{-1/2}(\mathbf{D}_c - \mathbf{A}_c)\mathbf{D}_c^{-1/2} = \mathbf{I} - \mathbf{S}_c,$$

where $\mathbf{S}_c$ is defined as

$$\mathbf{S}_c = \mathbf{D}_c^{-1/2}\mathbf{A}_c\mathbf{D}_c^{-1/2} = \mathbf{D}_c^{-1/2}\mathbf{J}\mathbf{W}_H\mathbf{J}^T\mathbf{D}_c^{-1/2}. \tag{4.12}$$

Intuitively, the similarity between two vertices in clique expansion is proportional to the weights of common labels, thus capturing the instance-label relations.

The Star Expansion

In star expansion, a new vertex is introduced for each hyperedge and this new vertex is connected to each vertex in this hyperedge. Specifically, for a hypergraph $G = (V,E)$, the vertex and edge sets of the star-expanded 2-graph, denoted as V_* and E_*, are defined as $V_* = V \cup E$ and $E_* = \{(u,e) : u \in e, e \in E\}$, respectively. Thus, each hyperedge in G is expanded into a star in G_*, which is a bipartite graph.

The weight $w_*(u,e)$ of edge (u,e) in G_* is defined as

$$w_*(u,e) \stackrel{\triangle}{=} \frac{w(e)}{\delta(e)}, \tag{4.13}$$

where $w(e)$ and $\delta(e)$ are the weight and degree of hyperedge e, respectively. Since $V_* = V \cup E$, we can assume that in V_*, all $v \in V$ are ordered before $v \in E$. Then the adjacency matrix $\mathbf{A}_* \in \mathbb{R}^{(|V|+|E|)\times(|V|+|E|)}$ can be formulated as

$$\mathbf{A}_* = \begin{bmatrix} 0 & \mathbf{M} \\ \mathbf{M}^T & 0 \end{bmatrix}, \tag{4.14}$$

where $\mathbf{M} \in \mathbb{R}^{|V|\times|E|}$ and its (u,e) entry is given as follows:

$$\mathbf{M}(u,e) \stackrel{\triangle}{=} \frac{\mathbf{J}(u,e)w(e)}{\delta(e)}. \tag{4.15}$$

The degree of each vertex in the star-expanded graph G^* can be computed as follows:

$$d_*(u) = \sum_{e\in E} \frac{\mathbf{J}(u,e)w(e)}{\delta(e)}, \quad \text{if} \quad u \in V, \tag{4.16}$$

$$d_*(e) = \sum_{u\in e} \frac{w(e)}{\delta(e)} = w(e), \quad \text{if} \quad e \in E. \tag{4.17}$$

We use $\mathbf{D}_{*v}$ and $\mathbf{D}_{*e}$ to denote the diagonal matrices of vertex degrees for vertices in V and E, respectively. Thus, the normalized Laplacian for G_* can be expressed as follows:

$$\mathscr{L}_* = \mathbf{I} - \begin{bmatrix} \mathbf{D}_{*v} & 0 \\ 0 & \mathbf{D}_{*e} \end{bmatrix}^{-\frac{1}{2}} \mathbf{A}_* \begin{bmatrix} \mathbf{D}_{*v} & 0 \\ 0 & \mathbf{D}_{*e} \end{bmatrix}^{-\frac{1}{2}}. \tag{4.18}$$

By substituting Eq. (4.14) into Eq. (4.18), it can be shown that $\mathscr{L}_*$ can be simplified as

$$\mathscr{L}_* = \begin{bmatrix} \mathbf{I} & -\mathbf{B} \\ -\mathbf{B}^T & \mathbf{I} \end{bmatrix}, \tag{4.19}$$

where $\mathbf{B} = \mathbf{D}_{*v}^{-1/2}\mathbf{M}\mathbf{D}_{*e}^{-1/2}$, or its (u,e)th entry is

$$\mathbf{B}(u,e) = \frac{\mathbf{M}(u,e)}{\sqrt{d_*(u)}\sqrt{d_*(e)}} = \frac{\mathbf{J}(u,e)w(e)}{\sqrt{d_*(u)}\sqrt{d_*(e)}\delta(e)}. \tag{4.20}$$

Suppose $\mathbf{x}^T = [\mathbf{x}_v^T, \mathbf{x}_e^T]$ is an eigenvector of Laplacian $\mathscr{L}_*$ with corresponding eigenvalue γ, where $\mathscr{L}_*$ is the normalized Laplacian for the expanded graph, then we have

$$\begin{bmatrix} \mathbf{I} & -\mathbf{B} \\ -\mathbf{B}^T & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_e \end{bmatrix} = \gamma \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_e \end{bmatrix}.$$

This eigenvalue problem can be simplified as follows:

$$\begin{aligned} & \mathbf{x}_v - \mathbf{B}\mathbf{x}_e = \gamma\mathbf{x}_v, -\mathbf{B}^T\mathbf{x}_v + \mathbf{x}_e = \gamma x_e \\ \Leftrightarrow \quad & \mathbf{B}\mathbf{x}_e = (1-\gamma)\mathbf{x}_v, \mathbf{B}^T\mathbf{x}_v = (1-\gamma)\mathbf{x}_e \\ \Rightarrow \quad & \mathbf{B}\mathbf{B}^T\mathbf{x}_v = (\gamma-1)^2\mathbf{x}_v. \end{aligned} \tag{4.21}$$

In spectral learning, the eigenpair of the Laplacian $\mathscr{L}_*$ is essential for learning. Recall that only $\mathbf{x}_v$ corresponds to the real vertices in the original hypergraph while $\mathbf{x}_e$ corresponds to the constructed vertices, thus $\mathbf{x}_v$ and its corresponding eigenvalue are more important for learning. It follows from Eq. (4.21) that we can obtain the eigenpair $(\mathbf{x}_v, \gamma)$ from $\mathbf{B}\mathbf{B}^T$ and simplify the computation. We denote $\mathbf{S}_* = \mathbf{B}\mathbf{B}^T$ in the following discussion, and the equivalent matrix form is

$$\mathbf{S}_* \quad = \quad \mathbf{D}_{*v}{}^{-1/2}\mathbf{M}\mathbf{D}_{*e}{}^{-1}\mathbf{M}^T\mathbf{D}_{*v}{}^{-1/2}. \tag{4.22}$$

Intuitively, the constructed vertices connect to the vertices from the corresponding label in star expansion, capturing the interaction between the data points and the labels. Thus,

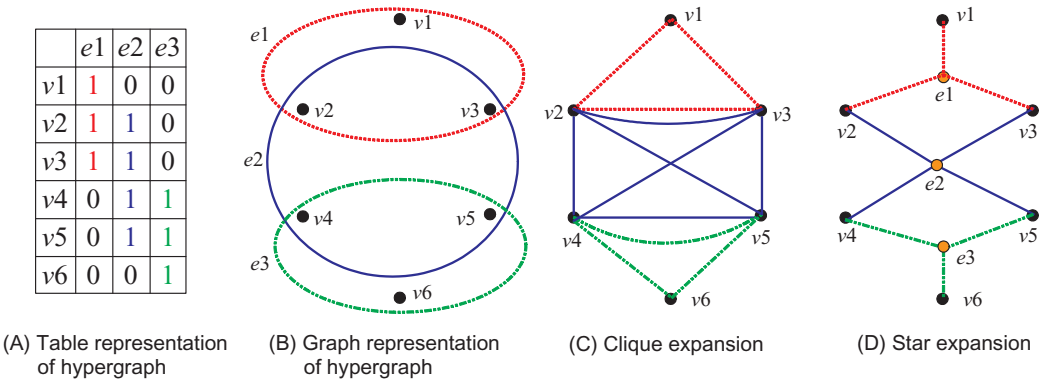


Figure 4.1: Illustration of a hypergraph and two hypergraph extensions (clique and star)

in multi-label learning the relationship among different labels can then be captured by their interactions with the data points. More details can be found in [132, 227].

A hypergraph example, including its table representation, graph representation, as well as the corespoinding clique expansion and star expansion, is demonstrated in Figure 4.1.

Hypergraph Laplacian

Several other definitions of hypergraph Laplacian have been proposed using analogies from the graph Laplacian for traditional 2-graph, such as Bolla's Laplacian [228], Rodriguez's Laplacian [229, 230] and Zhou's Laplacian [218]. In the following, we discuss Zhou's Laplacian due to its popularity and good performance [218, 231].

Following the random walk model, Zhou et al. [218] proposed the following normalized hypergraph Laplacian $\mathscr{L}_z$:

$$\mathscr{L}_z = \mathbf{I} - \mathbf{S}_z, \tag{4.23}$$

where

$$\mathbf{S}_z = \mathbf{D}_v^{-\frac{1}{2}} \mathbf{J} \mathbf{W}_H \mathbf{D}_e^{-1} \mathbf{J}^T \mathbf{D}_v^{-\frac{1}{2}}.$$

In the random walk model, given the current position $u \in V$, the walker first chooses a hyperedge e over all hyperedges incident with u with the probability proportional to $w(e)$, and then chooses a vertex $v \in e$ uniformly at random. Note that each label corresponds to a hyperedge in our framework. The transition probability between the nodes associated with two labels captures their similarity. More details can be found in [218].

We have discussed regression techniques in Chapter 2, including Ordinary Least Squares (OLS), Principal Component Regression (PCR), ridge regression and PLS regression. In this subsection, we discuss ordinary least squares, which is a classical technique for both regression and classification [173]. Given a set of observations $\mathbf{x}_i \in \mathbb{R}^d$ $(1 \leq i \leq n)$ and their corresponding targets $\mathbf{t}_i \in \mathbb{R}^k$ $(1 \leq i \leq n)$, the goal is to fit a linear model

$$\mathbf{t} = \mathbf{W}^T\mathbf{x} + \mathbf{b}$$

to the data, where $\mathbf{b} \in \mathbb{R}^k$ is the bias vector, and $\mathbf{W} \in \mathbb{R}^{d\times k}$ is the weight matrix. Assuming that both the observations $\{\mathbf{x}_i\}_{i=1}^n$ and the targets $\{\mathbf{t}_i\}_{i=1}^n$ are centered, the bias term becomes zero and can be ignored. The optimal $\mathbf{W}$ can be computed by minimizing the following sum-of-squares error function:

$$\sum_{i=1}^{n} \|\mathbf{W}^T\mathbf{x}_i - \mathbf{t}_i\|_2^2 = \|\mathbf{W}^T\mathbf{X} - \mathbf{T}\|_F^2, \tag{4.24}$$

where $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_n] \in \mathbb{R}^{k\times n}$. The optimal matrix $\mathbf{W}$ is

$$\mathbf{W}_{ls} = (\mathbf{X}\mathbf{X}^T)^{\dagger}\mathbf{X}\mathbf{T}^T. \tag{4.25}$$

When least squares is applied for classification, $\mathbf{T}$ is known as the class indicator matrix. Typically the 1-of-k binary coding scheme is applied: $\mathbf{T}_{ij} = 1$ if $\mathbf{t}_j$ belongs the the ith class or label, otherwise $\mathbf{T}_{ij} = 0$.

Least squares problems can be solved efficiently using existing techniques, which is very attractive for large-scale data analysis. Many iterative algorithms, including the conjugate gradient method, have been proposed in the literature [172, 221]. Compared with the direct methods, these iterative algorithms converge very fast for large and sparse problems.

4.3 Multi-label Learning with Hypergraph

In this section, we present our multi-label learning formulation based on hypergraph. We propose to learn from multi-label data by exploiting the spectral property of hypergraph that encodes the correlation information among labels. To capture the correlation among labels, we create a hyperedge for each label and include all data points relevant to this label into the

hyperedge. Based on the Laplacian of the constructed hypergraph, we propose the hypergraph spectral learning framework for learning a lower-dimensional embedding through a linear transformation $\mathbf{W}$ by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{W}} \quad & \mathrm{Tr}(\mathbf{W}^T\mathbf{X}\mathscr{L}\mathbf{X}^T\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W} = \mathbf{I}_k, \end{aligned} \tag{4.26}$$

where $\mathscr{L}$ is the normalized Laplacian of the hypergraph, and $\mathbf{W} \in \mathbb{R}^{d\times k}$ is the projection matrix. In this formulation, the objective function in Eq. (4.26) attempts to preserve the inherent relationship among data points captured by the Laplacian [232]. It follows from the traditional spectral graph embedding theory [135] that data points sharing many common labels tend to be close to each other in the embedded space.

Define the matrix $\mathbf{S}$ as

$$\mathbf{S} = \mathbf{I} - \mathscr{L}. \tag{4.27}$$

It follows from the property of the normalized Laplacian $\mathscr{L}$ that $\mathbf{S} \in \mathbb{R}^{n\times n}$ reflects the normalized similarities between different instances (or vertices). Hence, the original optimization problem in Eq. (4.26) can be reformulated equivalently into the following form:

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{Tr}(\mathbf{W}^T\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W} = \mathbf{I}_k. \end{aligned} \tag{4.28}$$

It can be verified that the optimal $\mathbf{W}^*$ to Eq. (4.28) consists of the eigenvectors corresponding to the largest eigenvalues of the following generalized eigenvalue problem:

$$\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{X}\mathbf{X}^T\mathbf{w}, \tag{4.29}$$

where $\gamma \in \mathbb{R}$ denotes the eigenvalue. In this chapter, we focus on the equivalent eigenvalue problem formulation:

$$(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{S}\mathbf{X}^T)\mathbf{w} = \lambda\mathbf{w}. \tag{4.30}$$

To avoid the singularity of $\mathbf{X}\mathbf{X}^T$, a regularization term $\gamma\mathbf{I}$ with $\gamma > 0$ is commonly added. This leads to the following regularized hypergraph spectral learning problem:

$$\left((\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{S}\mathbf{X}^T)\right)\mathbf{w} = \lambda\mathbf{w}. \tag{4.31}$$

Recall from Section 4.2 that different Laplacians can be constructed from the hypergraph that encodes the label information, resulting in different spectral learning algorithms.

4.4 A Class of Generalized Eigenvalue Problems

We have investigated the generalized eigenvalue problem in Chapter 3. In this section, we focus on a particular class of generalized eigenvalue problems in the following form:

$$\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{X}\mathbf{X}^T\mathbf{w}, \tag{4.32}$$

where $\mathbf{S} \in \mathbb{R}^{n\times n}$ is a symmetric and positive semi-definite matrix. In general, we are interested in the principal eigenvectors corresponding to nonzero eigenvalues. The generalized eigenvalue problem in Eq. (4.32) is often reformulated as the following eigenvalue problem:

$$(\mathbf{X}\mathbf{X}^T)^\dagger\mathbf{X}\mathbf{S}\mathbf{X}^T w = \lambda\mathbf{w}, \tag{4.33}$$

where $(\mathbf{X}\mathbf{X}^T)^\dagger$ is the Moore-Penrose pseudoinverse of $\mathbf{X}\mathbf{X}^T$. In addition, the generalized eigenvalue problem in Eq. (4.32) can also be formulated as the following optimization problem:

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{Tr}(\mathbf{W}^T\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T\mathbf{X}\mathbf{X}^T\mathbf{W} = \mathbf{I}. \end{aligned} \tag{4.34}$$

In the following derivation, we generally use the formulation in Eq.(4.33). Many existing dimensionality reduction algorithms exhibit the form in Eqs. (4.32) or (4.33). In particular, the matrix $\mathbf{S}$ can be represented in the following form in supervised learning:

$$\mathbf{S} = \mathbf{H}\mathbf{H}^T, \tag{4.35}$$

where $\mathbf{H} \in \mathbb{R}^{n\times k}$ is often constructed from the label information for the data. In order to control model complexity and avoid the singularity of $\mathbf{X}\mathbf{X}^T$, a regularization term $\gamma\mathbf{I}_d$ with $\gamma > 0$ is generally added to $\mathbf{X}\mathbf{X}^T$ in Eq. (4.32), leading to the following generalized eigenvalue problem:

$$\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{w} = \lambda(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I}_d)\mathbf{w}. \tag{4.36}$$

We discuss several dimensionality reduction algorithms involving the generalized eigenvalue problem in the form of Eq. (4.32). Specifically, they include Canonical Correlation Analysis (CCA), Orthonormalized Partial Least Squares (OPLS), Hypergraph Spectral Learning

(HSL), and Linear Discriminant Analysis (LDA). For supervised learning methods, the label information is encoded in the matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_n] \in \mathbb{R}^{k \times n}$, where $\mathbf{y}_i(j) = 1$ if $\mathbf{x}_i$ belongs to class j and $\mathbf{y}_i(j) = 0$ otherwise.

Canonical Correlation Analysis

Assume that $\mathbf{YY}^T$ is nonsingular in CCA, as discussed in Eq. (3.12) in Chapter 3. The projection vector $\mathbf{w}_x$ for $\mathbf{X}$ is the first principal eigenvector of the following generalized eigenvalue problem:

$$\mathbf{XY}^T(\mathbf{YY}^T)^{-1}\mathbf{YX}^T\mathbf{w}_x = \lambda \mathbf{XX}^T\mathbf{w}_x. \tag{4.37}$$

Multiple projection vectors can be obtained simultaneously by computing the first ℓ principal eigenvectors of the generalized eigenvalue problem in Eq. (4.37). It can be observed that CCA is in the form of the generalized eigenvalue problem in Eq. (4.32), and the matrices $\mathbf{S}$ and $\mathbf{H}$ are defined as

$$\mathbf{S} = \mathbf{Y}^T(\mathbf{YY}^T)^{-1}\mathbf{Y} \tag{4.38}$$

$$\mathbf{H} = \mathbf{Y}^T(\mathbf{YY}^T)^{-1/2}. \tag{4.39}$$

Orthonormalized Partial Least Squares

As we discussed in Eq. (2.14) in Chapter 2, the orthonormalized PLS involves the following generalized eigenvalue problem:

$$\mathbf{XY}^T\mathbf{YX}^T\mathbf{w} = \lambda \mathbf{XX}^T\mathbf{w}. \tag{4.40}$$

It follows from Eq. (4.40) that orthonormalized PLS involves a generalized eigenvalue problem in Eq. (4.32), and the matrices $\mathbf{S}$ and $\mathbf{H}$ are defined as

$$\mathbf{S} = \mathbf{Y}^T\mathbf{Y} \tag{4.41}$$

$$\mathbf{H} = \mathbf{Y}^T. \tag{4.42}$$

Hypergraph Spectral Learning

Given the normalized Laplacian $\mathscr{L}$ for the constructed hypergraph, HSL involves the following generalized eigenvalue problem:

$$\mathbf{XSX}^T\mathbf{w} = \lambda(\mathbf{XX}^T)\mathbf{w}, \text{ where } \mathbf{S} = \mathbf{I} - \mathscr{L}. \tag{4.43}$$

It turns out that the matrix $\mathbf{S}$ is symmetric and positive semi-definite for all three Laplacian matrices, which can be represented as $\mathbf{S} = \mathbf{H}\mathbf{H}^T$, where $\mathbf{H} \in \mathbb{R}^{n\times k}$. In all cases, the matrix $\mathbf{H}$ is constructed from the label information $\mathbf{Y}$ explicitly without using matrix decomposition. Specifically, the definitions of $\mathbf{H}$ for different Laplacian definitions are given as follows:

$$\mathbf{H}_* = \mathbf{D}_{*v}{}^{-1/2}\mathbf{M}\mathbf{D}_{*e}{}^{-1/2}, \tag{4.44}$$

$$\mathbf{H}_c = \mathbf{D}_c^{-1/2}\mathbf{J}\mathbf{W}_H^{1/2}, \tag{4.45}$$

$$\mathbf{H}_z = \mathbf{D}_v^{-\frac{1}{2}}\mathbf{J}\mathbf{W}_H^{1/2}\mathbf{D}_e^{-1/2}, \tag{4.46}$$

where the definitions $\mathbf{H}_*$, $\mathbf{H}_c$, and $\mathbf{H}_z$ can be found in the discussions of the star expansion, the clique expansion, and the Zhou's Laplacian, respectively.

Linear Discriminant Analysis

As a supervised dimensionality reduction algorithm, LDA attempts to minimize the within-class variance while maximizing the between-class variance after the linear projection. It has been shown that the optimal linear projection consists of the top eigenvectors of $\mathbf{S}_t^\dagger\mathbf{S}_b$ corresponding to nonzero eigenvalues [166], where $\mathbf{S}_t$ is the total covariance matrix and $\mathbf{S}_b$ is the between-class covariance matrix. Assuming that $\mathbf{X}$ is centered, i.e., $\sum_{i=1}^n \mathbf{x}_i = 0$, the matrices $\mathbf{S}_t$ and $\mathbf{S}_b$ can be defined as follows:

$$\mathbf{S}_t = \frac{1}{n}\mathbf{X}\mathbf{X}^T, \tag{4.47}$$

$$\mathbf{S}_b = \frac{1}{n}\sum_{j=1}^k n_j\bar{\mathbf{x}}_j\bar{\mathbf{x}}_j^T, \tag{4.48}$$

where $\bar{\mathbf{x}}_j$ is the centroid of the jth class. We also assume that the data matrix $\mathbf{X}$ is partitioned into k classes as $\mathbf{X} = [\mathbf{X}_1, \cdots, \mathbf{X}_k]$, where $\mathbf{X}_j \in \mathbb{R}^{d\times n_j}$ corresponds to the data points from the jth class, n_j is the size of the jth class, and $\sum_{j=1}^k n_j = n$. Note that $\bar{\mathbf{x}}_j = \frac{1}{n_j}\mathbf{X}_j 1_j$, where 1_j is a vector of all ones with length n_j. Thus, we have

$$\begin{aligned} n\mathbf{S}_b &= \sum_{j=1}^k \frac{1}{n_j}\mathbf{X}_j 1_j 1_j^T \mathbf{X}_j^T \\ &= \sum_{j=1}^k \mathbf{X}_j\left(\frac{1}{n_j}1_j 1_j^T\right)\mathbf{X}_j^T \\ &= \sum_{j=1}^k \mathbf{X}_j\mathbf{S}_j\mathbf{X}_j^T = \mathbf{X}\mathbf{S}\mathbf{X}^T, \end{aligned}$$

where $\mathbf{S}_j = \frac{1}{n_j} 1_j 1_j^T$, and $\mathbf{S}$ is defined as $\text{diag}\,(\mathbf{S}_1, \mathbf{S}_2, \cdots, \mathbf{S}_k)$ for LDA. Therefore, $\mathbf{S}_t^\dagger \mathbf{S}_b$ can also be expressed in the following form:

$$\mathbf{S}_t^\dagger \mathbf{S}_b \quad = \quad (\mathbf{X}\mathbf{X}^T)^\dagger (\mathbf{X}\mathbf{S}\mathbf{X}^T). \tag{4.49}$$

It can be verified that $\mathbf{S} = \mathbf{H}\mathbf{H}^T$, where

$$\mathbf{H} = \text{diag}\,(\frac{1}{\sqrt{n_1}} 1_1, \frac{1}{\sqrt{n_2}} 1_2, \cdots, \frac{1}{\sqrt{n_k}} 1_k) \in \mathbb{R}^{n \times k}. \tag{4.50}$$

4.5 Generalized Eigenvalue Problem versus the Least Squares Problem

In this section, we investigate the relationship between the generalized eigenvalue problem in Eq. (4.32) or its equivalent formulation in Eq. (4.33) and the least squares formulation. In particular, we show that under a mild condition[1], the eigenvalue problem in Eq. (4.33) can be formulated as a least squares problem with a specific target matrix.

Matrix Orthonormality Property

To simplify the discussion, we define matrices $\mathbf{C}_X$ and $\mathbf{C}_S$ as follows:

$$\mathbf{C}_X \quad = \quad \mathbf{X}\mathbf{X}^T \in \mathbb{R}^{d \times d}, \tag{4.51}$$

$$\mathbf{C}_S \quad = \quad \mathbf{X}\mathbf{S}\mathbf{X}^T \in \mathbb{R}^{d \times d}. \tag{4.52}$$

The eigenvalue problem in Eq. (4.33) can then be expressed as

$$\mathbf{C}_X^\dagger \mathbf{C}_S \mathbf{w} = \lambda \mathbf{w}. \tag{4.53}$$

Recall that $\mathbf{S}$ is symmetric and positive semi-definite, thus it can be decomposed as

$$\mathbf{S} = \mathbf{H}\mathbf{H}^T, \tag{4.54}$$

where $\mathbf{H} \in \mathbb{R}^{n \times s}$, and $s \leq n$. For all examples discussed in Section 4.4, the closed-form of $\mathbf{H}$ can be obtained and $s = k \ll n$.

[1] It states that $\{\mathbf{x}_i\}_{i=1}^n$ are linearly independent before centering, i.e., $\text{rank}(\mathbf{X}) = n - 1$ after the data is centered (of zero mean).

Since $\mathbf{X}$ is centered, i.e., $\mathbf{X}1 = 0$, we have $\mathbf{XC} = \mathbf{X}$, where $\mathbf{C} = \mathbf{I} - \frac{1}{n}11^T$ is the centering matrix satisfying $\mathbf{C}^T = \mathbf{C}$, and $\mathbf{C}1 = 0$. It follows that

$$\begin{aligned}
\mathbf{C}_S &= \mathbf{XSX}^T \\
&= (\mathbf{XC})\mathbf{S}(\mathbf{XC})^T \\
&= \mathbf{X}(\mathbf{CSC}^T)\mathbf{X}^T \\
&= \mathbf{X}(\mathbf{C}^T\mathbf{SC})\mathbf{X}^T \\
&= \mathbf{X}\tilde{\mathbf{S}}\mathbf{X}^T, \qquad (4.55)
\end{aligned}$$

where $\tilde{\mathbf{S}} = \mathbf{C}^T\mathbf{SC}$. Note that

$$1^T\tilde{\mathbf{S}}1 = 1^T\mathbf{C}^T\mathbf{SC}1 = 0. \qquad (4.56)$$

Thus, we can assume that $1^T\mathbf{S}1 = 0$, that is, both columns and rows of $\mathbf{S}$ are centered. Before presenting the main results, we have the following lemmas.

Lemma 4.1 *Assume that $1^T\mathbf{S}1 = 0$. Let $\boldsymbol{H}$ be defined in Eq. (4.54). Let $\boldsymbol{HP} = \boldsymbol{QR}$ be the QR decomposition of $\boldsymbol{H}$ with column pivoting, where $\boldsymbol{Q} \in \mathbb{R}^{n\times r}$ has orthonormal columns, $\boldsymbol{R} \in \mathbb{R}^{r\times k}$ is upper triangular, $r = \mathrm{rank}(\boldsymbol{H}) \leq k$, and $\boldsymbol{P} \in \mathbb{R}^{k\times k}$ is a permutation matrix. Then we have $\boldsymbol{Q}^T 1 = 0$.*

Proof Since $\mathbf{P}$ is a permutation matrix, we have $\mathbf{P}^T\mathbf{P} = \mathbf{PP}^T = \mathbf{I}_k$. Then $\mathbf{H}$ can be reformulated as follows:

$$\begin{aligned}
\mathbf{H} &= \mathbf{H}(\mathbf{PP}^T) \\
&= \mathbf{QRP}^T.
\end{aligned}$$

Hence, we have

$$\begin{aligned}
1^T\mathbf{S}1 &= 1^T\mathbf{HH}^T1 \\
&= 1^T\mathbf{QRP}^T\mathbf{PR}^T\mathbf{Q}^T1 \\
&= 1^T\mathbf{QRR}^T\mathbf{Q}^T1 \\
&= 0.
\end{aligned}$$

Note that $\mathbf{R}\mathbf{R}^T$ is positive definite, therefore, we can conclude that $\mathbf{Q}^T 1 = 0$. This completes the proof of this lemma. ∎

Lemma 4.2 *Let* $\mathbf{A} \in \mathbb{R}^{m\times(m-1)}$ *and* $\boldsymbol{B} \in \mathbb{R}^{m\times p}$ $(p \leq m)$ *be two matrices satisfying* $\mathbf{A}^T 1 = 0$, $\mathbf{A}^T\mathbf{A} = \boldsymbol{I}_{m-1}$, $\boldsymbol{B}^T\boldsymbol{B} = \boldsymbol{I}_p$, *and* $\boldsymbol{B}^T 1 = 0$. *Let* $\boldsymbol{F} = \mathbf{A}^T\boldsymbol{B}$. *Then* $\boldsymbol{F}^T\boldsymbol{F} = \boldsymbol{I}_p$.

Proof Since $\mathbf{A}^T 1 = 0$ and $\mathbf{A}^T\mathbf{A} = \mathbf{I}_{m-1}$, we can construct the orthogonal matrix $\mathbf{A}_x$ as follows:

$$\mathbf{A}_x = \left[\mathbf{A}, \frac{1}{\sqrt{m}}1\right] \in \mathbb{R}^{m\times m}. \tag{4.57}$$

Then we have

$$\mathbf{I}_m = \mathbf{A}_x\mathbf{A}_x^T = \mathbf{A}\mathbf{A}^T + \frac{1}{m}11^T \Leftrightarrow \mathbf{A}\mathbf{A}^T = \mathbf{I}_m - \frac{1}{m}11^T.$$

Since $\mathbf{B}^T 1 = 0$ and $\mathbf{B}^T\mathbf{B} = \mathbf{I}_p$, we obtain that

$$\begin{aligned}
\mathbf{F}^T\mathbf{F} &= \mathbf{B}^T\mathbf{A}\mathbf{A}^T\mathbf{B} \\
&= \mathbf{B}^T\left(\mathbf{I}_m - \frac{1}{m}11^T\right)\mathbf{B} \\
&= \mathbf{B}^T\mathbf{B} - \frac{1}{m}(\mathbf{B}^T 1)(\mathbf{B}^T 1)^T \\
&= \mathbf{I}_p.
\end{aligned}$$

This completes the proof of this lemma. ∎

Let $\mathbf{R} = \mathbf{U}_R\Sigma_R\mathbf{V}_R^T$ be the thin singular value decomposition (SVD) of $\mathbf{R} \in \mathbb{R}^{r\times k}$, where $\mathbf{U}_R \in \mathbb{R}^{r\times r}$ is orthogonal, $\mathbf{V}_R \in \mathbb{R}^{k\times r}$ has orthonormal columns, and $\Sigma_R \in \mathbb{R}^{r\times r}$ is diagonal. It follows that $(\mathbf{Q}\mathbf{U}_R)^T(\mathbf{Q}\mathbf{U}_R) = \mathbf{I}_r$, and the SVD of $\mathbf{S}$ can be derived as follows:

$$\begin{aligned}
\mathbf{S} &= \mathbf{H}\mathbf{H}^T \\
&= \mathbf{Q}\mathbf{R}\mathbf{R}^T\mathbf{Q}^T \\
&= \mathbf{Q}\mathbf{U}_R\Sigma_R^2\mathbf{U}_R^T\mathbf{Q}^T \\
&= (\mathbf{Q}\mathbf{U}_R)\Sigma_R^2(\mathbf{Q}\mathbf{U}_R)^T.
\end{aligned} \tag{4.58}$$

Assume that the columns of $\mathbf{X}$ are centered, i.e., $\mathbf{X}1 = 0$, and $\text{rank}(\mathbf{X}) = n - 1$. Let

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T = \mathbf{U}_1\Sigma_1\mathbf{V}_1^T$$

be the SVD of $\mathbf{X}$, where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, $\Sigma \in \mathbb{R}^{d\times n}$ is diagonal, $\mathbf{U}_1\Sigma_1\mathbf{V}_1^T$ is the compact SVD of $\mathbf{X}$, $\mathbf{U}_1 \in \mathbb{R}^{d\times(n-1)}$, $\mathbf{V}_1 \in \mathbb{R}^{n\times(n-1)}$, and $\Sigma_1 \in \mathbb{R}^{(n-1)\times(n-1)}$ is diagonal. Define $\mathbf{M}_1$ as

$$\mathbf{M}_1 = \mathbf{V}_1^T(\mathbf{Q}\mathbf{U}_R) \in \mathbb{R}^{(n-1)\times r}. \tag{4.59}$$

It can be shown that the columns of $\mathbf{M}_1$ are orthonormal as summarized in the following lemma:

Lemma 4.3 *Let $\boldsymbol{M}_1$ be defined as above. Then $\boldsymbol{M}_1^T\boldsymbol{M}_1 = \boldsymbol{I}_r$.*

Proof Since $\mathbf{X}1 = 0$, we have $\mathbf{V}_1^T 1 = 0$. Also note that $\mathbf{V}_1^T\mathbf{V}_1 = \mathbf{I}_{n-1}$ and $(\mathbf{Q}\mathbf{U}_R)^T(\mathbf{Q}\mathbf{U}_R) = \mathbf{I}_r$. It follows from Lemma 4.1 that

$$(\mathbf{Q}\mathbf{U}_R)^T 1 = \mathbf{U}_R^T\mathbf{Q}^T 1 = 0.$$

Then we have from Lemma 4.2 that $\mathbf{M}_1^T\mathbf{M}_1 = \mathbf{I}_r$. ∎

The Equivalence Relationship

We first derive the solution to the eigenvalue problem in Eq. (4.33) in the following theorem:

Theorem 4.1 *Let $\boldsymbol{U}_1$, Σ_1, $\mathbf{V}_1$, $\boldsymbol{Q}$, Σ_R, and $\boldsymbol{U}_R$ be defined as above. Assume that the columns of $\boldsymbol{X}$ are centered, i.e., $\boldsymbol{X}1 = 0$, and* $\mathrm{rank}(\boldsymbol{X}) = n-1$. *Then the nonzero eigenvalues of the eigenvalue problem in Eq. (4.33) are* $\mathrm{diag}\,(\Sigma_R^2)$*, and the corresponding eigenvectors are $\boldsymbol{W}_{eig} = \boldsymbol{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\boldsymbol{Q}\boldsymbol{U}_R$.*

Proof It follows from Lemma 4.3 that the columns of $\mathbf{M}_1 \in \mathbb{R}^{(n-1)\times r}$ are orthonormal. Hence, there exists $\mathbf{M}_2 \in \mathbb{R}^{(n-1)\times(n-1-r)}$ such that $\mathbf{M} = [\mathbf{M}_1, \mathbf{M}_2] \in \mathbb{R}^{(n-1)\times(n-1)}$ is orthogonal [172]. Then we can derive the eigen-decomposition of the matrix $\mathbf{C}_X^\dagger\mathbf{C}_S$ as follows:

$$\begin{aligned}
& \mathbf{C}_X^\dagger\mathbf{C}_S \\
= \;& \left(\mathbf{X}\mathbf{X}^T\right)^\dagger \mathbf{X}\mathbf{S}\mathbf{X}^T \\
= \;& (\mathbf{U}_1\Sigma_1^{-2}\mathbf{U}_1^T)\mathbf{U}_1\Sigma_1\mathbf{V}_1^T(\mathbf{Q}\mathbf{U}_R)\Sigma_R^2(\mathbf{Q}\mathbf{U}_R)^T\mathbf{V}_1\Sigma_1\mathbf{U}_1^T \\
= \;& \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_1^T(\mathbf{Q}\mathbf{U}_R)\Sigma_R^2(\mathbf{Q}\mathbf{U}_R)^T\mathbf{V}_1\Sigma_1\mathbf{U}_1^T \\
= \;& \mathbf{U}_1\Sigma_1^{-1}\mathbf{M}_1\Sigma_R^2\mathbf{M}_1^T\Sigma_1\mathbf{U}_1^T
\end{aligned}$$

$$
\begin{aligned}
&= \mathbf{U}\begin{bmatrix}\mathbf{I}_{n-1}\\ 0\end{bmatrix}\Sigma_1^{-1}\begin{bmatrix}\mathbf{M}_1 & \mathbf{M}_2\end{bmatrix}\begin{bmatrix}\Sigma_R^2 & 0\\ 0 & 0_{n-1-r}\end{bmatrix}\begin{bmatrix}\mathbf{M}_1^T\\ \mathbf{M}_2^T\end{bmatrix}\Sigma_1[\mathbf{I}_{n-1},0]\mathbf{U}^T\\
&= \mathbf{U}\begin{bmatrix}\mathbf{I}_{n-1}\\ 0\end{bmatrix}\Sigma_1^{-1}\mathbf{M}\begin{bmatrix}\Sigma_R^2 & 0\\ 0 & 0_{n-1-r}\end{bmatrix}\mathbf{M}^T\Sigma_1[\mathbf{I}_{n-1},0]\mathbf{U}^T\\
&= \mathbf{U}\begin{bmatrix}\Sigma_1^{-1}\mathbf{M} & \\ & 1\end{bmatrix}\begin{bmatrix}\Sigma_R^2 & \\ & 0_{n-r}\end{bmatrix}\begin{bmatrix}\mathbf{M}^T\Sigma_1 & \\ & 1\end{bmatrix}U^T. \qquad (4.60)
\end{aligned}
$$

There are r nonzero eigenvalues, which are $\mathrm{diag}(\Sigma_R^2)$, and the corresponding eigenvectors are

$$\mathbf{W}_{eig} = \mathbf{U}_1\Sigma_1^{-1}\mathbf{M}_1 = \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\mathbf{Q}\mathbf{U}_R. \qquad (4.61)$$

This completes the proof of the theorem. ∎

We summarize the main result of this section in the following theorem:

Theorem 4.2 *Assume that the class indicator matrix $\boldsymbol{T}$ for least squares classification is defined as*

$$\boldsymbol{T} = \boldsymbol{U}_R^T\boldsymbol{Q}^T \in \mathbb{R}^{r\times n}. \qquad (4.62)$$

Then the solution to the least squares formulation in Eq. (4.24) is given by

$$\boldsymbol{W}_{ls} = \boldsymbol{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\boldsymbol{Q}\boldsymbol{U}_R. \qquad (4.63)$$

Thus, the eigenvalue problem and the least squares problem are equivalent.

Proof When $\mathbf{T}$ is used as the class indicator matrix, then the solution to the least squares problem is

$$
\begin{aligned}
\mathbf{W}_{ls} &= (\mathbf{X}\mathbf{X}^T)^\dagger\mathbf{X}\mathbf{T}^T\\
&= (\mathbf{X}\mathbf{X}^T)^\dagger\mathbf{X}\mathbf{Q}\mathbf{U}_R\\
&= \mathbf{U}_1\Sigma_1^{-2}\mathbf{U}_1^T\mathbf{U}_1\Sigma_1\mathbf{V}_1^T\mathbf{Q}\mathbf{U}_R\\
&= \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\mathbf{Q}\mathbf{U}_R.
\end{aligned}
$$

It follows from Eq. (4.61) that $\mathbf{W}_{ls} = \mathbf{W}_{eig}$. This completes the proof of the theorem. ∎

Remark 4.1 *The analysis in [138, 219] is based on a key assumption that the* $\boldsymbol{H}$ *matrix in* $\mathbf{S} = \boldsymbol{H}\boldsymbol{H}^T$ *as defined in Eq. (4.54) has orthonormal columns, which is the case for LDA and CCA. However, this is in general not true, e.g., the* $\boldsymbol{H}$ *matrix in OPLS and HSL. The equivalence result established in this chapter significantly improves previous work by relaxing this assumption.*

The matrix $\mathbf{W}_{eig}$ is applied for dimensionality reduction (projection) for all examples of Eq. (4.32). The weight matrix $\mathbf{W}_{ls}$ in least squares can also be used for dimensionality reduction. If $\mathbf{T} = \mathbf{Q}^T$ is used as the class indicator matrix, the weight matrix becomes $\tilde{\mathbf{W}}_{ls} = \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\mathbf{Q}$. Thus, the difference between $\mathbf{W}_{eig}$ and $\tilde{\mathbf{W}}_{ls}$ is the orthogonal matrix $\mathbf{U}_R$. Note that the Euclidean distance is invariant to any orthogonal transformation. If a classifier, such as k-Nearest-Neighbor (kNN) and linear Support Vector Machines (SVM) [62] based on the Euclidean distance, is applied on the dimensionality-reduced data via $\mathbf{W}_{eig}$ and $\tilde{\mathbf{W}}_{ls}$, they will achieve the same classification performance.

In some cases, the number of nonzero eigenvalues, i.e. r, is large (comparable to n). It is common to use the top eigenvectors corresponding the largest $\ell < r$ eigenvalues as in PCA. From Theorems 4.1 and 4.2, if we keep the top ℓ singular vectors of $\mathbf{S}$ as the class indicator matrix, the equivalence relationship between the generalized eigenvalue problem and the least squares problem holds, as summarized below:

Corollary 4.1 *The top* $\ell < r$ *eigenvectors in Eq. (4.33) can be computed by solving a least squares problem with the top* ℓ *singular vectors of* $\mathbf{S}$ *employed as the class indicator matrix.*

4.6 Extensions

Based on the equivalence relationship established in Section 4.5, the original generalized eigenvalue problem in Eq. (4.32) can be extended using the regularization technique. Regularization is commonly used to control the complexity of the model and improve the generalization performance. As we discussed in Chapter 2, ridge regression [79] minimizes the penalized sum-of-squares error function by incorporating the ℓ_2-norm regularization. By using the target matrix $\mathbf{T}$ in Eq. (4.62), we obtain the ℓ_2-norm regularized least squares formulation by minimizing the

following objective function:

$$L_2(\mathbf{W}, \gamma) = \|\mathbf{W}^T\mathbf{X} - \mathbf{T}\|_F^2 + \gamma \sum_{j=1}^{k} \|\mathbf{w}_j\|_2^2, \tag{4.64}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_k]$ and $\gamma > 0$ is the regularization parameter. We can then apply the LSQR algorithm, a conjugate gradient method proposed in [221] for solving large-scale sparse least-squares problems. The complexity analysis is discussed in detail in the next section.

It is well-known that model sparsity can often be achieved by applying the ℓ_1-norm regularization [124, 208]. This has been introduced into the least squares formulation and the resulting model is called lasso [124]. Based on the established equivalence relationship between the original generalized eigenvalue problem and the least squares formulation, we derive the ℓ_1-norm regularized least squares formulation by minimizing the following objective function:

$$L_1 = \|\mathbf{W}^T\mathbf{X} - \mathbf{T}\|_F^2 + \gamma \sum_{j=1}^{k} \|\mathbf{w}_j\|_1, \tag{4.65}$$

for some tuning parameter $\gamma > 0$ [124]. The lasso can be solved efficiently using the state-of-the-art algorithms [212–214]. In addition, the entire solution path for all values of γ can be obtained by applying the least angle regression algorithm [233].

In the following discussion, the equivalent least squares formulation for all dimensionality reduction algorithms are named using a prefix "LS" such as "LS-Star", and the resulting ℓ_1-norm and ℓ_2-norm regularized formulations are named by adding subscripts 1 and 2, respectively, e.g., "LS-Star$_1$" and "LS-Star$_2$". For the original eigenvalue problem with regularization in Eq. (4.31), we name it using a prefix "r" before the corresponding method, e.g., "rStar".

4.7 Efficient Implementation via LSQR

Recall that we deal with the generalized eigenvalue problem in Eq. (4.32), although an equivalent eigenvalue problem in Eq. (4.33) is used in our theoretical derivation. Large-scale generalized eigenvalue problems are known to be much harder than regular eigenvalue problems [234]. There are two options to transform the problem in Eq. (4.32) into a standard eigenvalue problem [234]: (1) factor $\mathbf{X}\mathbf{X}^T$; and (2) employ the standard Lanczos algorithm for the matrix $(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{S}\mathbf{X}^T$ using the $\mathbf{X}\mathbf{X}^T$ inner product. The second option has its own issue for singular matrices, which is the case for high-dimensional problems with a small regularization. Thus, we factor $\mathbf{X}\mathbf{X}^T$ and solve a symmetric eigenvalue problem using the Lanczos algorithm.

Algorithm 5 Efficient Implementation of the GEP via LSQR

Input: $\mathbf{X}$, $\mathbf{H}$.
Output: $\mathbf{W}$.
Compute the QR decomposition of $\mathbf{H}$: $\mathbf{HP} = \mathbf{QR}$.
Compute the SVD of $\mathbf{R}$: $\mathbf{R} = \mathbf{U}_R\Sigma_R\mathbf{V}_R^T$.
Compute the class indicator matrix $\mathbf{T} = \mathbf{U}_R^T\mathbf{Q}^T$.
Compute $\mathbf{W}$ by regressing $\mathbf{X}$ on $\mathbf{T}$ using LSQR.

The equivalent least squares formulation leads to an efficient implementation. The pseudo-code of the algorithm is given in Algorithm 5. Next we analyze the time complexity of the Algorithm 5. The complexity of the QR decomposition in the first step is $O(nk^2)$. Note that k is the number of classes (or labels), and $k \ll n$. The SVD of $\mathbf{R}$ costs $O(k^3)$. In the last step, we solve k least squares problems. In our implementation, we use the LSQR algorithm proposed in [221], which is a conjugate gradient method for solving large-scale least squares problems. In practice, the original data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ may be sparse in many applications such as text document modeling. Note that the centering of $\mathbf{X}$ is necessary in some techniques such as CCA. However, $\mathbf{X}$ is no longer sparse after centering. In order to keep the sparsity of $\mathbf{X}$, we follow the method proposed in [220] to augment the vector $\mathbf{x}_i$ by an additional component as $\tilde{\mathbf{x}}_i^T = [1, \mathbf{x}_i^T]$, and the extended data matrix $\mathbf{X}$ is denoted as $\tilde{\mathbf{X}} \in \mathbb{R}^{(d+1)\times n}$. This new component acts as the bias for least squares. The revised least squares problem is formulated as

$$\min_{\tilde{\mathbf{W}}} \|\tilde{\mathbf{W}}^T\tilde{\mathbf{X}} - \mathbf{T}\|_F^2, \tag{4.66}$$

where $\tilde{\mathbf{W}} \in \mathbb{R}^{(d+1)\times k}$. For a new data point $\mathbf{x} \in \mathbb{R}^d$, its projection is given by $\tilde{\mathbf{W}}^T[1;\mathbf{x}]$.

For the dense data matrix, the overall computational cost of each iteration of LSQR is $O(3n+5d+2dn)$ [221]. Since the least squares problems are solved k times, the overall cost of LSQR is $O(Nk(3n+5d+2dn))$, where N is the total number of iterations. When the matrix $\tilde{\mathbf{X}}$ is sparse, the cost is significantly reduced. Let the number of nonzero elements in $\tilde{\mathbf{X}}$ be z, then the overall cost of LSQR is reduced to $O(Nk(3n+5d+2z))$. In summary, the total time complexity for solving the least squares formulation via LSQR is $O(nk^2+Nk(3n+5d+2z))$.

4.8 Incremental Dimensionality Reduction Algorithms

Most of dimensionality reduction algorithms work in a batch or offline mode, i.e., they assume that the whole data set is available in advance from the beginning. However, in many data

mining and machine learning tasks, the data accumulates over time. When the new data points arrive, one needs to rerun the entire algorithm with the updated data set, which is prohibitively expensive especially for large-scale problems. Hence, the batch algorithms are unsuitable for sequentially coming data. In this case, it is suitable to apply the incremental algorithms, or online algorithms to process the accumulated data and update the model sequentially. The online or incremental algorithms process the input piece-by-piece in a serial fashion since the entire training data set is not available in the beginning. The basic idea of online algorithms is to maintain a model which can be updated incrementally by the information conveyed by the new data points with low computational cost. Recently, some incremental dimensionality reduction algorithms have been proposed in the literature, such as incremental algorithms for LDA [235–239]. Unfortunately, most incremental algorithms for LDA only provide approximate solutions.

In this section, we present an incremental implementation for the same class of dimensionality reduction algorithms discussed in this chapter using the equivalence relationship between them and the corresponding least squares formulation. Under the assumption that all data points are linearly independent before centering, the solution to the generalized eigenvalue problem in Eq. (4.32) can be computed as follows using the target matrix $\mathbf{T}$ defined in Eq. (4.62):

$$\mathbf{W} = (\mathbf{X}\mathbf{X}^T)^\dagger \mathbf{X}\mathbf{T}^T = \mathbf{X}^\dagger \mathbf{T}^T. \tag{4.67}$$

Thus, in the incremental dimensionality reduction algorithms, the key is to update $\mathbf{X}^\dagger$ and $\mathbf{T}$ incrementally as new data point arrives sequentially. Compared with existing online dimensionality reduction algorithms, one advantage of the proposed implementation is that the exact solution can be obtained in each step.

Formally, assume that at the pth($1 \leq p \leq n$) step, the data point $(\mathbf{x}_p, \mathbf{y}_p)$ arrives. The data matrix and the label matrix are denoted as $\mathbf{X}_p = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_p]$ and $\mathbf{Y}_p = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_p]$, respectively. Throughout the algorithm, we assume that all the acquired data points are linearly independent, i.e., $\text{rank}(\mathbf{X}_p) = p$ before centering. In the following, we will discuss how to update $\mathbf{X}_p^\dagger$ and $\mathbf{Y}_p$ incrementally.

Update $X^\dagger$ Incrementally

Denote $\mathbf{X}_{p-1} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{p-1}] \in \mathbb{R}^{d\times(p-1)}$ at the $(p-1)$th step. When the new data point $\mathbf{x}_p$ arrives, the data matrix $\mathbf{X}_p$ is updated as

$$\mathbf{X}_p = [\mathbf{X}_{p-1}, \mathbf{x}_p] \in \mathbb{R}^{d\times p}. \tag{4.68}$$

Applying the method proposed in [240], we can update the pseudoinverse of $\mathbf{X}_p$ using the following formula:

$$\mathbf{X}_p^\dagger = \begin{bmatrix} \mathbf{X}_{p-1}^\dagger - \mathbf{d}_p\mathbf{b}_p \\ \mathbf{b}_p \end{bmatrix},$$

where $\mathbf{d}_p$, $\mathbf{c}_p$ and $\mathbf{b}_p$ are defined as follows:

$$\begin{aligned} \mathbf{d}_p &= \mathbf{X}_{p-1}^\dagger \mathbf{x}_p, \\ \mathbf{c}_p &= \mathbf{x}_p - \mathbf{X}_{p-1}\mathbf{d}_p, \\ \mathbf{b}_p &= \begin{cases} \mathbf{c}_p^\dagger, & \text{if } \mathbf{c}_p \neq 0 \\ \dfrac{\mathbf{d}_p^T \mathbf{X}_{p-1}^\dagger}{1+\mathbf{d}_p^T\mathbf{d}_p}, & \text{if } \mathbf{c}_p = 0. \end{cases} \end{aligned}$$

Note that we assume that $\text{rank}(\mathbf{X}_p) - \text{rank}(\mathbf{X}_{p-1}) = 1$, thus $\mathbf{b}_p = \mathbf{c}_p^\dagger = \frac{\mathbf{c}_p}{\mathbf{c}_p^T\mathbf{c}_p}$. Another issue is the centering of $\mathbf{X}_p$ using the current mean $\bar{\mathbf{x}}_p = \frac{1}{p}\sum_{i=1}^p \mathbf{x}_i$. In this case we maintain the mean $\bar{\mathbf{x}}_p$ at the pth step, and update $\mathbf{X}_p$ and $\mathbf{X}_p^\dagger$ accordingly.

Update $\boldsymbol{T}$ Incrementally

Denote $\mathbf{Y}_{p-1} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{p-1}] \in \mathbb{R}^{d\times(p-1)}$ at the $(p-1)$th step. When the new data $\mathbf{y}_p$ arrives, the data matrix $\mathbf{X}_p$ is updated as

$$\mathbf{Y}_p = \left[\mathbf{Y}_{p-1}, \mathbf{y}_p\right] \in \mathbb{R}^{k\times p}. \tag{4.69}$$

Note that $k \ll n$ and $k \ll d$ where k is the number of labels. Thus we can apply the procedures in Algorithm 5 to compute the updated target matrix $\mathbf{T}_p$ explicitly at the pth step.

Note that for some specific dimensionality reduction algorithm, e.g., LDA, the update of the target matrix can be performed more efficiently by utilizing special properties of the corresponding target matrix. More details can be found in [241].

Table 4.1: Summary of statistics of the data sets.

Data Set	n	d	k
Scene	2407	294	6
Yeast	2417	103	14
USPS	9298	256	10
Yahoo\Arts&Humanities	3712	23146	26
Yahoo\Computers&Internet	6270	34096	33

4.9 Experiments

In this section, we empirically evaluate the effectiveness and efficiency of hypergraph spectral learning. We also report experimental results that validate the established equivalence relationship for all dimensionality reduction algorithms exhibiting the form of the generalized eigenvalue problem in Eq. (4.32). The scalability of the proposed least squares extensions is also demonstrated.

Experimental Setup

These dimensionality reduction algorithms discussed in Section 4.4 can be divided into two categories: (1) CCA, PLS, and HSL for multi-label learning; and (2) LDA for multi-class learning. We use both multi-label and multi-class data sets in the experiments, including the Scene data set [4], the Yeast data set [28], the USPS data set [242], and two high-dimensional document data sets [21, 22]. The Scene, Yeast and USPS data sets and the two document data sets from Yahoo! are available online[2,3]. For all data sets, the labels that contain less than 50 instances are removed. We follow the feature selection methods studied in [122] for text documents and extract different numbers of terms to investigate the performance of algorithms. The statistics of these data sets are summarized in Table 4.1, where n is number of data points, d is the dimensionality, and k is the number of labels or classes.

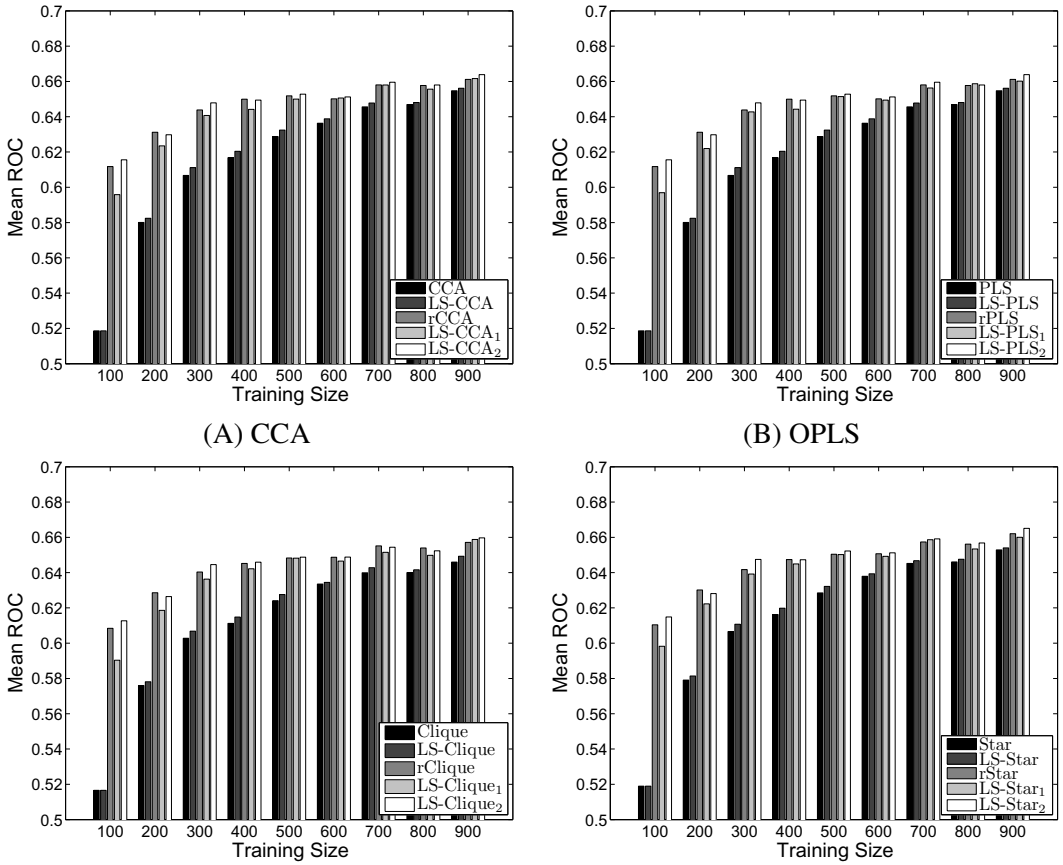
For each data set, a transformation matrix $\mathbf{W}$ is learned from the training set, and it is then used to project the test data onto a lower-dimensional space. The linear Support Vector Machine (SVM) is applied for each label separately for classification. The Receiver Operating

[2] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets`
[3] `http://www.kecl.ntt.co.jp/as/members/ueda/yahoo.tar.gz`

Characteristic (ROC) score and classification accuracy are used to evaluate the performance of multi-label and multi-class tasks, respectively. Ten random partitions of the data sets into training and test sets are generated in the experiments, and the mean performance over all labels and all partitions are reported. All experiments are performed on a PC with Intel Core 2 Duo T7200 2.0G CPU and 2G RAM. All algorithms in our experiments are implemented in Matlab.

Performance of Hypergraph Spectral Learning

In this section we investigate the performance of hypergraph spectral learning on multi-label data sets. Three different Laplacian matrices of the constructed hypergraph are studied, including clique expansion, star expansion and Zhou's Laplacian matrix. The performance of HSL is



(A) CCA (B) OPLS

(C) HSL-Clique (A) HSL-Star

Figure 4.2: Comparison of different formulations in terms of ROC scores for different algorithms on the Yeast data set. CCA, OPLS and HSL are applied on the Yeast data set. For regularized algorithms, the optimal value of γ is estimated from $\{1e-6, 1e-4, 1e-2, 1, 10, 100, 1000\}$ using cross validation.

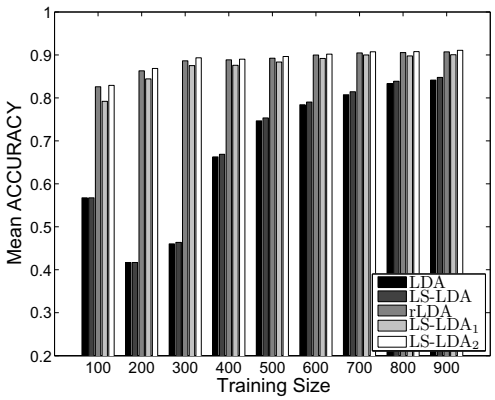


Figure 4.3: Comparison of different formulations in terms of accuracies for LDA on the USPS data set. For regularized algorithms, the optimal value of γ is estimated from $\{1e-6, 1e-4, 1e-2, 1, 10, 100, 1000\}$ using cross validation.

compared with CCA and Rank-SVM [28]. We also apply SVM to each label independently and our results show that it is less effective than Rank-SVM and is thus omitted in the table. For all algorithms with regularization, three-fold cross validation is applied to select the optimal value of the regularization parameter. All the features of the Scene and Yeast data sets are used while the most frequent 2000 terms of the Arts and Computers data sets are used. We report the mean ROC score over all labels and all partitions for each algorithm in Table 4.2.

Note that for all data sets and the training/test partitions, the assumption that $\text{rank}(\mathbf{X}) = n-1$ is not satisfied, thus the equivalence relationship between the generalized eigenvalue formulation in Eq. (4.32) and the corresponding least squares formulation does not hold. We can observe from Table 4.2 that the performance of the hypergraph spectral learning formulations and the corresponding least squares formulations is different, although very close in all cases. Moreover, regularized algorithms always outperform those without regularization, which justifies the use of regularization. Results in Table 4.2 show that the proposed methods with regularization perform better than the Rank-SVM algorithm. Results also show that three different Laplacian matrices lead to similar performance, which is consistent with the theoretical analysis in [132].

Table 4.2: Summary of mean ROC scores over all labels of HSL, CCA and RankSVM.

Algorithm	Scene	Yeast	Arts	Computers
Training size	900	900	2000	2000
Clique	0.8548	0.6460	0.5215	0.5369
rClique	0.9216	0.6496	0.8106	0.7968
LS-Clique	0.8598	0.6493	0.5213	0.5375
LS-Clique$_2$	**0.9255**	0.6523	**0.8130**	0.7996
Star	0.8546	0.6529	0.5209	0.5384
rStar	0.9216	0.6558	0.8051	0.7970
LS-Star	0.8597	0.6540	0.5209	0.5381
LS-Star$_2$	**0.9255**	0.6565	0.8067	0.7999
Zhou	0.8547	0.6518	0.5206	0.5389
rZhou	0.9216	0.6521	0.8074	0.7991
LS-Zhou	0.8597	0.6537	0.5205	0.5389
LS-Zhou$_2$	0.9254	0.6559	0.8087	**0.8025**
CCA	0.8538	0.6547	0.5213	0.5341
rCCA	0.9215	0.6555	0.8015	0.7917
LS-CCA	0.8586	0.6561	0.5215	0.5340
LS-CCA$_2$	**0.9255**	**0.6568**	0.8036	0.7952
Rank-SVM	0.9001	0.6294	0.8073	0.7893

Evaluation of the Equivalence Relationship

In this experiment, we show the equivalence relationship between the generalized eigenvalue problem and its corresponding least squares formulation for all dimensionality reduction algorithms discussed in Section 4.4, i.e., CCA, OPLS, HSL and LDA. We observe that for all data sets, when the data dimensionality d is larger than the sample size n, $\text{rank}(\mathbf{X}) = n-1$ is likely to hold. We also observe that the generalized eigenvalue problem and its corresponding least squares formulation achieve the same performance when $\text{rank}(\mathbf{X}) = n-1$ holds. These are consistent with the theoretical results in Theorems 4.1 and 4.2.

We compare the performance of the generalized eigenvalue problem and the corresponding least squares formulation when the assumption in Theorems 4.1 and 4.2 is violated. Figure 4.2 shows the mean ROC score over all labels of different formulations when the size of training set varies from 100 to 900 with a step size about 100 on the Yeast data set. The performance of CCA, OPLS and two variants of HSL is summarized in Figure 4.2. We can

observe from the figure that when n is small, the assumption in Theorem 4.1 holds and the two formulations achieve the same performance; when n is large, the assumption in Theorem 4.1 does not hold and the two formulations achieve different performance, although the difference is always very small in the experiment. We can also observe from Figure 4.2 that the regularized methods outperform the unregularized ones, which validates the effectiveness of regularization.

Similar experiment is performed for LDA on the multi-class USPS data, and its performance is summarized in Figure 4.3. From Figure 4.3 we can make the similar observations.

Evaluation of Scalability

In this experiment, we compare the scalability of the original generalized eigenvalue problem and the equivalent least squares formulation. Since regularization is commonly employed in practice, we compare the regularized version of the generalized eigenvalue problem and its corresponding ℓ_2-norm regularized least squares formulation. The least squares problem is solved by the LSQR algorithm [221]. We solve the original generalized eigenvalue problem by factoring $\mathbf{X}\mathbf{X}^T$ and solving a symmetric eigenvalue problem using the Lanczos method [234].

The computation time of the two formulations on the high-dimensional multi-label Yahoo! data set is shown in Figure 4.4, where the data dimensionality increases and the training sample size is fixed at 1000. All dimensionality reduction algorithms applid for multi-label data are tesetd. It can be observed that the computation time for both algorithms increases steadily as the data dimensionality increases. However, the computation time of the least squares formulation is substantially less than that of the original one.

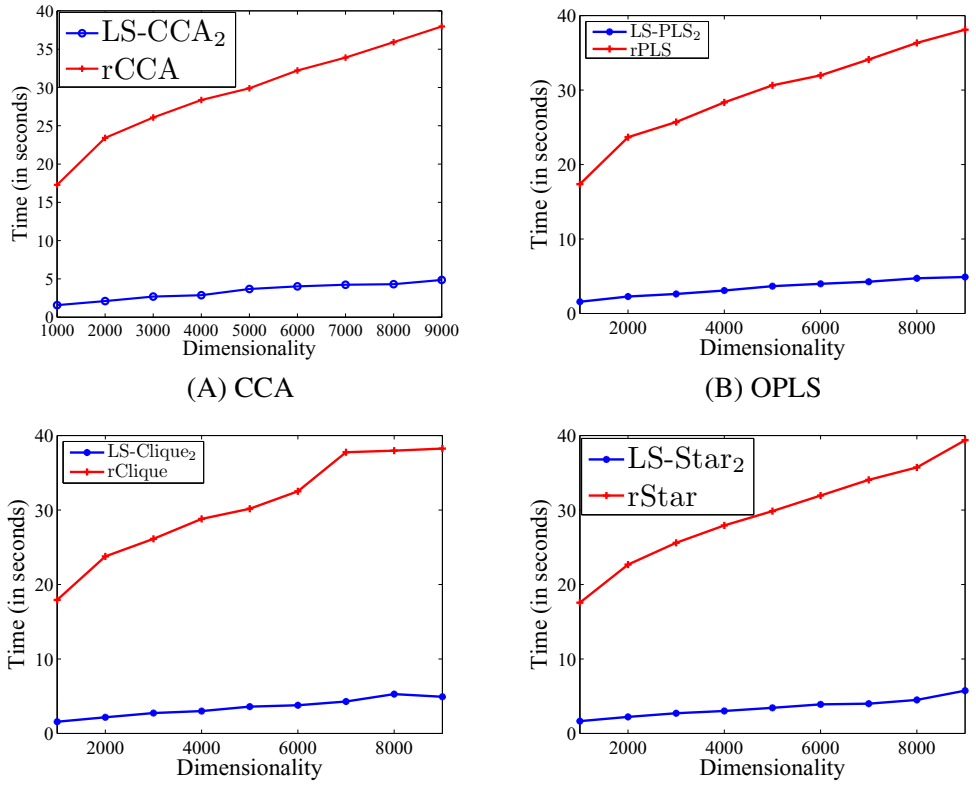
We also evaluate the scalability of the two formulations in terms of the training sample size. Figure 4.5 shows the computation time of the two formulations on the Yahoo! data set as the training sample size increases with the data dimensionality fixed at 5000. We can also observe that the least squares formulation is much more scalable than the original one.

4.10 Conclusions

In this chapter we present a hypergraph spectral learning formulation for multi-label classification. In this formulation, a hypergraph is constructed to capture the correlation contained in different labels. We show that HSL reduces to a generalized eigenvalue problem, which may

be computationally expensive for large-scale problems. To reduce the computational cost, we propose an efficient least squares formulation, which is shown to be equivalent to hypergraph spectral learning under a mild condition. Furthermore, we show that a class of dimensionality reduction algorithms exhibiting the form of the generalized eigenvalue problem similar to H-SL are equivalent to the least squares formulation. This class of algorithms include Canonical Correlation Analysis (CCA), Orthonormalized PLS (OLS), and Linear Discriminant Analysis (LDA). Based on the least squares formulation, efficient algorithms for solving least squares problems can be applied to scale these techniques to very large data sets. In addition, existing regularization techniques for least squares can be incorporated to improve the generalization ability and induce sparsity in the resulting model.

We have conducted experiments using benchmark data sets, and experimental results show that hypergraph spectral learning is effective in capturing the higher-order relations in multi-label problems. Our experimental results confirm the established equivalence relationship under the given assumption for all dimensionality reduction algorithms exhibiting the form of the generalized eigenvalue problem similar to HSL. Results show that the performance of the least squares formulation and the original generalized eigenvalue problem is very close even when the assumption is violated. Our experiments also demonstrate the effectiveness and scalability of the least squares extensions.



(A) CCA (B) OPLS

(A) HSL-Clique (B) HSL-Clique

Figure 4.4: Computation time (in seconds) of the generalized eigenvalue problem and the corresponding least squares formulation on the Yahoo\Arts&Humanities data set as the data dimensionality increases. The x-axis represents the data dimensionality and the y-axis represents the computation time.

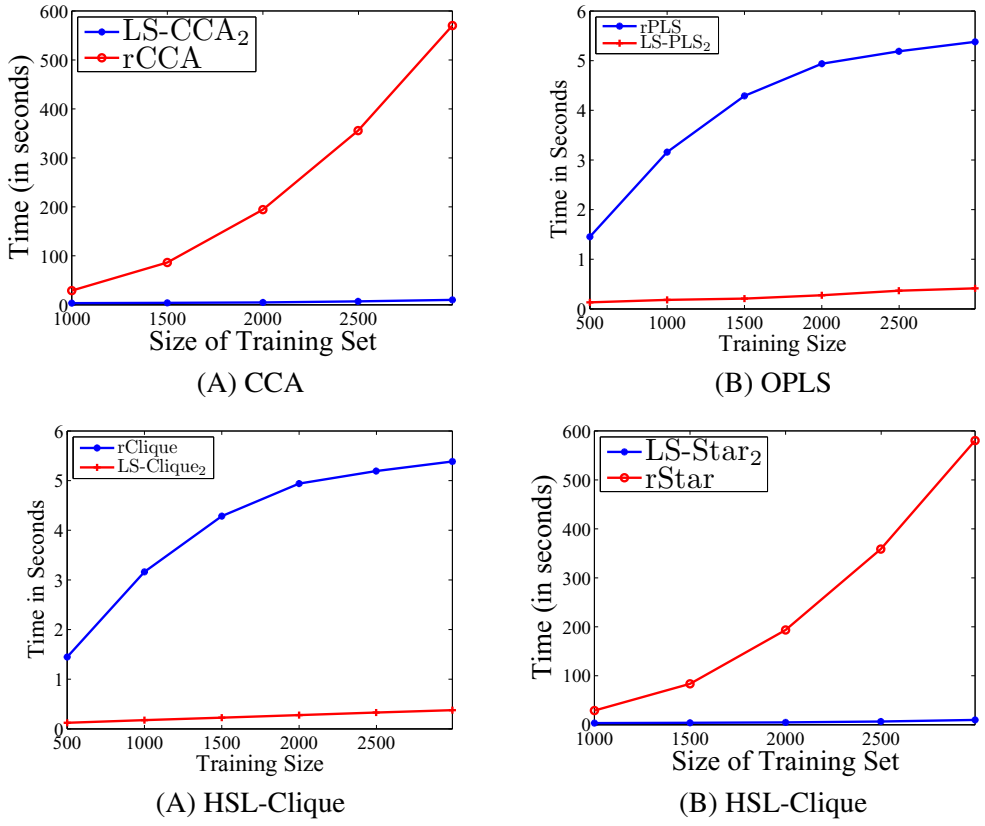


Figure 4.5: Computation time (in seconds) of the generalized eigenvalue problem and the corresponding least squares formulation on the Yahoo\Arts&Humanities data set as the training sample size increases. The x-axis represents the training sample size and the y-axis represents the computation time.

Chapter 5

THE SCALABLE TWO-STAGE APPROACH

5.1 Introduction

Recent technological innovations have allowed us to collect massive amounts of data with a large number of features. One of the key issues in such data analysis is the *curse of dimensionality* [11], i.e., an enormous number of samples is required to perform accurate prediction on problems with large numbers of features. Dimensionality reduction, which extracts a small number of features by removing the irrelevant, redundant, and noisy information, is an effective way to overcome the curse of dimensionality. Many dimensionality reduction algorithms have been proposed in the past, including Canonical Correlation Analysis (CCA) [129], Partial Least Squares (PLS) [97], Linear Discriminant Analysis (LDA) [166] and Hypergraph Spectral Learning (HSL) [56]. A common characteristic of these algorithms is that they can be formulated as a generalized eigenvalue problem. Although well-established algorithms in numerical linear algebra exist to solve generalized eigenvalue problems [172, 234], they are in general computationally expensive to solve and hence may not scale to large-size problems.

There have been several recent attempts to improve the scalability of dimensionality reduction algorithms [56, 138, 139, 220]. In Chapter 4, we transform some dimensionality reduction techniques that exhibit the form of the generalized eigenvalue problem into an equivalent least squares formulation, which can be solved efficiently using existing algorithms such as the iterative conjugate gradient algorithm [172, 221]. However, it suffers from several drawbacks, which limits its applicability in practice. First, the equivalent transformation relies on a key assumption that all the data points are linear independent. This assumption tends to hold for high-dimensional data, but it is likely to fail for (relatively) lower-dimensional data. Secondly, the equivalence relationship between the least squares formulation and the original formulation does not hold when the regularization is employed. However, regularization has been shown to be effective in practice, and it has been used in many data mining and machine learning algorithms including support vector machines [62].

In this chapter, we introduce an efficient two-stage approach to solve a class of dimensionality reduction algorithms, including CCA, OPLS, LDA and HSL. In the first stage we

solve a least squares problem using the iterative conjugate gradient algorithm [172, 221]. The distinct property of this stage is its low time complexity. In the second stage, the original data is projected onto a lower-dimensional space, and then we solve a generalized eigenvalue problem with a significantly reduced size. The proposed two-stage approach scales linearly in terms of both the sample size and the data dimensionality, thus applicable for large-size problems. The main contributions include:

- We rigorously prove the equivalence relationship between the two-stage approach and the direct approach which solves the generalized eigenvalue problem directly. Compared with previous work, the two-stage approach does not require any assumption and can be applied in all cases.
- We show that the two-stage approach can be further extended to the regularization setting. The equivalence relationship is also rigorously proved in this case.

We have conducted extensive experiments to evaluate the proposed two-stage approach using both synthetic and real-world benchmark data sets. Our experimental results confirm the equivalence relationship established in this chapter. Results also demonstrate the scalability of the proposed two-stage approach.

The rest of this chapter is organized as follows: Section 5.2 reviews the class of dimensionality reduction algorithms discussed in the chapter. In Section 5.3, we present the two-stage approach and establish the equivalence relationship. We further extend the two-stage approach to the regularization setting in Section 5.4. A comprehensive empirical study is reported in Section 5.5. Finally, we conclude in Section 5.6.

5.2 A Class of dimensionality reduction algorithms

In this section we discuss a class of dimensionality reduction techniques. More details about these techniques have been presented in Section 4.4. Specifically, the class of generalized eigenvalue problems considered in this chapter are in the following form:

$$\mathbf{X}\mathbf{S}\mathbf{X}^T\mathbf{w} = \lambda\mathbf{X}\mathbf{X}^T\mathbf{w}, \tag{5.1}$$

where $\mathbf{S} \in \mathbb{R}^{n \times n}$ is a symmetric and positive semi-definite matrix. The principal eigenvectors corresponding to nonzero eigenvalues of this generalized eigenvalue problem are of great interest in machine learning and data mining. Typically, we reformulate the generalized eigenvalue problem in Eq. (5.1) into the following standard eigenvalue problem:

$$(\mathbf{X}\mathbf{X}^T)^\dagger \mathbf{X}\mathbf{S}\mathbf{X}^T w = \lambda \mathbf{w}, \tag{5.2}$$

where $(\mathbf{X}\mathbf{X}^T)^\dagger$ is the Moore-Penrose pseudoinverse of $\mathbf{X}\mathbf{X}^T$. As discussed in Section 3.6, we can formulate the generalized eigenvalue problem in Eq. (5.1) equivalently as an optimization problem:

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{Tr}(\mathbf{W}^T \mathbf{X}\mathbf{S}\mathbf{X}^T \mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^T \mathbf{X}\mathbf{X}^T \mathbf{W} = \mathbf{I}. \end{aligned} \tag{5.3}$$

Many existing dimensionality reduction algorithms are in the form in Eq. (4.32) or (4.33), including CCA, OPLS, HSL and LDA. In particular, the matrix $\mathbf{S}$ can be expressed in the following form:

$$\mathbf{S} = \mathbf{H}\mathbf{H}^T, \tag{5.4}$$

where $\mathbf{H} \in \mathbb{R}^{n \times k}$ is often constructed from the label information for the data. The definitions of $\mathbf{S}$ and $\mathbf{H}$ for different dimensionality reduction algorithms are given in Section 4.4.

Regularization is commonly used to improve the generalization performance and control the model complexity. Specifically, a regularization term $\gamma \mathbf{I}_d$ with $\gamma > 0$ is added to $\mathbf{X}\mathbf{X}^T$ in Eq. (4.32) to avoid the singularity of $\mathbf{X}\mathbf{X}^T$, resulting in the following generalized eigenvalue problem with regularization:

$$\mathbf{X}\mathbf{S}\mathbf{X}^T \mathbf{w} = \lambda (\mathbf{X}\mathbf{X}^T + \gamma \mathbf{I}_d)\mathbf{w}. \tag{5.5}$$

5.3 The Two-Stage Approach without Regularization

In this section, we present our two-stage approach and show that this approach is equivalent to the direct approach, which solves the generalized eigenvalue problem in Eq. (5.1).

The Algorithm

In the two-stage approach, we first solve a least squares problem by regressing $\mathbf{X}$ on $\mathbf{H}^T$. In other words, $\mathbf{H}^T$ can be considered as the "latent target" encoded by the label information $\mathbf{Y}$. After

Algorithm 6 The Two-Stage Approach without Regularization

Input: $\mathbf{X}$, $\mathbf{H}$.
Output: $\mathbf{W}$.
Stage 1: Solve the following least squares problem:

$$\min_{\mathbf{W}_1} \|\mathbf{W}_1^T\mathbf{X} - \mathbf{H}^T\|_F^2. \tag{5.6}$$

Stage 2: Compute $\tilde{\mathbf{X}} = \mathbf{W}_1^T\mathbf{X}$, and solve the following optimization problem:

$$\begin{aligned} \max_{\mathbf{W}_2} \quad & \mathrm{Tr}(\mathbf{W}_2^T\tilde{\mathbf{X}}\mathbf{H}\mathbf{H}^T\tilde{\mathbf{X}}^T\mathbf{W}_2) \\ \text{s.t.} \quad & \mathbf{W}_2^T\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T\mathbf{W}_2 = \mathbf{I}_\ell. \end{aligned} \tag{5.7}$$

Compute $\mathbf{W} = \mathbf{W}_1\mathbf{W}_2$ as the final solution.

projecting the data matrix $\mathbf{X}$ onto the subspace, we solve the resulting generalized eigenvalue problem by replacing the data matrix in Eq. (5.1) with the projected data matrix. Note that the data dimensionality is reduced dramatically after the projection, thus the resulting generalized eigenvalue problem in the second step can be solved efficiently. The two-stage approach is summarized in Algorithm 6.

Time Complexity Analysis

In our implementation, we apply the LSQR algorithm [221], a conjugate gradient method for solving the least squares problem in the first stage. Previous studies have shown that LSQR is reliable even for ill-conditioned problems [221]. In addition, when the data matrix $\mathbf{X}$ is sparse, the least squares can be solved very efficiently using LSQR. Note that k least squares problem are solved in the first stage of Algorithm 6, thus the total computational cost of the first stage is $O(Nk(3n+5d+2dn))$ using LSQR when $\mathbf{X}$ is dense, where N is the total number of iterations. When the data matrix $\mathbf{X}$ is sparse, the cost of LSQR is reduced to $O(Nk(3n+5d+2z))$, where z is number of nonzero entries in $\mathbf{X}$.

In the second stage, the cost of computing $\tilde{\mathbf{X}}$ is $O(ndk)$ when $\mathbf{X}$ is dense or $O(kz)$ when $\mathbf{X}$ is sparse. Since the size of $\tilde{\mathbf{X}}$ is significantly reduced, the cost of solving the optimization problem is $O(nk^2)$. The cost of combining $\mathbf{W}_1$ and $\mathbf{W}_2$ is $O(nk\ell)$, where $\ell(\ell \leq k)$ is the number of final projection vectors. Therefore, the total computational cost is $O(Nk(3n+5d+2z)+kz)$ when $\mathbf{X}$ is sparse.

The Equivalence Relationship

Next we show the equivalence relationship between the two-stage approach and the direct approach, which solves the original eigenvalue problem in Eq.(4.33). It follows from the standard techniques in linear algebra that the solution to the least squares problem in Eq. (5.6) is given by

$$\mathbf{W}_1 = (\mathbf{X}\mathbf{X}^T)^\dagger \mathbf{X}\mathbf{H} \in \mathbb{R}^{d\times k}. \tag{5.8}$$

Let the singular value decomposition (SVD) [172] of $\mathbf{X}$ be

$$\mathbf{X} \quad = \quad \mathbf{U}\Sigma\mathbf{V}^T = \mathbf{U}_1\Sigma_1\mathbf{V}_1^T, \tag{5.9}$$

where $\mathbf{U} \in \mathbb{R}^{d\times d}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are orthogonal, $\mathbf{U}_1 \in \mathbb{R}^{d\times r}$ and $\mathbf{V}_1 \in \mathbb{R}^{n\times r}$ have orthonormal columns, $\Sigma \in \mathbb{R}^{d\times n}$ and $\Sigma_1 \in \mathbb{R}^{r\times r}$ are diagonal, and $r = \text{rank}(\mathbf{X})$. Then $\mathbf{W}_1$ can be represented as

$$\mathbf{W}_1 = \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_1^T\mathbf{H}. \tag{5.10}$$

It follows that $\tilde{\mathbf{X}}$ can be expressed as

$$\tilde{\mathbf{X}} = \mathbf{W}_1^T\mathbf{X} = \mathbf{H}^T\mathbf{V}_1\Sigma_1^{-1}\mathbf{U}_1^T\mathbf{U}_1\Sigma_1\mathbf{V}_1^T = \mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T, \tag{5.11}$$

and

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^T = \mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{H} = \mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{H}.$$

Thus, the optimization problem in Eq. (5.7) can be simplified into the following form:

$$\begin{aligned} \max_{\mathbf{W}_2} \quad & \text{Tr}(\mathbf{W}_2^T\mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{H}\mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{H}\mathbf{W}_2) \\ \text{s.t.} \quad & \mathbf{W}_2^T\mathbf{H}^T\mathbf{V}_1\mathbf{V}_1^T\mathbf{H}\mathbf{W}_2 = \mathbf{I}_\ell. \end{aligned}$$

Denote

$$\mathbf{A} = \mathbf{H}^T\mathbf{V}_1 \in \mathbb{R}^{k\times r}. \tag{5.12}$$

Then the optimization problem in the second stage can be reformulated as follows:

$$\begin{aligned} \max_{\mathbf{W}_2} \quad & \text{Tr}(\mathbf{W}_2^T\mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T\mathbf{W}_2) \\ \text{s.t.} \quad & \mathbf{W}_2^T\mathbf{A}\mathbf{A}^T\mathbf{W}_2 = \mathbf{I}_\ell. \end{aligned} \tag{5.13}$$

Let the compact SVD of $\mathbf{A}$ be

$$\mathbf{A} = \mathbf{U}_A \Sigma_A \mathbf{V}_A^T, \tag{5.14}$$

where $\mathbf{U}_A \in \mathbb{R}^{k\times t}$, $\Sigma_A \in \mathbb{R}^{t\times t}$, $\mathbf{V}_A \in \mathbb{R}^{r\times t}$, and $t = \text{rank}(\mathbf{A})$. Based on the above definitions, the solution to the two-stage approach is summarized in the following theorem.

Theorem 5.1 *The top ℓ projection vectors computed by Eq. (5.7) are given by*

$$\boldsymbol{W}_2 = (\boldsymbol{U}_A \Sigma_A^{-1})_\ell, \tag{5.15}$$

where $(\boldsymbol{U}_A \Sigma_A^{-1})_\ell$ consists of the first $\ell(\ell \leq \text{rank}(\mathbf{A}))$ columns of $(\boldsymbol{U}_A \Sigma_A^{-1})$. Thus, the projection vectors computed by the two-stage approach are

$$\boldsymbol{W} = \boldsymbol{W}_1 \boldsymbol{W}_2 = \boldsymbol{U}_1 \Sigma_1^{-1} \mathbf{V}_{A\ell}. \tag{5.16}$$

When $\ell = \text{rank}(\mathbf{A})$, $\boldsymbol{W}$ can be simplified as

$$\boldsymbol{W} = \boldsymbol{U}_1 \Sigma_1^{-1} \mathbf{V}_A. \tag{5.17}$$

Proof It follows from the Lagrange dual function in optimization theory that the optimization problem in Eq. (5.13) can be expressed as the following eigenvalue problem:

$$\left(\mathbf{A}\mathbf{A}^T\right)^\dagger \mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T\mathbf{w}_2 = \lambda \mathbf{w}_2. \tag{5.18}$$

Next we derive the eigendecomposition of the matrix $\left(\mathbf{A}\mathbf{A}^T\right)^\dagger \mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T$ as follows:

$$\begin{aligned}
& \left(\mathbf{A}\mathbf{A}^T\right)^\dagger \mathbf{A}\mathbf{A}^T\mathbf{A}\mathbf{A}^T \\
= \; & \left(\mathbf{U}_A \Sigma_A^2 \mathbf{U}_A^T\right)^\dagger \mathbf{U}_A \Sigma_A^2 \mathbf{U}_A^T \mathbf{U}_A \Sigma_A^2 \mathbf{U}_A^T \\
= \; & \mathbf{U}_A \Sigma_A^{-2} \mathbf{U}_A^T \mathbf{U}_A \Sigma_A^4 \mathbf{U}_A^T \\
= \; & \mathbf{U}_A \Sigma_A^2 \mathbf{U}_A \\
= \; & [\mathbf{U}_A, \mathbf{U}_A^\perp] \begin{bmatrix} \Sigma_A^2 & 0 \\ 0 & 0 \end{bmatrix} [\mathbf{U}_A, \mathbf{U}_A^\perp]^T,
\end{aligned}$$

where $[\mathbf{U}_A, \mathbf{U}_A^\perp] \in \mathbb{R}^{k\times k}$ is orthogonal. Thus, the eigenvectors corresponding to the top ℓ eigenvalues are given by

$$\mathbf{W}_2 = \mathbf{U}_{A\ell},$$

where $\mathbf{U}_{A\ell}$ consists of the first ℓ columns of $\mathbf{U}_A$. To ensure that the constraint in Eq. (5.13) is satisfied, we normalize the columns of $\mathbf{W}_2$ without affecting the range space of $\mathbf{W}_2^T$:

$$\mathbf{W}_2 = (\mathbf{U}_A \Sigma_A^{-1})_\ell.$$

Combing Eqs. (5.10) and (5.15), we have

$$\mathbf{W} = \mathbf{W}_1 \mathbf{W}_2 = \mathbf{U}_1 \Sigma_1^{-1} \mathbf{A}^T (\mathbf{U}_A \Sigma_A^{-1})_\ell = \mathbf{U}_1 \Sigma_1^{-1} \mathbf{V}_{A\ell}.$$

When $\ell = \text{rank}(\mathbf{A})$, we have $\mathbf{V}_{A\ell} = \mathbf{V}_A$ and $\mathbf{W} = \mathbf{U}_1 \Sigma_1^{-1} \mathbf{V}_A$. This completes the proof of this theorem. ∎

Note that the solution to the generalized eigenvalue problem in Eq. (5.1) consists of the principal eigenvectors of matrix $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$. We follow [138] to derive the eigen-decomposition of $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ and show the equivalence relationship between the two-stage approach and the direct approach. The results are summarized in the following theorem:

Theorem 5.2 *The eigenvectors corresponding to the top ℓ eigenvalues of $(\boldsymbol{X}\boldsymbol{X}^T)^\dagger(\boldsymbol{X}\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{X}^T)$ are given by*

$$\boldsymbol{W}_0 = \boldsymbol{U}_1 \Sigma_1^{-1} \mathbf{V}_{A\ell}, \tag{5.19}$$

where $\mathbf{V}_{A\ell}$ consists of the first ℓ columns of $\mathbf{V}_A$. Thus, the two-stage approach produces the same solution as the direct approach which solves the original generalized eigenvalue problem directly.

Proof Given $\mathbf{V}_A \in \mathbb{R}^{r\times t}$ with orthonormal columns, there exists $\mathbf{V}_A^\perp \in \mathbb{R}^{r\times(r-t)}$ such that $[\mathbf{V}_A, \mathbf{V}_A^\perp] \in \mathbb{R}^{r\times r}$ is an orthogonal matrix [172]. Hereafter, we denote $[\mathbf{V}_A, \mathbf{V}_A^\perp] = \mathbf{V}_A^s$.

We can decompose $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ as follows:

$$\begin{aligned}
& (\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T) \\
= \; & \mathbf{U}_1 \Sigma_1^{-2} \mathbf{U}_1^T \mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T \\
= \; & \mathbf{U}_1 \Sigma_1^{-2} \mathbf{U}_1^T \mathbf{U}_1 \Sigma_1 \mathbf{V}_1^T \mathbf{H}\mathbf{H}^T \mathbf{V}_1 \Sigma_1 \mathbf{U}_1^T \\
= \; & \mathbf{U}_1 \Sigma_1^{-1} \mathbf{A}^T \mathbf{A} \Sigma_1 \mathbf{U}_1^T
\end{aligned}$$

$$
\begin{aligned}
&= \mathbf{U}\begin{bmatrix}\mathbf{I}_r\\ 0\end{bmatrix}\Sigma_1^{-1}\mathbf{A}^T\mathbf{A}\Sigma_1\begin{bmatrix}\mathbf{I}_r & 0\end{bmatrix}\mathbf{U}^T\\
&= \mathbf{U}\begin{bmatrix}\Sigma_1^{-1}\mathbf{A}^T\mathbf{A}\Sigma_1 & 0\\ 0 & 0\end{bmatrix}\mathbf{U}^T\\
&= \mathbf{U}\begin{bmatrix}\Sigma_1^{-1}\mathbf{V}_A^s & 0\\ 0 & \mathbf{I}\end{bmatrix}\begin{bmatrix}\Sigma_A^2 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{V}_A^{s\,T}\Sigma_1 & 0\\ 0 & \mathbf{I}\end{bmatrix}\mathbf{U}^T.
\end{aligned}
$$

It is clear that the eigenvectors corresponding to the top ℓ eigenvalues of $(\mathbf{X}\mathbf{X}^T)^\dagger(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ are given by

$$\mathbf{W}_0 = \mathbf{U}_1\Sigma_1^{-1}\mathbf{V}_{A\ell}.$$

The equivalence relationship follows from Eqs. (5.16) and (5.19). This completes the proof of the theorem. ∎

A consequence of Theorem 5.1 is that solving the optimization problem in Eq. (5.7) in the second stage amounts to computing the SVD of the matrix $\mathbf{A}$. Note that $\mathbf{A} \in \mathbb{R}^{k\times r}$, where $r = \text{rank}(\mathbf{X}) \leq \min\{d,n\}$, thus the computational cost of the SVD of $\mathbf{A}$ is quite low. In practice we can perform SVD on $\mathbf{A}$ directly instead of solving the optimization problem in Eq. (5.7).

Remark 5.1 *In Chapter 4 we discussed a equivalent least squares formulation for the same class of dimensionality reduction algorithms, which is with the same computational cost as the two-stage approach discussed in this chapter. However, the analysis in Chapter 4 assumes that the data matrix $\boldsymbol{X}$ is of full rank (before centering). This tends to fail for (relatively) lower-dimensional data. In particular, when the number of data points is larger than the number of dimensions, this assumption is likely to be violated. The two-stage algorithm presented in this chapter significantly improves previous work by relaxing this assumption.*

5.4 The Two-Stage Approach with Regularization

Regularization is commonly employed to control the model complexity and avoid overfitting. In this section we present the two-stage approach with regularization. We show the equivalence relationship between the two-stage approach and the direct approach, which solves the regularized generalized eigenvalue problem in Eq. (5.5).

Algorithm 7 The Two-Stage Approach with Regularization

Input: $\mathbf{X}$, $\mathbf{H}$, γ.
Output: $\mathbf{W}$.
Stage 1: Solve the following least squares problem:

$$\min_{\mathbf{W}_1} \|\mathbf{W}_1^T\mathbf{X} - \mathbf{H}^T\|_F^2 + \gamma\|\mathbf{W}_1\|_F^2. \tag{5.20}$$

Stage 2: Compute $\tilde{\mathbf{X}} = \mathbf{W}_1^T\mathbf{X}$ and $\mathbf{D} = \tilde{\mathbf{X}}\mathbf{H}$. Compute the compact SVD of $\mathbf{D} = \mathbf{U}_D\Sigma_D\mathbf{U}_D^T$ and set $\mathbf{W}_2 = \mathbf{U}_D\Sigma_D^{-1/2}$.
Compute $\mathbf{W} = \mathbf{W}_1\mathbf{W}_2$ as the final solution.

The Algorithm

To handle the regularization in the generalized eigenvalue problem in Eq. (5.5), we solve a penalized least squares problem, or ridge regression [79] in the first step. Note that the "latent target" is the same as the one used in Algorithm 6, and the difference is the regularization term included in the least squares problem. After projecting the data matrix $\mathbf{X}$ onto the subspace, we compute an auxiliary matrix $\mathbf{D} \in \mathbb{R}^{k\times k}$ and its SVD. Intuitively, the SVD computation of $\mathbf{D}$ amounts to solving the original optimization problem by replacing $\mathbf{X}$ with $\tilde{\mathbf{X}}$. Note that the size of $\mathbf{D}$ is very small, thus the cost of computing the SVD of $\mathbf{D}$ is relatively low. The algorithm outline is summarized in Algorithm 7.

Time Complexity Analysis

Similar to Algorithm 6, the least squares problem in the first stage is solved using the LSQR algorithm [221] with the same cost. In the second stage, since $k \ll d$, the most expensive part is the computation of $\tilde{\mathbf{X}}$ with a cost of $O(kdn)$ if $\mathbf{X}$ is dense or $O(kz)$ if $\mathbf{X}$ is sparse, where z is the number of nonzero entries in the data matrix $\mathbf{X}$. Therefore, the total computational cost is $O(Nk(3n+5d+2z)+kz)$ if $\mathbf{X}$ is sparse.

The Equivalence Relationship

Next we show that the two-stage approach with regularization yields the same solution as the direct approach, which solves the regularized generalized eigenvalue problem in Eq. (5.5). Following the standard techniques in linear algebra, the solution to the least squares problem with regularization in Eq. (5.20) is given by

$$\mathbf{W}_1 = (\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^\dagger\mathbf{X}\mathbf{H} = \mathbf{U}_1\left(\Sigma_1^2 + \gamma\mathbf{I}\right)^{-1}\Sigma_1\mathbf{V}_1^T\mathbf{H}. \tag{5.21}$$

Then $\tilde{\mathbf{X}}$ can be expressed as

$$\tilde{\mathbf{X}} = \mathbf{W}_1^T \mathbf{X} = \mathbf{H}^T \mathbf{V}_1 \Sigma_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1} \Sigma_1 \mathbf{V}_1^T, \tag{5.22}$$

and the matrix $\mathbf{D} \in \mathbb{R}^{k \times k}$ can be represented as:

$$\mathbf{D} = \tilde{\mathbf{X}}\mathbf{H} = \mathbf{H}^T \mathbf{V}_1 \Sigma_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1} \Sigma_1 \mathbf{V}_1^T \mathbf{H} = \mathbf{B}\mathbf{B}^T, \tag{5.23}$$

where $\mathbf{B} \in \mathbb{R}^{k \times r}$ is defined as

$$\mathbf{B} = \mathbf{H}^T \mathbf{V}_1 \Sigma_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1/2}. \tag{5.24}$$

The solution to the two-stage approach in Algorithm 7 is summarized in the following theorem:

Theorem 5.3 *Let the compact SVD of $\boldsymbol{B}$ be*

$$\boldsymbol{B} = \boldsymbol{U}_B \Sigma_B \mathbf{V}_B^T, \tag{5.25}$$

where $\boldsymbol{U}_B \in \mathbb{R}^{k \times q}$, $\Sigma_B \in \mathbb{R}^{q \times q}$, $\mathbf{V}_B \in \mathbb{R}^{r \times q}$, and $q = \text{rank}(\boldsymbol{B})$. The top ℓ ($\ell \leq \text{rank}(\boldsymbol{B})$) projection vectors computed by Algorithm 7 are given by

$$\boldsymbol{W} = \boldsymbol{W}_1 \boldsymbol{W}_2 = \boldsymbol{U}_1 (\Sigma_1^2 + \gamma \boldsymbol{I})^{-1/2} \mathbf{V}_{B\ell}, \tag{5.26}$$

where

$$\boldsymbol{W}_2 = (\boldsymbol{U}_B \Sigma_B^{-1})_\ell, \tag{5.27}$$

$\mathbf{V}_{B\ell}$ consists of the first ℓ columns of $\mathbf{V}_B$, and $(\boldsymbol{U}_B \Sigma_B^{-1})_\ell$ consists of the first ℓ columns of $\boldsymbol{U}_B \Sigma_B^{-1}$. When $\ell = \text{rank}(\boldsymbol{B})$, $\boldsymbol{W}$ can be simplified as

$$\boldsymbol{W} = \boldsymbol{U}_1 (\Sigma_1^2 + \gamma \boldsymbol{I})^{-1/2} \mathbf{V}_B. \tag{5.28}$$

Proof Note that $\mathbf{D} = \mathbf{B}\mathbf{B}^T$. The SVD of $\mathbf{D}$ can be obtained based on the SVD of $\mathbf{B}$ as follows:

$$\mathbf{D} = \mathbf{U}_B \Sigma_B^2 \mathbf{U}_B^T = \mathbf{U}_D \Sigma_D \mathbf{U}_D^T. \tag{5.29}$$

It follows from Algorithm 7 that

$$\mathbf{W}_2 = \mathbf{U}_B \Sigma_B^{-1}.$$

Recall that $\mathbf{W}_1$ can also be represented using $\mathbf{B}$ as:

$$\mathbf{W}_1 \quad = \quad \mathbf{U}_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1} \Sigma_1 \mathbf{V}_1^T \mathbf{H} = \mathbf{U}_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1/2} \mathbf{B}^T.$$

We can thus derive $\mathbf{W}$ as follows:

$$\begin{aligned} \mathbf{W} &= \mathbf{W}_1 \mathbf{W}_2 \\ &= \mathbf{U}_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1/2} \mathbf{B}^T \mathbf{U}_B \Sigma_B^{-1} \\ &= \mathbf{U}_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1/2} \mathbf{V}_B \Sigma_B \mathbf{U}_B^T \mathbf{U}_B \Sigma_B^{-1} \\ &= \mathbf{U}_1 \left(\Sigma_1^2 + \gamma \mathbf{I}\right)^{-1/2} \mathbf{V}_B. \end{aligned}$$

If only the first ℓ projection vectors are required, then the resulting $\mathbf{W}$ is given by

$$\mathbf{W} = \mathbf{U}_1 (\Sigma_1^2 + \gamma \mathbf{I})^{-1/2} \mathbf{V}_{B\ell}.$$

This completes the proof of the theorem. ∎

We follow [130] for the eigendecomposition of the matrix $\left(\mathbf{X}\mathbf{X}^T + \gamma \mathbf{I}\right)^{-1} (\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$. The eigendecomposition is summarized in the following theorem, based on which we also obtain the equivalence relationship between the two-stage approach and the direct approach.

Theorem 5.4 *The eigenvectors corresponding to the top ℓ ($\ell \leq \text{rank}(\boldsymbol{B})$) eigenvalues of matrix $(\boldsymbol{X}\boldsymbol{X}^T + \gamma \boldsymbol{I})^{-1}(\boldsymbol{X}\boldsymbol{H}\boldsymbol{H}^T\boldsymbol{X}^T)$ are given by*

$$\boldsymbol{W}_0 = \boldsymbol{U}_1 (\Sigma_1^2 + \gamma \boldsymbol{I})^{-1/2} \mathbf{V}_{B\ell}, \tag{5.30}$$

where $\mathbf{V}_{B\ell}$ consists of the first ℓ columns of $\mathbf{V}_B$. Thus, the two-stage approach in Algorithm 7 is equivalent to the direct approach, which solves the generalized eigenvalue problem with regularization directly.

Proof Given $\mathbf{V}_B \in \mathbb{R}^{r \times q}$ with orthonormal columns, there exists $\mathbf{V}_B^{\perp} \in \mathbb{R}^{r \times (r-q)}$ such that $[\mathbf{V}_B, \mathbf{V}_B^{\perp}] \in \mathbb{R}^{r \times r}$ is an orthogonal matrix. Hereafter, we denote $[\mathbf{V}_B, \mathbf{V}_B^{\perp}] = \mathbf{V}_B^s$. We can di-

agonalize $(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ as follows:

$$
\begin{aligned}
& (\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T) \\
= \; & \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1}\Sigma_1\mathbf{V}_1^T\mathbf{H}\mathbf{H}^T\mathbf{V}_1\Sigma_1\mathbf{U}_1^T \\
= \; & \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\Sigma_1\mathbf{V}_1^T\mathbf{H}\mathbf{H}^T \\
& \mathbf{V}_1\Sigma_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}(\Sigma_1^2 + \gamma\mathbf{I})^{1/2}\mathbf{U}_1^T \\
= \; & \mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}^T\mathbf{B}(\Sigma_1^2 + \gamma\mathbf{I})^{1/2}\mathbf{U}_1^T \\
= \; & \mathbf{U}\begin{bmatrix}\mathbf{I}_r \\ 0\end{bmatrix}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}^T\mathbf{B}(\Sigma_1^2 + \lambda\mathbf{I})^{1/2}\begin{bmatrix}\mathbf{I}_r & 0\end{bmatrix}\mathbf{U}^T \\
= \; & \mathbf{U}\begin{bmatrix}(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{B}^T\mathbf{B}(\Sigma_1^2 + \gamma\mathbf{I})^{1/2} & 0 \\ 0 & 0\end{bmatrix}\mathbf{U}^T \\
= \; & \mathbf{U}\begin{bmatrix}(\Sigma_1^2 + \lambda\mathbf{I})^{-1/2}\mathbf{V}_B^s & 0 \\ 0 & \mathbf{I}\end{bmatrix}\begin{bmatrix}\Sigma_B^2 & 0 \\ 0 & 0\end{bmatrix}\begin{bmatrix}\mathbf{V}_B^{s\,T}(\Sigma_1^2 + \gamma\mathbf{I})^{1/2} & 0 \\ 0 & \mathbf{I}\end{bmatrix}\mathbf{U}^T.
\end{aligned}
$$

Thus, the eigenvectors corresponding to the top ℓ eigenvalues of $(\mathbf{X}\mathbf{X}^T + \gamma\mathbf{I})^{-1}(\mathbf{X}\mathbf{H}\mathbf{H}^T\mathbf{X}^T)$ are given by $\mathbf{U}_1(\Sigma_1^2 + \gamma\mathbf{I})^{-1/2}\mathbf{V}_{B\ell}$. The equivalence between the two-stage approach and the direct approach follows from Eqs. (5.26) and (5.30). This completes the proof of the theorem. ∎

Remark 5.2 *The least squares algorithm discussed in Chapter 4 only works for dimensionality reduction algorithms without regularization. This drawback limits its applicability in practice, since the regularized algorithms are more effective in practice due to its better generalization performance. The two-stage algorithm presented in this chapter significantly improves previous work by extending the equivalence to the regularization setting.*

Remark 5.3 *The two-stage approach can be extended to the kernel-induced feature space by following the trick in [204]. The equivalence relationship still holds for all dimensionality reduction techniques discussed above with and without regularization. Due to the space constraint, we skip detailed proof in this chapter.*

5.5 Experiments

We have performed extensive experiments to verify the established equivalence relationship and demonstrate the scalability of the proposed two-stage approach. All the experiments are

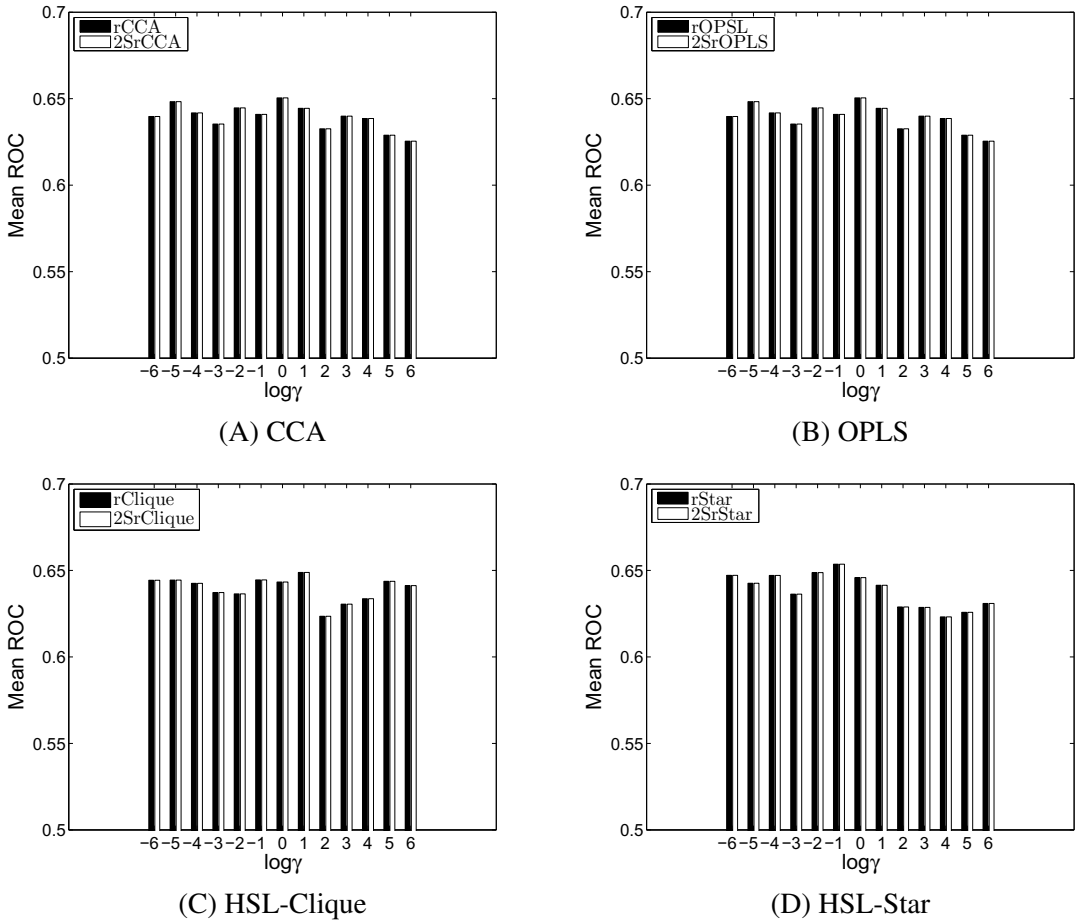
Table 5.1: Statistics of the data sets: n is the number of samples, d is the data dimensionality, and k is the number of labels (classes).

Data Set	Type	n	d	k
Syn1	Multi-class	1000	100	5
Syn2	Multi-class	1000	5000	5
Syn3	Multi-label	1000	100	5
Syn4	Multi-label	1000	5000	5
Ionosphere	Multi-class	351	34	2
Optical digits	Multi-class	5620	64	10
Satimage	Multi-class	6435	36	6
USPS	Multi-class	9298	256	10
Wine	Multi-class	178	13	3
Scene	Multi-label	2407	294	6
Yeast	Multi-label	2417	103	14
News20	Multi-class	15935	62061	20
RCV1-v2	Multi-label	3000	47236	101

Table 5.2: The value of $\|\mathbf{W}\mathbf{W}^T - \mathbf{W}_0\mathbf{W}_0^T\|_2$ under different values of the regularization parameter γ on the synthetic and real-world data sets. $\mathbf{W}_0$ is the solution of the generalized eigenvalue problem in Eq. (4.36) by solving it directly, and $\mathbf{W}$ is the solution of Eq. (4.36) by solving it using the two-stage approach. Each row corresponds to a specific algorithm and each column corresponds to a specific value of the regularization parameter γ.

Data	Technique	0	1.0e-6	1.0e-4	1.0e-2	1.0e+0	1.0e+2	1.0e+4	1.0e+6
Syn1	LDA	2.9e-18	3.6e-18	3.4e-18	3.1e-18	2.6e-18	2.5e-18	3.1e-19	3.0e-21
Syn2	LDA	5.8e-19	1.4e-18	1.2e-18	8.9e-19	1.2e-18	9.9e-19	2.3e-19	2.9e-21
Syn3	CCA	4.9e-18	8.4e-18	7.0e-18	6.5e-18	9.5e-18	6.0e-18	5.1e-19	7.2e-21
	OPLS	4.6e-18	5.0e-18	8.7e-18	5.0e-18	6.6e-18	6.1e-18	5.4e-19	5.0e-21
	HSL-Clique	1.0e-17	1.8e-17	1.2e-17	1.2e-17	1.5e-17	1.4e-17	2.9e-18	2.5e-20
	HSL-Star	1.4e-17	2.4e-17	9.3e-18	2.6e-17	2.1e-17	5.0e-17	9.8e-19	1.3e-20
Syn4	CCA	1.3e-18	5.2e-18	3.2e-18	1.8e-18	1.3e-18	1.8e-18	4.2e-19	5.9e-21
	OPLS	1.0e-18	1.1e-18	1.3e-18	1.5e-18	1.3e-18	1.3e-18	2.9e-19	5.9e-21
	HSL-Clique	2.7e-18	2.9e-18	2.7e-18	5.0e-18	3.2e-18	2.7e-18	8.9e-19	1.4e-20
	HSL-Star	2.5e-18	3.7e-18	2.9e-18	5.7e-18	4.1e-18	2.9e-18	1.1e-18	3.1e-20
Scene	CCA	2.4e-15	2.1e-15	6.1e-15	3.7e-15	1.2e-15	1.8e-16	6.0e-18	9.0e-20
	OPLS	2.0e-15	3.4e-15	3.8e-15	2.5e-15	1.1e-15	2.3e-16	1.1e-17	1.4e-19
	HSL-Clique	4.5e-15	9.1e-15	2.6e-14	1.2e-14	3.6e-15	1.3e-15	5.9e-17	1.0e-18
	HSL-Star	4.6e-15	3.3e-14	2.1e-14	7.7e-15	1.1e-14	2.5e-16	1.0e-16	6.5e-19
Yeast	CCA	1.6e-12	1.5e-11	1.2e-12	1.4e-15	6.9e-16	5.9e-17	1.7e-18	1.4e-20
	OPLS	4.1e-12	1.6e-11	3.7e-12	1.2e-14	1.5e-15	3.7e-16	3.2e-18	2.9e-20
	HSL-Clique	1.5e-12	1.4e-11	3.7e-12	3.9e-15	1.6e-15	2.7e-16	5.1e-18	2.5e-20
	HSL-Star	2.1e-12	1.0e-11	2.4e-12	1.1e-14	9.4e-15	1.1e-15	1.5e-17	4.4e-19
Wine	LDA	5.9e-17	2.1e-16	2.3e-16	2.1e-16	3.2e-17	2.2e-18	1.3e-20	2.0e-20
Satimage	LDA	4.6e-16	2.2e-15	8.4e-16	7.3e-16	7.7e-16	8.1e-17	3.9e-17	6.2e-19
Ionosphere	LDA	8.5e-18	1.0e-17	4.3e-18	2.1e-17	6.8e-18	6.6e-18	6.6e-20	1.1e-21
Optical	LDA	6.2e-18	7.2e-18	6.7e-18	5.7e-18	1.9e-18	1.5e-19	5.9e-20	5.6e-21
USPS	LDA	7.0e-15	3.0e-14	2.6e-14	6.6e-15	1.1e-16	3.0e-18	4.1e-19	6.6e-21

performed on a PC with Intel Core 2 Duo T9500 2.6G CPU and 4GB RAM. We implement all algorithms in Matlab, and all codes and synthetic data sets are available online[1].



(A) CCA (B) OPLS

(C) HSL-Clique (D) HSL-Star

Figure 5.1: Comparison of different approaches in terms of average ROC score for different techniques on the Yeast data set as the regularization parameter γ varies.

Experiment Setup

The dimensionality reduction algorithms discussed in this chapter can be divided into two categories: 1) LDA for multi-class classification; 2) HSL, CCA, and OPLS for multi-label classification. We use both multi-class and multi-label data sets, including synthetic and real-world data sets, in the experiments. Two synthetic data sets for multi-class classification as well as two synthetic data sets for multi-label classification are generated. In the synthetic data sets, each entry of the data matrix $\mathbf{X}$ is generated independently from the standard Gaussian distribution $\mathcal{N}(0,1)$. The number of classes is $k = 5$, and the labels are generated uniformly with random.

[1] www.public.asu.edu/~lsun27/Code/TwoStage.html

Five real-world data sets from UCI machine learning repository [243] and two benchmark data sets in multi-label classification [4, 28] are used in our experiments. To investigate the scalability of the two-stage approach, two large-scale data sets news20 [244] and RCV1-v2 [24] are used. The statistics of all data sets are summarized in Table 5.1, where n is the number of samples, d is the data dimensionality, and k is the number of labels (classes).

To distinguish different techniques tested in the experiments, we name the regularized techniques using a prefix "r" before the corresponding technique, e.g., "rCCA". The two-stage approach version are named using a prefix "2S" ("2S" means two stage) such as "2SCCA" and "2SrCCA", which are the two-stage approach versions of CCA and regularized CCA, respectively.

Performance Comparison

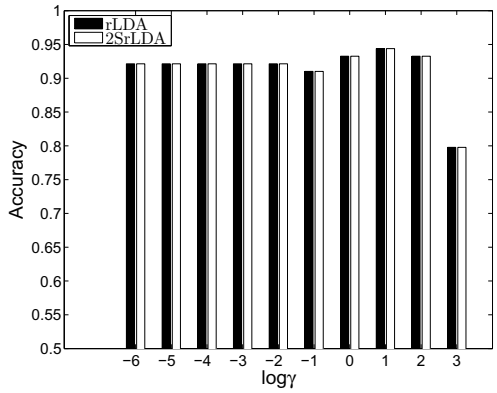


Figure 5.2: Comparison of different approaches in terms of classification accuracy for LDA on the Wine data set as the regularization parameter γ varies.

In this experiment, we compare the performance of different approaches for all techniques using both synthetic and real-world data sets. Denote $\mathbf{W}_0$ as the solution of the generalized eigenvalue problem in Eq. (5.5) obtained directly, and $\mathbf{W}$ as the solution of Eq. (5.5) obtained by using the two-stage approach.

To verify whether both approaches produce equivalent projections, we compute $\|\mathbf{W}_0\mathbf{W}_0^T - \mathbf{W}\mathbf{W}^T\|_2$ under different values of the regularization parameter γ. We vary the value of γ from 1e-6 to 1e6. It follows from [130] that $\|\mathbf{W}_0\mathbf{W}_0^T - \mathbf{W}\mathbf{W}^T\|_2 = 0$ if and only if $\mathbf{W}_0 = \mathbf{W}\mathbf{R}$, where $\mathbf{R}$ is an orthogonal matrix. Thus, both $\mathbf{W}$ and $\mathbf{W}_0$ project the original data onto the same lower-

dimensional space. Note that a direct comparison between $\mathbf{W}$ and $\mathbf{W}_0$ is possible only when the generalized eigenvalue problem in Eq. (5.5) admits a unique solution. This is not always the case, e.g., when two eigenvalues coincide.

The values of $\|\mathbf{W}_0\mathbf{W}_0^T - \mathbf{W}\mathbf{W}^T\|_2$ for all data sets are summarized in Table 5.2. In Table 5.2, each row corresponds to a specific technique and each column corresponds to a specific value of the regularization parameter γ. Note that two different variants of HSL, i.e., HSL-Clique and HSL-Star, which compute the Laplacian using different expansion schemes, are tested. It can be observed from Table 5.2 that for all values of the regularization parameter γ, $\|\mathbf{W}_0\mathbf{W}_0^T - \mathbf{W}\mathbf{W}^T\|_2$ is always very small, which confirms the equivalence relationship between $\mathbf{W}$ and $\mathbf{W}_0$ for projection.

Next, we investigate the classification performance of different techniques. We compare the performance of different approaches on the multi-label data set Yeast [28]. The data set is randomly partitioned into a training set and a test data set with equal size. After the projection matrix is learned from the training set, the test data set is projected onto the lower-dimensional space. In our experiments, the linear Support Vector Machines (SVM) is applied for classification. The average ROC score, namely, the area under ROC curve, over all labels are summarized in Figure 5.1 for all techniques. The regularization parameter γ varies from 1e-6 to 1e6. From Figure 5.1 we conclude that the proposed two-stage approach always produces the same classification results as the direct approach in all cases.

A similar experiment is performed on the multi-class data set Wine [243] for LDA. The classification accuracies of LDA under different values of γ are summarized in Figure 5.2, and similar observations can be made.

Scalability Comparison

In this experiment, we study the scalability of the two-stage approach in comparison with the direct approach. Since regularization is commonly used in practice, we compare the scalability of different algorithms with regularization. In terms of the implementation, the least squares problem in the first stage of the two-stage approach is solved using the LSQR algorithm [221]. For the direct approach which solves the generalized eigenvalue problem directly, the Lanczos algorithm [172] is applied. It is well-known that solving large-scale generalized eigenvalue

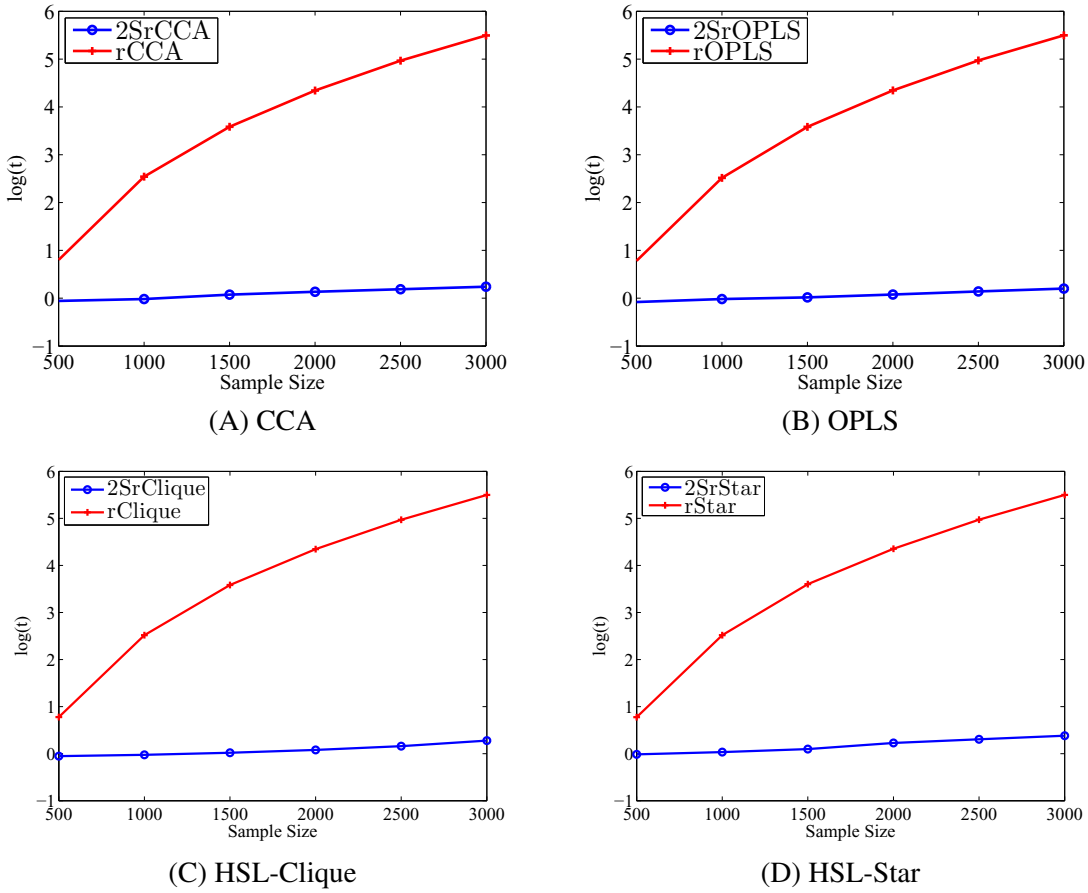


Figure 5.3: Scalability comparison on the RCV1-v2 data set as the sample size increases. The horizontal axis is the sample size, and the vertical axis is $\log(t)$, where t is the computation time (in seconds).

problems is much more difficult than the regular eigenvalue problems [234]. In order to transform the generalized eigenvalue problem into the regular eigenvalue problem, we can factor $\mathbf{X}\mathbf{X}^T$ or apply the standard Lanczos algorithm for the matrix $(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{S}\mathbf{X}^T$ using the $\mathbf{X}\mathbf{X}^T$ inner product [234]. Due to the issue of singularity for high-dimensional data set with small regularization for the second method, in this experiment we factor $\mathbf{X}\mathbf{X}^T$ and solve a symmetric eigenvalue problem using the Lanczos algorithm.

The computation time (in log scale) of different techniques on the large-scale data set RCV1-v2 is shown in Figures 5.3 and 5.4. In Figure 5.3, we increase the sample size from 500 to 3000 with a step size 500, and the dimensionality is fixed at 5000. The computation time of both approaches increases as the sample size increases. However, it can be observed that the

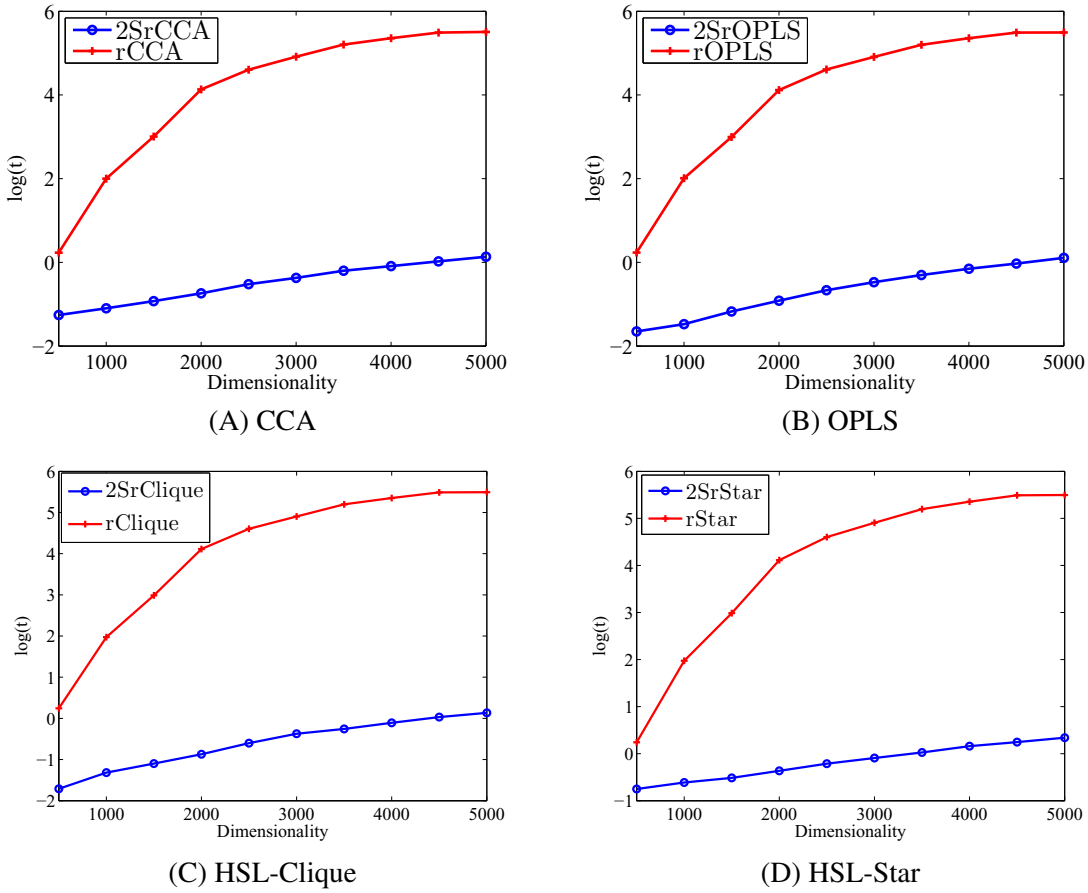


Figure 5.4: Scalability comparison on the RCV1-v2 data set as the dimensionality increases. The horizontal axis is the dimensionality, and the vertical axis is $\log(t)$, where t is the computation time (in seconds).

computation time of the two-stage approach is significantly less than that of the direct approach. In Figure 5.4 we fix the sample size at 3000 and increase the dimensionality from 500 to 5000 with a step size 500. Similar observations can be made from Figure 5.4.

We perform a similar experiment on the News20 data set for LDA (multi-class classification). The experimental results are summarized in Figure 5.5. In Figure 5.5 (left), we fix the dimensionality at 5000 and increase the sample size from 500 to 3000 with a step size 500. In Figure 5.5 (right), the dimensionality is increased from 500 to 3000 with a step size 500 while the sample size is fixed at 5000. It can be observed from these figures that the two-stage approach is much more efficient than the direct approach.

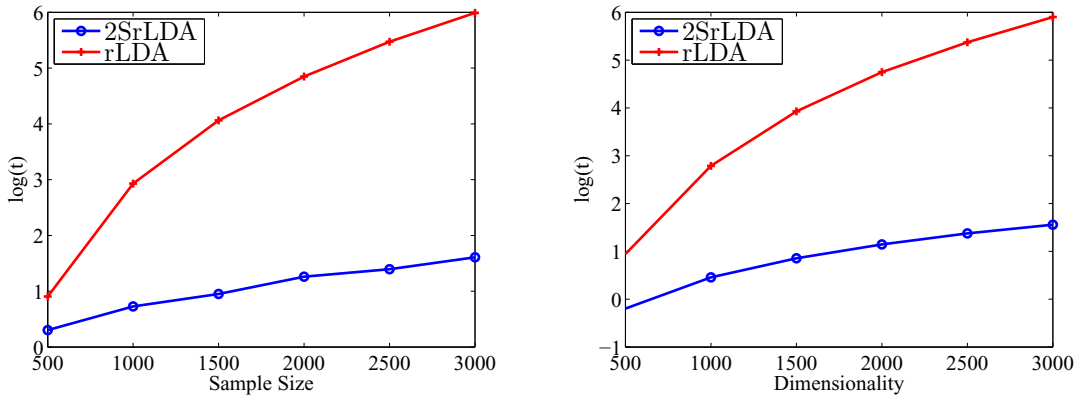


Figure 5.5: Scalability comparison on the News20 data set as the sample size (left) and dimensionality (right) increase. The horizontal axis is the sample size (left) or the dimensionality (right), and the vertical axis is $\log(t)$, where t is the computation time (in seconds).

5.6 Conclusions

In this chapter we present an efficient two-stage approach for a class of dimensionality reduction algorithms, including CCA, OPLS, HSL and LDA. We rigorously prove the equivalence relationship between the two-stage approach and the direct approach solving the generalized eigenvalue problem directly. Compared with previous work, one appealing feature of the two-stage approach is that no assumption is required. In addition, the two-stage approach can be further extended to the regularization setting. We show that the proposed two-stage approach scales linearly in terms of both the sample size and data dimensionality. We have performed extensive experiments on both synthetic and real-world data sets. Our experimental results confirm the established equivalence relationship. Results also demonstrate the scalability of the proposed two-stage approach.

Chapter 6

AUTOMATED ANNOTATION OF *DROSOPHILA* GENE EXPRESSION PATTERNS

6.1 Introduction

Detailed knowledge of the expression and interaction of genes is crucial to deciphering the mechanisms underlying cell-fate specification and tissue differentiation. DNA microarrays and RNA *in situ* hybridization are two primary methods for monitoring gene expression levels on a large scale. Microarrays provide a quantitative overview of the relative changes of expression levels of a large number of genes, but they do not often document the spatial information on individual genes. In contrast, RNA *in situ* hybridization uses gene-specific probes and can determine the spatial patterns of gene expression precisely. Recent high-throughput investigations have yielded spatiotemporal information for thousands of genes in organisms such as *Drosophila* [33, 71] and mouse [245, 246]. These data have the potential to provide significant insights into the functions and interactions of genes [37, 247].

The *Drosophila melanogaster* is one of the model organisms in developmental biology, and its patterns of gene expression have been studied extensively [30, 31, 33, 71]. The comprehensive atlas of spatial patterns of gene expression during *Drosophila* embryogenesis has been created by *in situ* hybridization techniques, and the patterns are documented in the form of digital images [34–36, 71]. Comparative analysis of gene expression pattern images can potentially reveal new genetic interactions and yield insights into the complex regulatory networks governing embryonic development [37–39, 71].

To facilitate pattern comparison and searching, the images of *Drosophila* gene expression patterns are annotated with anatomical and developmental ontology terms using a controlled vocabulary [34, 71]. The basic requirement for annotation is to assign a unique term, not only for each terminally differentiated embryonic structure, but also for the developmental intermediates that correspond to it. Four general classes of terms, called anlage *in statu nascendi*, anlage, primordium, and organ (ordered in terms of developmental time), are used in the annotation. Such an elaborate naming scheme describes a developing "path", starting from the cellular blastoderm stage until organs are formed, that documents the dynamic process of *Drosophila* embryogenesis. Due to the overwhelming complexity of this task, the images

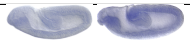
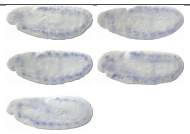
Stage range		BDGP terms
9-10		trunk mesoderm primordium anterior endoderm primordium posterior endoderm primordium inclusive hindgut primordium
11-12		embryonic central brain glia lateral cord glia neuroblasts of ventral nervous system procephalic neuroblasts

Figure 6.1: Sample image groups and their associated terms in the Berkeley *Drosophila* Genome Project (BDGP) database in two stage ranges. Only images taken from lateral view with the anterior to the left are shown.

are currently annotated manually by human experts. However, the number of available images produced by high-throughput *in situ* hybridization is now rapidly increasing [37, 38, 40–42]. It is therefore tempting to design computational methods for the automated annotation of gene expression patterns.

The automated annotation of *Drosophila* gene expression patterns was originally considered difficult due to the lack of a large reference data set from which to learn. Moreover, the "variation in morphology and incomplete knowledge of the shape and position of various embryonic structures" have made this task more elusive [71]. We attempt to address this problem by resorting to advanced tools developed recently in the computer vision and machine learning research communities and on the large set of annotated data available from the Berkeley *Drosophila* Genome Project (BDGP) [71]. There are several challenging questions that need to be addressed when approaching this problem by computational methods. As has been stated in [71], the first challenge is to deal with the issue that the same embryonic structure can appear in different shapes and positions due to the distortions caused by genetic variations and the image acquisition process. Fortunately, recent advances in object recognition research have led to robust methods that can detect regions of interest and extract features that are invariant to a class of local transformations from these regions. These two correlated lines of research have reached some maturity now (see [248] and [249] for an overview).

The second challenge of this task lies in the data representation. The embryogenesis of *D*rosophila has been divided into six discrete stage ranges (1-3, 4-6, 7-8, 9-10, 11-12, and

13-16) in the BDGP high-throughput study [71]. Gene expression patterns are documented collectively by a group of images in a specific stage range. Similarly, annotation terms are also associated with a group of patterns sharing a subset of the named structures (Fig. 6.1). These attributes of the existing biological data pose challenges, because traditional machine learning tools require that each object in question be represented by a feature vector of fixed length. It is challenging to encode the variable number of images in a group into a fixed-length vector. The existing approach [250] is based on the simplifying assumption that terms are associated with individual images instead of image groups. Kernel methods developed in machine learning are a class of versatile tools for learning from unconventional data types, since they only require that the similarity between objects be abstracted into the so-called kernel matrix [62]. Kernels between various data types, e.g., strings, trees, graphs, and sets of vectors, have been proposed in the literature [251–253]. We propose to extract a number of locally invariant features from each gene expression pattern image, and to compute kernels between sets of images based on the pyramid match algorithm [254].

A recent comprehensive study shows that when local features are used to compute kernels between images, a combination of multiple feature types tends to yield better results than even the most discriminative individual feature type [255]. This motivates us to extract multiple feature types from each image and obtain multiple kernel matrices, one for each feature type. Thus, the third challenge for automated gene expression pattern annotation is to develop methods that can combine the multiple kernel matrices effectively. Automated methods for combining multiple kernel matrices, called Multiple Kernel Learning (MKL), have been studied in machine learning recently. In such a framework, the optimal kernel matrix is obtained as a convex combination of a set of pre-defined candidate kernel matrices, and the coefficients for the combination can be computed by optimizing certain criterion. Methods for MKL have been proposed in the contexts of binary-class [256] and multi-class classification [257], and they have been applied successfully to various biological applications [258, 259]. For the problem of gene expression pattern annotation, a variable number of terms from the controlled vocabulary can be assigned to a group of patterns. Hence, this problem belongs to the more general framework of multi-label learning. Thus, the final challenge for automated gene expression pattern annotation is how to perform multi-label learning effectively and predict all relevant labels for a given

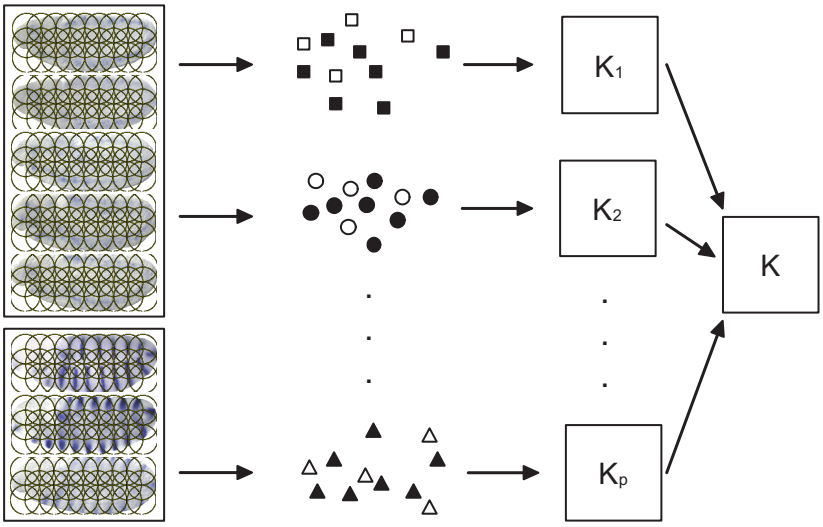


Figure 6.2: Illustration of the framework for annotating gene expression patterns. We extract multiple types of features from image groups and construct multiple kernels using the pyramid match algorithm. The multiple kernels are then combined to annotate the image groups. Different shapes represent different types of features, and filled and hollow shapes are used to distinguish features from the two different image groups.

image group. In this chapter, we propose to apply Hypergraph Spectra Learning (HSL) to project and combine the multiple kernel matrices for multi-label data. The overall flowchart of the proposed framework is depicted in Fig. 6.2.

The rest of this chapter is organized as follows. We discuss feature generation and kernel construction in Section 6.2. The formulation for multi-label multiple kernel learning is presented in Section 6.3. We report the results on gene expression pattern annotation in Section 6.4, and conclude this chapter in Section 6.5.

6.2 Feature Generation and Kernel Construction

In this section, we present our methods for extracting features from gene expression pattern images and constructing kernels between sets of patterns.

Feature Generation

There are two primary methods for extracting features. When the images are not well-aligned, the covariant region detector is first applied on the images to detect regions of interest. Then, local descriptor is used to extract features from the detected regions. An alternative approach is to apply local descriptor on a dense regular grid, instead of regions of interest [260, 261]. Such

an approach is motivated from the bag-of-words model from the text-modeling literature, and competitive performance has been achieved on image applications [262]. Since the images in our FlyExpress [35] database are already well-aligned, we take the second approach (Fig. 6.2). Instead of tuning the local descriptor and grid size manually, we apply several popular local descriptors on regular grids of different sizes, and rely on the MKL framework to select the appropriate local descriptors and grid size. More details on feature generation are described in Section 6.4.

Pyramid Match Kernels

In kernel methods, a symmetric function is called a "kernel function" if it satisfies the Mercer's condition [62]. When used for a finite number of samples in practice, this condition amounts to requiring that the kernel matrix is positive semidefinite. The wide applicability of kernel methods can be partially attributed to the fact that they only require the characterization of similarities between objects.

The pyramid match algorithm [253, 254, 260] computes kernels for variable-sized sets of feature vectors. The main idea of this approach is to convert sets of feature vectors to multi-dimensional, multi-resolution histograms, and then compute the similarity between the corresponding histograms based on histogram intersections. The final similarity between two sets of vectors is computed as a weighted sum of the similarities at the histogram levels. This similarity is an approximation to the similarity of the best partial matching between the feature sets. The resulting similarity matrix based on this measure is provably positive definite, and it can be used in existing kernel-based learning algorithms.

In the pyramid match algorithm, the feature space is partitioned into bins (regions) with increasingly larger granularity. At the finest level, all distinct vectors are guaranteed to be in different bins. The size of the bin increases gradually until a single bin occupies the entire feature space. Each partition of the feature space results in a histogram, and the different partitions lead to a hierarchy (pyramid) of histograms with increasingly coarser resolution. It follows from this construction that no vectors share the same bin at the finest level of resolution while all vectors share the same bin at the coarsest level. Hence, any two feature vectors from any two sets begin to share a bin at some level in this histogram pyramid, and they are considered to be matched at

this point. The distance between any two vectors can be bounded from above by the size of the bin in which they are matched. Thus the pyramid match algorithm can extract an approximate matching score between two sets of vectors without computing any of the pairwise similarities.

The original pyramid match algorithm proposed in [253] generates uniform bins over the feature space, and the size of the bin is doubled at each of the successive steps in the pyramid. Such a pre-determined construction fails to take advantage of the underlying structure in the feature space, and it suffers from distortion factors that increase linearly with the dimension of the features [254]. [254] proposed the vocabulary-guided pyramid match algorithm in which the positions and sizes of bins in the multi-resolution histograms are determined by applying the hierarchical clustering algorithm on the feature vectors. Each level in the hierarchical clustering corresponds to one level in the histogram pyramid, and the position and size of each bin at a particular level is determined by the clusters at that level. The weight for each match can also be made data-adaptive by estimating the inter-feature distance geometrically. It was shown that this data-dependent hierarchical decomposition scheme can maintain consistent accuracy when the dimension of the feature space increases [254].

The pyramid match algorithms proposed in [253, 254, 260] treat the sets of features to be matched as orderless. In some applications, the spatial layout of features within a set may convey critical discriminative information. [261] proposed the spatial pyramid matching algorithm to perform pyramid matching in the two-dimensional image space, thus taking the spatial information into account directly. The main idea of this approach is to quantize the local features in images into a number of discrete types by applying clustering algorithms, and then place multi-resolution histogram pyramid on the two-dimensional images. It is also possible to integrate geometric information directly into the original pyramid match algorithm by adding the image coordinates as two additional dimensions into each feature vector [261], and we adopt this approach. Note that the original pyramid match algorithms are proposed to match two images, and that we extend them to match two sets of images.

6.3 Multi-label Multiple Kernel Learning

In this section, we present a multi-label multiple kernel learning formulation based on hypergraph for integrating the kernel matrices derived from various local descriptors. Results in

Section 6.4 show that the integrated kernels yield better performance than that of the best individual kernel.

Kernel Hypergraph Spectral Learning

Hypergraph generalizes traditional graph by allowing edges, known as "hyperedges", to connect more than two vertices, thus capturing the joint relationship among multiple vertices. We propose to construct a hypergraph (for the collection of gene expression patterns in question) in which each pattern is represented as a vertex. To document the joint similarity among patterns annotated with a common term, we propose to construct a hyperedge for each term in the vocabulary, and include all patterns annotated with a common term into one hyperedge. Hence, the number of hyperedges in this hypergraph equals the number of terms in the vocabulary.

Laplacian is commonly used to learn from a graph [135]. To learn from a hypergraph, one can either define hypergraph Laplacian directly, or expand it into a traditional graph for which Laplacian is constructed. Since it has been shown that the Laplacians defined in both ways are similar [132], we use the expansion-based approaches. The star and clique expansions are two commonly-used schemes for expanding hypergraphs. Following the spectral graph embedding theory [135], we propose to project the patterns into a lower-dimensional space in which patterns sharing a common term are close to each other. Specifically, kernel hypergraph spectral learning solves the following optimization problem:

$$\begin{aligned} \max_{\mathbf{B}} \quad & \mathrm{Tr}\left(\mathbf{B}^T(\mathbf{KCK})\mathbf{B}\right) \\ \text{s.t.} \quad & \mathbf{B}^T(\mathbf{K}^2+\lambda\mathbf{K})\mathbf{B}=\mathbf{I}, \end{aligned} \tag{6.1}$$

where $\mathbf{K}\in\mathbb{R}^{n\times n}$ is the kernel matrix, n is the number of image sets, $\mathbf{C}=\mathbf{I}-\mathscr{L}$ in which $\mathscr{L}$ is the normalized Laplacian matrix derived from the hypergraph, $\mathbf{B}$ is the coefficient matrix for reconstructing the projection in feature space, and $\lambda>0$ is the regularization parameter.

Kernel Canonical Correlation Analysis (kCCA) [194] is a widely-used method for dimensionality reduction. It can be shown that kCCA involves the following optimization problem:

$$\begin{aligned} \max_{\mathbf{B}} \quad & \mathrm{Tr}\left(\mathbf{B}^T\mathbf{K}\left(\mathbf{Y}^T\left(\mathbf{YY}^T\right)^{-1}\mathbf{Y}\right)\mathbf{KB}\right) \\ \text{s.t.} \quad & \mathbf{B}^T\left(\mathbf{K}^2+\lambda\mathbf{K}\right)\mathbf{B}=\mathbf{I}, \end{aligned} \tag{6.2}$$

where $\mathbf{Y}$ is the label matrix. Thus, kCCA is a special case of kernel hypergraph spectral learning.

A Convex Formulation

It follows from the theory of reproducing kernels [62] that the kernel $\mathbf{K}$ in Eq. (6.1) uniquely determines a mapping of the patterns to some feature space. Thus, kernel selection (learning) is one of the central issues in kernel methods. Following the multiple kernel learning framework [256], we propose to obtain an optimal kernel matrix by integrating multiple kernel matrices constructed from various features, that is, $\mathbf{K} = \sum_{j=1}^{p} \theta_j \mathbf{K}_j$ where $\{\mathbf{K}_j\}_{j=1}^{p}$ are the p kernels constructed from various local descriptors and $\{\theta_j\}_{j=1}^{p}$ are the weights satisfying $\sum_{j=1}^{p} \theta_j \operatorname{Tr}(\mathbf{K}_j) = 1$. We show that the optimal weights that maximize the objective function in Eq. (6.1) can be obtained by solving a Semi-Infinite Linear Program (SILP) [263] in which a linear objective is optimized subject to an infinite number of linear constraints. This is summarized in the following theorem:

Theorem 6.1 *Given a set of p kernel matrices $\boldsymbol{K}_1, \ldots, \boldsymbol{K}_p$, the optimal kernel matrix, in the form of a linear combination of the given p kernel matrices that maximizes the objective function in Eq. (6.1), can be obtained by solving the following SILP problem:*

$$\begin{aligned} \max_{\theta,\gamma} \quad & \gamma && (6.3) \\ \text{s.t.} \quad & \theta \geq 0,\ \theta^T \boldsymbol{r} = 1, \\ & \sum_{j=1}^{p} \theta_j S_j(\Psi) \geq \gamma, \quad \textit{for all } \Psi \in \mathbb{R}^{n \times k}, && (6.4) \end{aligned}$$

where $S_j(\Psi)$, for $j = 1, \cdots, p$, is defined as

$$S_j(\Psi) = \sum_{i=1}^{k} \left(\frac{r_j}{4} \psi_i^T \psi_i + \frac{1}{4\lambda} \psi_i^T \boldsymbol{K}_j \psi_i - r_j \psi_i^T h_i \right), \tag{6.5}$$

$\Psi = [\psi_1, \ldots, \psi_k]$, $\boldsymbol{r} = (r_1, \cdots, r_p)^T$, *and* $r_j = \operatorname{Tr}(\boldsymbol{K}_j)$.

We show in the following lemma that maximization of the objective function in Eq. (6.1) is equivalent to the minimization of an alternative criterion.

Lemma 6.1 *The kernel matrix that maximizes the objective function in Eq. (6.1) is also the minimizer of the following objective function:*

$$F_1(\boldsymbol{K},\boldsymbol{B}) = \sum_{i=1}^{k} \Big(\|\boldsymbol{K}\beta_i - \boldsymbol{h}_i\|^2 + \lambda \beta_i^T \boldsymbol{K} \beta_i \Big), \tag{6.6}$$

where $\boldsymbol{B} = (\beta_1, \ldots, \beta_k)$, $\boldsymbol{h}_i$ *is the ith column of* $\boldsymbol{H}$ *where* $\boldsymbol{H}\boldsymbol{H}^T = \mathbf{C}$.

Proof Since the null space of $\mathbf{K}^2 + \lambda\mathbf{K}$ lies in the null space of $\mathbf{KCK}$, the optimal value achieved by the optimization problem in Eq. (6.1) is given by $\mathrm{Tr}\left((\mathbf{K}^2 + \lambda\mathbf{K})^{+}\mathbf{KCK}\right)$. Consider the maximization of the following objective function with respect to $\mathbf{B}$:

$$F_2(\mathbf{K},\mathbf{B}) = \sum_{i=1}^{k} \frac{\left(\beta_i^T \mathbf{K}\mathbf{H}_i\right)^2}{\beta_i^T \left(\mathbf{K}^2 + \lambda\mathbf{K}\right)\beta_i}. \tag{6.7}$$

The optimal β_i is given by $\beta_i^* = (\mathbf{K}^2 + \lambda\mathbf{K})^{+}\mathbf{K}\mathbf{h}_i$. Thus the maximum value of $F_2(\mathbf{B},\mathbf{K})$ achieved by $\mathbf{B}^* = [\beta_1^*, \ldots, \beta_k^*]$ is given by $F_2^*(\mathbf{K}) = \mathrm{Tr}\left((\mathbf{K}^2 + \lambda\mathbf{K})^{+}\mathbf{KCK}\right)$. This shows that, when optimized with respect to $\mathbf{B}$, the objective functions in Eqs. (6.1) and (6.7) achieve the same value.

We next show the equivalence between objective functions in Eqs. (6.6) and (6.7). The objective function $F_1(\mathbf{K},\mathbf{B})$ in Eq. (6.6) is the least squares cost function in the kernel-induced feature space, and its optimal value is

$$F_1^*(\mathbf{K}) = -\sum_{i=1}^{k} (\mathbf{K}\mathbf{h}_i)^T (\mathbf{K}^2 + \lambda\mathbf{K})^{+}\mathbf{K}\mathbf{h}_i + \mathbf{h}_i^T\mathbf{h}_i. \tag{6.8}$$

Thus maximizing $\sum_{i=1}^{k} (\mathbf{K}\mathbf{h}_i)^T (\mathbf{K}^2 + \lambda\mathbf{K})^{+}\mathbf{K}\mathbf{h}_i$ with respect to $\mathbf{K}$ is equivalent to minimizing $F_1^*(\mathbf{K})$. This completes the proof. ∎

Note that the matrix $\mathbf{C}$ obtained from standard clique and star expansions is symmetric and positive semidefinite. Thus the matrix $\mathbf{H}$ is always well-defined. We are now ready to prove Theorem 6.1.

Proof Let $\xi_i = \mathbf{K}\beta_i - \mathbf{h}_i$. Define the Lagrangian function for the optimization problem in Eq. (6.6) as follows:

$$L = \sum_{i=1}^{k} \|\xi_i\|^2 + \lambda \sum_{i=1}^{k} \beta_i^T \mathbf{K}\beta_i - \sum_{i=1}^{k} \psi_i^T (\mathbf{K}\beta_i - h_i - \xi_i), \tag{6.9}$$

where the ψ_i's are the vectors of Lagrangian dual variables. By following the standard Lagrangian technique, we get the Lagrangian dual function as

$$g(\psi_1,\ldots,\psi_k)=\sum_{i=1}^{k}\left(-\frac{1}{4}\psi_i^T\left(\mathbf{I}+\frac{1}{\lambda}\mathbf{K}\right)\psi_i+\Psi^T\mathbf{h}_i\right). \tag{6.10}$$

The optimal $\psi_1^*,\ldots,\psi_k^*$ can be obtained by maximizing the dual function. Since strong duality holds, the primal and dual objectives coincide and the optimal $\mathbf{K}$ can be computed by solving the following optimization problem:

$$\min_{\theta:\theta\geq 0,\theta^T\mathbf{r}=1}\quad \max_{\psi_1,\ldots,\psi_k}\left\{\sum_{i=1}^{k}\left(-\frac{1}{4}\psi_i^T\left(\mathbf{I}+\frac{1}{\lambda}\sum_{j=1}^{p}\theta_j\mathbf{K}_j\right)\psi_i+\psi_i^T\mathbf{h}_i\right)\right\}.$$

Since $\sum_{j=1}^{p}\theta_j r_j=1$, the above objective function can be expressed as

$$\sum_{j=1}^{p}\theta_j\sum_{i=1}^{k}\left(-\frac{r_j}{4}\psi_i^T\psi_i-\frac{1}{4\lambda}\psi_i^T\mathbf{K}_j\psi_i+r_j\psi_i^T\mathbf{h}_i\right),$$

Thus it follows from the definition of $S_j(\Psi)$ in Eq. (6.5) that the optimization problem in Eq. (6.3) can be expressed equivalently as

$$\max_{\theta:\theta\geq 0,\theta^T\mathbf{r}=1}\quad \min_{\Psi}\sum_{j=1}^{p}\theta_j S_j(\Psi). \tag{6.11}$$

Assume Ψ^* is the optimal solution to the problem in Eq. (6.11), and define $\gamma^*=\sum_{j=1}^{p}\theta_j S_j(\Psi^*)$ as the minimum value. We have $\sum_{j=1}^{p}\theta_j S_j(\Psi)\geq\gamma^*$ for allΨ. By defining $\gamma=\min_{\Psi}\sum_{j=1}^{p}\theta_j S_j(\Psi)$ and substituting γ into the objective, we prove this theorem. ∎

6.4 Results

In this section, we apply the proposed framework for annotating gene expression patterns. We use a collection of images obtained from the FlyExpress database [35], which contains standardized and aligned images. All the images used are taken from lateral view with the anterior to the left. The size of each raw image is 128×320.

Experimental Setup

We apply nine local descriptors on regular grids of two different sizes on each image. The nine local descriptors are SIFT, shape context, PCA-SIFT, spin image, steerable filters, differential invariants, complex filters, moment invariants, and cross correlation. These local descriptors are commonly used in objection recognition problems (more details can be found in [249]).

The sizes of the grids we used are 16 and 32 pixels in radius and spacing (Fig. 6.2), and 133 and 27 local features are produced for each image, respectively.

It is known that local textures are important discriminative features of gene expression pattern images, and features constructed from filter banks and raw pixel intensities are effective in capturing such information [264]. We therefore apply Gabor filters with different wavelet scales and filter orientations on each image to obtain global features of 384 and 2592 dimensions. We also sample the pixel values of each image using a bilinear technique, and obtain features of 10240, 2560, and 640 dimensions. The resulting features are called "global features".

After generating the features, we apply the vocabulary-guided pyramid match algorithm [254] to construct kernels between image sets. A total of 23 kernel matrices (2 grid sizes $\times$ 9 local descriptors $+$ 2 Gabor $+$ 3 pixel) are obtained. Then, the proposed MKL formulation is employed to obtain the optimal integrated kernel matrix. The performance of kernel matrices (either single or integrated) is evaluated by applying the Support Vector Machine (SVM) for each term and treating image sets annotated with this term as positive, and all other image sets as negative. We extract different numbers of terms from the FlyExpress database and use various numbers of image sets annotated with the selected terms for the experiments.

Precision and recall are two commonly-used criteria for evaluating the performance of multi-label classification systems [265]. For each term, let Π and Λ denote the indices of patterns that are annotated with this term by the proposed framework and by human curators in BDGP, respectively. Then, precision and recall for this term are defined to be $P = |\Pi \cap \Lambda|/|\Pi|$ and $R = |\Pi \cap \Lambda|/|\Lambda|$, respectively, where $|\cdot|$ denotes the set cardinality. The F1 score is the harmonic mean of precision and recall as $F1 = (2 \times P \times R)/(P+R)$. To measure performance across multiple terms, we use both the macro F1 (average of F1 across all terms) and the micro F1 (F1 computed from the sum of per-term contingency tables) scores, which are commonly used in text and image applications [265]. In each case, the entire data set is randomly partitioned into training and test sets with ratio 1:1. This process is repeated ten times, and the averaged performance is reported.

Table 6.1: Performance of integrated kernels on gene expression pattern annotation in terms of macro F1 score.

# of terms	10			20			30			40			50			60		
# of sets	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000
Star	**0.5661**	**0.5741**	**0.5434**	0.4396	0.4903	0.4575	0.3852	0.4437	0.4162	0.3768	0.4019	0.3927	0.3522	0.3850	0.3862	0.3219	0.3364	0.3426
Clique	0.5251	0.5220	0.4876	**0.4536**	**0.5125**	**0.4926**	**0.4065**	**0.4747**	**0.4563**	**0.4145**	**0.4346**	**0.4283**	**0.3872**	**0.4106**	**0.4198**	**0.3594**	**0.3631**	**0.3639**
kCCA	0.5487	0.5608	0.5323	0.3987	0.4635	0.4477	0.3497	0.4240	0.4063	0.3538	0.3872	0.3759	0.3303	0.3642	0.3666	0.2996	0.3137	0.3263
SVM1	0.4924	0.5413	0.5353	0.3780	0.4640	0.4356	0.3523	0.4352	0.4200	0.3741	0.4048	0.3955	0.3481	0.3869	0.3991	0.3316	0.3462	0.3570
Uniform	0.4947	0.5498	0.5418	0.3727	0.4703	0.4480	0.3513	0.4410	0.4191	0.3719	0.4111	0.3986	0.3436	0.3920	0.4023	0.3298	0.3548	0.3586
BIK	0.5418	0.5430	0.5185	0.4241	0.4515	0.4344	0.3782	0.4312	0.3996	0.3914	0.3954	0.3827	0.3701	0.3849	0.3763	0.3456	0.3448	0.3419
Z&P	0.3756	0.3810	0.3775	0.2695	0.2759	0.2804	0.2086	0.2470	0.2379	0.2117	0.2171	0.2310	0.1926	0.2284	0.2167	0.1764	0.1827	0.1679

Table 6.2: Performance of integrated kernels on gene expression pattern annotation in terms of micro F1 score.

# of terms	10			20			30			40			50			60		
# of sets	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000
Star	**0.5841**	**0.6011**	0.5728	0.4861	0.5199	0.4847	0.4472	0.4837	0.4473	0.4277	0.4470	0.4305	0.4168	0.4347	0.4212	0.4000	0.4171	0.3999
Clique	0.5424	0.5429	0.5079	**0.5039**	**0.5422**	**0.5247**	**0.4682**	**0.5127**	**0.4894**	**0.4610**	**0.4796**	**0.4660**	**0.4454**	**0.4546**	**0.4580**	**0.4314**	**0.4420**	**0.4251**
kCCA	0.5727	0.5922	0.5643	0.4581	0.4994	0.4887	0.4209	0.4737	0.4532	0.4095	0.4420	0.4271	0.4000	0.4241	0.4086	0.3778	0.4042	0.3920
SVM1	0.5290	0.5781	0.5786	0.4361	0.5024	0.4844	0.4239	0.4844	0.4632	0.4248	0.4570	0.4415	0.4095	0.4420	0.4429	0.3947	0.4234	0.4188
Uniform	0.5341	0.5870	**0.5837**	0.4390	0.5096	0.4975	0.4242	0.4939	0.4683	0.4268	0.4673	0.4492	0.4092	0.4518	0.4482	0.3999	0.4358	0.4226
BIK	0.5585	0.5650	0.5637	0.4614	0.4735	0.4562	0.4189	0.4484	0.4178	0.4100	0.4196	0.4009	0.3914	0.4051	0.3957	0.3869	0.3905	0.3781
Z&P	0.4031	0.4032	0.3796	0.3034	0.2985	0.2827	0.2612	0.2441	0.2125	0.2406	0.2310	0.2203	0.2203	0.2174	0.2114	0.1977	0.1826	0.1586

Table 6.3: Performance of integrated kernels on gene expression pattern annotation in terms of precision.

# of terms	10			20			30			40			50			60		
# of sets	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000
Star	0.5246	0.5141	0.4861	0.4629	0.5349	0.4842	0.4674	0.5533	0.5089	0.5122	0.5559	0.5510	0.4968	0.5611	0.5509	0.5256	0.5439	0.5614
Clique	0.4586	0.4375	0.3968	0.4531	0.5244	0.5053	0.4674	0.5510	0.5379	0.5219	0.5502	0.5660	0.5078	0.5433	0.5831	0.5240	0.5501	0.5665
kCCA	0.5448	0.5443	0.5230	0.4917	0.5737	0.5585	0.5056	0.6120	0.6102	0.5235	0.6116	0.6421	0.5124	0.6154	0.6139	0.5373	0.5894	0.6642
SVM1	0.5973	0.6163	0.5985	0.5387	0.6121	0.6211	0.5124	0.6323	0.6227	0.5253	0.6151	0.6476	0.5196	0.6126	0.6429	0.5176	0.5628	0.6427
Uniform	**0.6258**	**0.6462**	**0.6155**	**0.5691**	**0.6417**	**0.6495**	**0.5379**	**0.6576**	**0.6450**	**0.5596**	**0.6511**	**0.6766**	**0.5349**	**0.6504**	**0.6782**	**0.5625**	**0.5986**	**0.6717**
BIK	0.4956	0.4830	0.4687	0.4247	0.4994	0.4814	0.4265	0.5089	0.4779	0.4626	0.5200	0.5299	0.4470	0.5093	0.5519	0.4744	0.5125	0.573
Z&P	0.3298	0.3244	0.3182	0.2311	0.2455	0.2453	0.1897	0.2164	0.2106	0.1877	0.1958	0.2127	0.1765	0.2037	0.1976	0.1570	0.1627	0.1515

Table 6.4: Performance of integrated kernels on gene expression pattern annotation in terms of recall.

# of terms	10			20			30			40			50			60		
# of sets	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000	1000	1500	2000
Star	0.6482	0.6892	0.6694	0.5019	0.5303	0.5117	0.4033	0.4535	0.4338	0.3675	0.3901	0.3811	0.3346	0.3573	0.3606	0.2961	0.3124	0.3143
Clique	0.6331	0.6654	0.6527	0.5238	0.5649	0.5479	0.4284	0.4820	0.4636	0.4075	0.4210	0.4063	0.3701	0.3808	0.3834	0.3336	0.3323	0.3244
kCCA	0.5952	0.6285	0.5966	0.4111	0.4483	0.4259	0.3292	0.3854	0.3603	0.3171	0.3366	0.3174	0.2905	0.3075	0.3008	0.2575	0.2603	0.2597
SVM1	0.4890	0.5383	0.5211	0.3494	0.4237	0.3890	0.3178	0.3898	0.3665	0.3322	0.3528	0.3300	0.3035	0.3300	0.3298	0.2844	0.2945	0.2878
Uniform	0.4830	0.5378	0.5198	0.3403	0.4219	0.3948	0.3098	0.3919	0.3626	0.3218	0.3569	0.3285	0.2963	0.3325	0.3280	0.2789	0.3014	0.2873
BIK	**0.6625**	**0.6991**	**0.6954**	**0.5507**	0.5726	**0.5613**	**0.4648**	**0.5156**	**0.4983**	**0.4555**	**0.4764**	**0.4767**	**0.4337**	0.4638	**0.4678**	**0.4158**	**0.4335**	**0.4374**
Z&P	0.4990	0.5504	0.5881	0.4242	0.3990	0.4460	0.2599	0.3869	0.3990	0.3271	0.2985	0.3327	0.2783	0.3427	0.3107	0.2757	0.3023	0.3069

Annotation Results

We apply the proposed formulations (star, clique, and kCCA) to combine the various kernel matrices derived from different local descriptors. The performance of multiple kernel learning based on the soft margin 1-norm SVM (SVM1) criterion proposed in [256] is also reported. Since the SVM1 formulation is only applicable to binary-class problems, we apply the formulation for each term by treating image sets annotated with this term as positive, and all other image sets as negative. To demonstrate the effectiveness of the proposed formulation for integrating kernels, we also report results obtained by combining the candidate kernels with uniform weight, along with the performance of the best individual kernel (among the 23 kernels) for each data set. To compare with the existing method proposed in [250], we extract wavelet features from images and apply the min-redundancy max-relevance feature selection algorithm to select a subset of features. As was done in [250], we assign terms to individual images and apply linear discriminant analysis to annotate each image. Note that this setup does not consider the image group information and is the same as the one proposed in [250]. The annotation results measured by F1 score, precision, and recall are summarized in Tables 6.1–6.4.

In this experiment, we first select a number of terms and then extract certain number of image sets annotated with at least one of the selected terms. The number of terms used are 10, 20, 30, 40, 50, and 60, and the number of image sets used are 1000, 1500, and 2000 in each case. The first three rows in Tables 6.1–6.4 report the performance obtained by kernels combined with star expansion, clique expansion, and CCA, respectively. The fourth row presents the performance achieved by kernels combined with the soft margin 1-norm SVM (SVM1) formulation in which an optimal kernel is learned for each term separately. The fifth row shows the performance achieved by kernels combined from the candidate kernels with uniform weights. The performance of the best individual kernel (BIK) over all local descriptors and grid sizes on the same data set is reported in the sixth row. The results obtained by the method proposed in [250] are reported in the last row. The performance shown in Tables 6.1–6.4 is the averaged scores over ten random partitions of the entire data set into training and test sets with ratio 1:1.

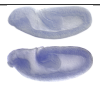
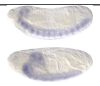
Stage range		BDGP terms	Predicted terms
1-3		maternal	maternal
4-6		cellular blastoderm	cellular blastoderm
7-8		trunk mesoderm anlage anterior endoderm anlage posterior endoderm anlage head mesoderm anlage	trunk mesoderm anlage anterior endoderm anlage posterior endoderm anlage
9-10		trunk mesoderm primordium anterior endoderm primordium posterior endoderm primordium inclusive hindgut primordium	trunk mesoderm primordium anterior endoderm primordium posterior endoderm primordium
11-12		embryonic central brain glia lateral cord glia neuroblasts of ventral nervous system procephalic neuroblasts	embryonic central brain glia lateral cord glia neuroblasts of ventral nervous system procephalic neuroblasts embryonic central brain neuron lateral cord neuron
13-16		embryonic central nervous system ventral nerve cord embryonic central brain neuron lateral cord neuron ventral midline lateral cord glia embryonic central brain glia	embryonic central nervous system ventral nerve cord embryonic central brain neuron lateral cord neuron lateral cord glia embryonic central brain glia embryonic central brain

Figure 6.3: Annotation results for sample patterns in the six stage ranges. BDGP terms denote terms that are assigned by human curators in the Berkeley *Drosophila* Genome Project [71], and predicted terms denote terms predicted by the proposed computational framework. These patterns are randomly sampled from each stage range, and hence they may not correspond to the same gene.

It can be observed from the results that in terms of both macro and micro F1 scores, the kernels integrated by either star or clique expansions achieve the highest performance on all but one of the data sets. This shows that the proposed formulation is effective in combining multiple kernels and potentially exploiting the complementary information contained in different kernels. For all data sets, the integrated kernels outperform the best individual kernel. In terms of precision and recall, our results indicate that SVM1 and Uniform achieve higher precision than the proposed formulations, while they both yield significantly lower recall. On the other hand, the best individual kernel produces slightly higher recall than the proposed formulations, while it yields significantly lower precision. Note that precision and recall are two competing criteria, and one can always achieve a perfect score on one of them at the price of the other. Hence, the proposed formulation achieves a harmonic balance between precision and recall, as indicated by the F1 scores. Note that BIK can have both higher precision and higher recall than

1 2 3 4 5

Figure 6.4: The original five images in stage range 13-16 from BDGP. The first and the third images are taken from lateral view; the second and the fourth images are taken from ventral view; the fifth image is taken from dorsal view. Only the first and the third images are used in our experiments shown in the bottom of Fig. 6.3.

the proposed formulation, since we report the highest precision and the highest recall among all of the candidate kernels separately. Hence, the BIK for precision and recall may not correspond to the same kernel. For all the four measures, the proposed formulations outperform the method proposed in [250] significantly. This shows that the annotation performance can be improved by considering the image group information.

Fig. 6.3 shows some annotation results obtained by clique expansion for sample patterns in each stage range. Note that the pyramid match algorithm can compute kernels between variable-sized sets of images. Thus, terms can be predicted for image sets of any size. Overall, the proposed computational framework achieves promising performance on annotating gene expression patterns. Meanwhile, we realize that the current framework suffers from some potential limitations. By comparing the BDGP terms and the predicted terms for patterns in stage ranges 7-8 and 9-10, we can see that the structures related to endoderm are predicted correctly while some of those related to mesoderm are prone to error. This may be due to the fact that, when viewed laterally, structures related to mesoderm are more prone to be hidden than those related to endoderm. This phenomenon becomes clearer when we examine the results for stage range 13-16 in Figs. 6.3 and 6.4. As shown in Fig. 6.4, there are a total of five images in this set in the original BDGP database. Among these five images, only two of them (the first and third) are taken from the lateral view and hence are used in our experiments. The second and the fourth images are taken from the ventral view, and the fifth image is taken from the dorsal view. The structure *ventral midline* can only be documented by digital images taken from the ventral view as can be seen from the second and the fourth images in Fig. 6.4. Since we only use images from the lateral view, it can be seen from Fig. 6.3 that the proposed framework can-

not predict this term correctly. This problem can potentially be solved by using images taken from other views such as ventral and dorsal. However, incorporation of images with multiple views may complicate the computational procedure, so requires a special care.

To evaluate the scalability of the proposed formulations, we vary the number of terms and the number of image sets, and compare the change of computation time. On a machine with Pentium 4 3.40 GHz CPU and 1 GB of RAM, when the number of terms is increased from 20 to 60 on a data set of 500 image sets, the computation time increases from approximately 4 seconds to 11 seconds. In terms of the number of image sets, data sets of 1500 and 2000 image sets with 60 terms take around 3 and 4 minutes, respectively.

6.5 Conclusions and Discussions

In this chapter, we have presented a computational framework for annotating gene expression patterns of *Drosophila*. We propose to extract invariant features from gene expression pattern images and construct kernels between these sets of features. To integrate multiple kernels effectively, we propose multi-label, multiple kernel learning formulations based on kernel hypergraph spectral learning. Experimental evaluation shows that the integrated kernels consistently outperform the best individual kernel and other representative methods. Currently, the annotation of patterns by human curators requires multiple passes, and the proposed framework could now be incorporated into the annotation process as a preprocessing step, which is then refined by human curators.

In future work, we plan to perform a detailed analysis of the weights obtained by the MKL formulation, and investigate how they are related to the relevance of each kernel. Our experimental results show that features extracted on smaller grids tend to yield better results. However, computational resource limitations prevent the use of a grid size smaller than 16 pixels. We plan to explore ways to overcome this problem. Retrieving gene expression patterns by combining information from images and annotations is an interesting and challenging research issue. The proposed framework can assign a probability of associating each term to each image, producing a probability vector for unannotated images from various high-throughput experiments. Such information can potentially be exploited to facilitate pattern retrieval. Detailed analysis of the annotation results produced by the proposed framework indicates that integra-

tion of gene expression pattern images taken from multiple views can potentially improve the annotation performance. In this case, the current pyramid match algorithms need to be adapted so that only images taken from the same view are matched. It can be seen from the third and fifth images in Fig. 6.4 that the annotation terms can also be associated with partial patterns in the images. These partial patterns have been removed in our FlyExpress database (Fig. 6.3), so these terms cannot be predicted correctly by the proposed framework. We plan to explore ways to incorporate these partial patterns in the future.

Chapter 7

CONCLUSIONS AND FUTURE WORK

7.1 Summary of Contributions

Multi-label learning concerns the classification of data instances which may be associated with multiple labels simultaneously. Compared with traditional binary and multi-class classification, multi-label learning is more general and has many real-world applications, e.g., text categorization, semantic scene classification, functional genomics and the *Drosophila* gene expression pattern image annotation. Multi-label learning is more challenging than multi-class classification due to possible overlapping between different labels. Similar to many machine learning and data mining tasks, multi-label learning also suffers from the curse of dimensionality. Although the dimensionality reduction is studied extensively in the literature, there is limited work on the dimensionality reduction for multi-label learning.

In this dissertation we concentrate on multi-label dimensionality reduction. One fundamental challenge in multi-label dimensionality reduction is how to effectively exploit the label structure to preserve label discriminatory information. In addition, as the data size increases, how to perform dimensionality reduction efficiently is critical in the practice use of multi-label dimensionality reduction. In this dissertation, we have made significant progress in addressing these two challenges. Specifically, we have investigated multi-label dimensionality reduction in the following three aspects: 1) the design and analysis of dimensionality reduction algorithms for multi-label learning; 2) efficient implementations of dimensionality reduction algorithms, including CCA, HSL, LDA and Orthonormalized PLS; and 3)the application of multi-label dimensionality reduction on the *Drosophila* gene expression pattern image annotation. In addition, we have developed a Matlab toolbox for multi-label dimensionality reduction. The toolbox integrates many popular dimensionality reduction algorithms in multi-label learning. In particular, it provides three different implementations and some extensions for them, including the direct least squares approach, the two-stage approach and the online algorithm.

In this dissertation we first analyzed and investigated the use of the classical dimensionality reduction algorithm Canonical Correlation Analysis (CCA) in multi-label learning. The effects of regularization on CCA and some important properties of CCA are elucidated in this

dissertation. The relationship and comparison between CCA and PLS are also studied in depth. To effectively capture the correlation among different labels in multi-label dimensionality reduction, we propose a novel multi-label dimensionality reduction algorithm called Hypergraph Spectral Learning (HSL) to deal with multi-label data. HSL is a rather general dimensionality reduction algorithm; for example, it can be shown that CCA is a special case of HSL by defining a specific Laplacian matrix.

Canonical Correlation Analysis (CCA) is a classical dimensionality reduction algorithm in statistics and machine learning. In this dissertation, the relationship between CCA and orthonormalized PLS has been investigated. Both CCA and PLS extract features from two sets of multi-dimensional variables. They can be applied in multi-label dimensionality reduction where the two sets of variables are derived from the data and the associated labels, respectively. By optimizing certain criteria, the data can be projected onto a lower-dimensional space directed by the label information. The fundamental difference between CCA and PLS is that CCA maximizes the correlation while PLS maximizes the covariance. We have revealed the equivalence relationship between CCA and orthonormalized PLS, a variant of PLS. We further extend the equivalence relationship to the case when regularization is employed for both sets of variables. In addition, we show that the CCA projection for one set of variables is independent of the regularization on the other set of variables. A sparse extension of CCA is also proposed in multi-label dimensionality reduction.

To effectively capture the correlation among different labels in multi-label dimensionality reduction, we propose hypergraph spectral learning. A hypergraph is a generalization of the traditional graph in which the edges are arbitrary non-empty subsets of the vertex set. It has been applied successfully to capture high-order relations in various domains. In hypergraph spectral learning, each data instance is mapped to a vertex in the hypergraph. Then we construct a hyperedge for each label and include all data points relevant to this label into the hyperedge. By exploiting the spectral property of the constructed hypergraph which encodes the correlation among different labels, the data are projected onto a lower-dimensional space. Intuitively, data points that share many common labels tend to be close to each other in the embedded space. In addition, different definitions of the hypergraph Laplacian can be applied in HSL, leading to

different dimensionality reduction algorithms. For example, CCA can be considered as special case of HSL by defining a specific Laplacian matrix for the hypergraph.

Many popular dimensionality reduction algorithms can be formulated as a Generalized Eigenvalue Problem (GEP), e.g., CCA and HSL. Although well-established algorithms in numerical linear algebra exist to solve generalized eigenvalue problems [172, 234], they are in general computationally expensive to solve and hence may not scale to large-size problems. To improve the scalability of these dimensionality reduction algorithms, we propose two different approaches for solving a class of dimensionality reduction algorithms, including CCA, Orthonormalized Partial Least Squares (OPLS), Linear Discriminant Analysis (LDA), and HSL. The first approach is a direct least squares approach which allows the use of different regularization penalties. However, it is applicable under a certain assumption. We further propose a scalable two-stage approach for the same class of dimensionality reduction algorithms. One appealing feature of the two-stage approach is that it can be applied in the regularization setting without any assumption.

The key idea of the proposed two efficient implementations is to transform the generalized eigenvalue problem into an ordinary least squares problem. The ordinary least squares which minimizes the sum-of-squares error function in Multivariate Linear Regression (MLR) is a classical technique for both regression and classification [173]. In the literature, numerous numerical algorithms have been proposed to solve the least squares efficiently [172], such as the conjugate gradient algorithm [266]. The least squares formulation can be readily extended using the regularization technique [79, 124, 126]. For example, the ℓ_1-norm regularization can be incorporated into the least squares formulation to improve the model sparsity and control the model complexity [124, 267]. Motivated by the mathematical and numerical properties of the generalized eigenvalue problem and the least squares formulation, several researchers have attempted to connect these two approaches. In particular, it has been shown that there is close relationship between LDA, CCA, and least squares [79, 138, 173, 219, 220, 268]. However, existing work either imposes strong assumption or is limited to specific dimensionality reduction algorithms.

In this dissertation, we first study the relationship between the least squares formulation and a class of dimensionality reduction algorithms involving the generalized eigenvalue problem, including CCA, OPLS, LDA and HSL. In particular, we establish the equivalence relationship between these two formulations under a mild condition. Based on the equivalent least squares formulation, we propose an efficient and scalable algorithm for these dimensionality reduction algorithms by applying the iterative conjugate gradient algorithm such as LSQR [221, 266]. In addition, we extend these dimensionality reduction algorithms by applying the ℓ_1-norm and ℓ_2-norm regularization to improve the model sparsity and the generalization ability.

However, the direct least squares formulation suffers from several drawbacks which limit its applicability in practice. First, the equivalence relationship relies on a key assumption that all the data points are linearly independent. This assumption tends to hold for high-dimensional data, but it is likely to fail for (relatively) lower-dimensional data. Secondly, the equivalence relationship does not hold when the regularization is employed. However, regularization has been shown to be critical in many data mining and machine learning algorithms including support vector machines [62].

In this dissertation, we further propose a two-stage approach for the same class of dimensionality reduction algorithms, including CCA, OPLS, LDA and HSL. In the first stage we solve a least squares problem using the iterative conjugate gradient algorithm [172,221,266]. The distinct property of this stage is its low time complexity. This stage is similar to the direct least squares formulation. In the second stage, the original data is projected onto a lower-dimensional space, and then we solve a generalized eigenvalue problem with a significantly reduced size. The proposed two-stage approach is shown to scale linearly in terms of both the sample size and the data dimensionality, thus it is applicable for large-size problems. We rigorously prove the equivalence relationship between the two-stage approach and the direct approach which solves the generalized eigenvalue problem directly. Compared with previous work, the two-stage approach does not require any assumption. In addition, we show that the two-stage approach can be further extended to the regularization setting. The equivalence relationship is also rigorously proved in this case.

The hypergraph spectral learning has been applied successfully in the *Drosophila* gene expression pattern image annotation. In this dissertation, we present a novel computational framework for annotating gene expression patterns using a controlled vocabulary to deal with the increasing number of pattern images generated by high-throughput *in situ* hybridization. The anatomical terms from the controlled vocabulary can be considered as labels, thus this problem can be modeled as a multi-label classification problem. However, annotation terms are assigned to groups of pattern images rather than to individual images in the currently available high-throughput data. To process groups of pattern images, we propose to extract invariant features from images, and construct pyramid match kernels to measure the similarity between sets of patterns. To exploit the complementary information conveyed by different features and incorporate the correlation among patterns sharing common structures, we propose efficient convex formulations to integrate the kernels derived from various features. The comparison between our automated annotation results and those of human curators shows promising performance of the proposed framework.

7.2 Future Work

One challenge in multi-label learning is how to deal with a large number of labels [53, 269]. Many popular multi-label algorithms assume that the number of labels is rather small [8]. In many applications, such as text categorization [19], protein function classification [43] and semantic annotation of multimedia [270], the label space is typically of high dimensionality (e.g., 10^3, 10^4). In addition, the computational cost of many algorithms becomes burden or even infeasible in this case. For example, the binary relevance approach [8, 55] is prohibitively expensive due to a large number of labels.

Although much work has been done on multi-label learning, how to deal with a large number of labels still remains an open question. Recently, some algorithms have been proposed to attack this problem by utilizing the underlying structure of the label space [43, 53, 59, 269]. For example, [269] transforms the multi-label classification task with a large number of labels into a tree-shaped hierarchy of simpler multi-label classification tasks which deals with a small number of labels. In [59], the singular value decomposition is used to reduce the dimensionality of the label space.

As discussed early, the label information for each instance can be encoded by a vector with length k, where k is the number of labels. In fact, although the total number of labels is very large, the actual labels are often sparse, i.e., the label vector for each instance has small support. In other words, the instances are only relevant to a small number of labels. Recent studies have shown that compressed sensing [65, 66, 271] is an ideal tool to process the sparse structures. Compressed sensing (CS), also known as compressive sampling, has recently received increasing attention in many areas of science and engineering [66]. Recent theoretical studies in CS show that one can recover certain sparse signals from far fewer samples or measurements than traditional methods [65, 272], thus the dimensionality can be reduced significantly using CS. For example, the dimensionality of label space is reduced using CS in the framework proposed in [53]

In the future, I plan to study multi-label learning by taking advantage of the underlying structure information in the label space. One promising direction is to effectively capture the label information by utilizing the structured sparsity in label space, such as group sparsity [273], and particularly the tree-structured sparsity structure [274, 275], to reduce the dimensionality of the label space. On the other hand, how to efficiently recover the label for the new instance in the original label space remains an open problem. Although the label dimensionality is reduced by using compressed sensing in the framework proposed in [53], the testing performance is still prohibitively expensive since we have to run the reconstruction algorithm, e.g., Orthogonal Matching Pursuit (OMP) [276], or Basis Pursuit (BP) [63], for each test instance to recover the true label in original label space. I plan to investigate efficient reconstruction algorithms to recover the true labels of test instances in the future.

[1] A. K. McCallum, "Multi-label text classification with a mixture model trained by EM," in *AAAI 99 Workshop on Text Learning*, 1999.

[2] M.-L. Zhang and Z.-H. Zhou, "Multilabel neural networks with applications to functional genomics and text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1338–1351, 2006.

[3] S. Ji, L. Sun, R. Jin, S. Kumar, and J. Ye, "Automated annotation of *Drosophila* gene expression patterns using a controlled vocabulary," *Bioinformatics*, vol. 24, no. 17, pp. 1881–1888, 2008.

[4] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[5] H. Blockeel, L. Schietgat, J. Struyf, S. Dzeroski, and A. Clare, "Decision trees for hierarchical multilabel classification: A case study in functional genomics," in *Knowledge Discovery in Databases (PKDD)*. Springer, 2006, vol. 4213, pp. 18–29.

[6] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Estimating 3D hand pose using hierarchical multi-label classification," *Image Vision Computing*, vol. 25, pp. 1885–1894, 2007.

[7] B. Jin, B. Muller, C. Zhai, and X. Lu, "Multi-label literature classification based on the gene ontology graph," *BMC Bioinformatics*, vol. 9, no. 1, p. 525, 2008.

[8] G. Tsoumakas, I. Katakis, and I. Vlahavas, *Data Mining and Knowledge Discovery Handbook*, 2nd ed. New York, NY: Springer, 2010, ch. Mining Multi-label Data, pp. 667–686.

[9] G. Tsoumakas and I. Katakis, "Multi label classification: An overview," *International Journal of Data Warehousing and Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[10] J. Read, "Scalable multi-label classification," Ph.D. dissertation, University of Waikato, Hamilton, New Zealand, 2010.

[11] R. E. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton, NJ: Princeton Universsity Press, 1961.

[12] K. Yu, S. Yu, and V. Tresp, "Multi-label informed latent semantic indexing," in *Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval (SIGIR)*, 2005, pp. 258–265.

[13] J. Arenas-Garca, K. Petersen, and L. Hansen, "Sparse kernel orthonormalized PLS for feature extraction in large data sets," in *Advances in Neural Information Processing Systems 19*, 2007, pp. 33–40.

[14] Y. Zhang and Z. Zhou, "Multi-label dimensionality reduction via dependence maximization," in *Proceedings of the 23rd national conference on artificial intelligence (AAAI)*, 2008, pp. 1503–1505.

[15] D. R. Hardoon and J. Shawe-Taylor, "KCCA for different level precision in content-based image retrieval," in *The 3rd International Workshop on Content-Based Multimedia Indexing*, 2003.

[16] T. Joachims and F. Sebastiani, "Guest editors' introduction to the special issue on automated text categorization," *Journal of Intelligent Information Systems*, vol. 18, no. 2-3, pp. 103–105, 2002.

[17] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, vol. 34, pp. 1–47, 2002.

[18] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proceedings of the 14th International Conference on Machine Learning (ICML)*, 1997, pp. 170–178.

[19] R. Schapire and Y. Singer, "Boostexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, pp. 135–168, 2000.

[20] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua, "A MFoM learning approach to robust multiclass multi-label text categorization," in *Proceedings of the 21st international conference on Machine learning (ICML)*, 2004, pp. 329–336.

[21] N. Ueda and K. Saito, "Parametric mixture models for multi-labeled text," in *Advances in Neural Information Processing Systems 16*, 2003, pp. 721–728.

[22] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda, "Maximal margin labeling for multi-topic text categorization," in *Advances in Neural Information Processing Systems 17*, 2004, pp. 649–656.

[23] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting,," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119 – 139, 1997.

[24] D. Lewis, Y. Yang, T. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[25] T. R. Hvidsten, J. Komorowski, A. K. Sandvik, and A. Laegreid, "Predicting gene function from gene expressions and ontologies." *Pacific Symposium on Biocomputing*, pp. 299–310, 2001.

[26] C. Lippert, Z. Ghahramani, and K. M. Borgwardt, "Gene function prediction from synthetic lethality networks via ranking on demand," *Bioinformatics*, vol. 26, no. 7, pp. 912–918, 2010.

[27] Y. Liu, "Yeast gene function prediction from different data sources: An empirical comparison," *The Open Bioinformatics Journal*, no. 5, pp. 69–76, 2011.

[28] A. Elisseeff and J. Weston, "A kernel method for multi-labeled classification," in *Advances in Neural Information Processing Systems 14*, 2001, pp. 681–687.

[29] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, and D. Haussler, "Knowledge-based analysis of microarray gene expression data by using support vector machines," *Proceedings of the National Academy of Sciences*, vol. 97, no. 1, pp. 262–267, 2000.

[30] J. A. Campos-Ortega and V. Hartenstein, *The Embryonic Development of* Drosophila Melanogaster, 2nd ed. New York, NY: Springer, 1997.

[31] M. N. Arbeitman and *et al.*, "Gene expression during the life cycle of *drosophilae melanogaster*," *Science*, vol. 297, no. 5590, pp. 2270–2275, 2002.

[32] P. Tomancak and *et al.*, "Systematic determination of patterns of gene expression during *drosophila* embryogenesis," *Genome Biology*, vol. 3, no. 12, 2002.

[33] E. Lécuyer, H. Yoshida, N. Parthasarathy, C. Alm, T. Babak, T. Cerovina, T. Hughes, P. Tomancak, and H. Krause, "Global analysis of mRNA localization reveals a prominent role in organizing cellular aarchitecture and function," *Cell*, vol. 131, pp. 174–187, 2007.

[34] G. Grumbling, V. Strelets, and The FlyBase Consortium, "FlyBase: anatomical data, images and queries," *Nucleic Acids Research*, vol. 34, pp. D484–488, 2006.

[35] B. Van Emden, H. Ramos, S. Panchanathan, S. Newfeld, and S. Kumar, "FlyExpress: An image-matching web-tool for finding genes with overlapping patterns of expression in *drosophila* embryos," Arizona State University, Tempe, AZ, Tech. Rep., 2006.

[36] C. Harmon and *et al.*, "Comparative analysis of spatial patterns of gene expression in *drosophila melanogaster* imaginal discs," in *RECOMB*, 2007, pp. 533–547.

[37] S. Kumar, K. Jayaraman, S. Panchanathan, R. Gurunathan, A. Marti-Subirana, and S. J. Newfeld, "BEST: a novel computational approach for comparing gene expression patterns from early stages of *drosophlia melanogaster* develeopment," *Genetics*, vol. 169, pp. 2037–2047, 2002.

[38] H. Peng and E. W. Myers, "Comparing *in situ* mRNA expression patterns of *drosophila* embryos," in *RECOMB*, 2004, pp. 157–166.

[39] B. Estrada and *et al.*, "An integrated strategy for analyzing the unique developmental programs of different myoblast subtypes," *PLoS Genetics*, vol. 2, no. 2,e16, pp. 160–171, 2006.

[40] R. Gurunathan, B. V. Emden, S. Panchanathan, and S. Kumar, "Identifying spatially similar gene expression patterns in early stage fruit fly embryo images: binary feature versus invariant moment digital representations," *BMC Bioinformatics*, vol. 5, no. 202, p. 13, 2004.

[41] J. Ye, J. Chen, Q. Li, and S. Kumar, "Classification of *drosophila* embryonic developmental stage range based on gene expression pattern images," in *CSB*, 2006, pp. 293–298.

[42] P. Tomancak, B. Berman, A. Beaton, R. Weiszmann, E. Kwan, V. Hartenstein, S. Celniker, and G. Rubin, "Global analysis of patterns of gene expression during *drosophila* embryogenesis," *Genome Biology*, vol. 8, no. 7, p. R145, 2007.

[43] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, 2001, pp. 42–53.

[44] S. Godbole and S. Sarawagi, "Discriminative methods for multi-labeled classification," in *Advances in Knowledge Discovery and Data Mining*, 2004, pp. 22–30.

[45] M. Zhang and Z. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[46] H. Wang, C. Ding, and H. Huang, "Multi-label classification: Inconsistency and class balanced k-nearest neighbor," in *AAAI Conference on Artificial Intelligence*, 2010.

[47] K. Crammer and Y. Singer, "A family of additive online algorithms for category ranking," *Journal of Machine Learning Research*, vol. 3, pp. 1025–1058, 2003.

[48] G. Tsoumakas and I. Vlahavas, "Random *k*-labelsets: An ensemble method for multilabel classification," in *Machine Learning: ECML 2007*, 2007, vol. 4701, pp. 406–417.

[49] R. Yan, J. Tesic, and J. Smith, "Model-shared subspace boosting for multi-label classification," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007, pp. 834–843.

[50] A. Esuli, T. Fagni, and F. Sebastiani, "Boosting multi-label hierarchical text categorization," *Information Retrieval*, vol. 11, pp. 287–313, 2008.

[51] F. Kang, R. Jin, and R. Sukthankar, "Correlated label propagation with application to multi-label learning," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2006, pp. 1719 – 1726.

[52] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM)*, 2005, pp. 195–200.

[53] D. Hsu, S. K., J. Langford, and T. Zhang, "Multi-label prediction via compressed sensing," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 772–780.

[54] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2010, pp. 999–1008.

[55] K. Brinker, J. Fürnkranz, and E. Hüllermeier, "A unified model for multilabel classification and ranking," in *Proceeding of the 17th European Conference on Artificial Intelligence (ECAI)*, 2006.

[56] L. Sun, S. Ji, and J. Ye, "Hypergraph spectral learning for multi-label classification," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2008, pp. 668–676.

[57] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, "Multi-label learning with weak label," in *AAAI Conference on Artificial Intelligence*, 2010.

[58] C. Silla and A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, pp. 31–72, 2011.

[59] F. Tai and H.-T. Lin, "Multi-label classification with principle label space transformation," in *The 2nd International Workshop on learning from Multi-Label Data*, 2010.

[60] L. Tenenboim-Chekina, L. Rokach, and B. Shapira, "Identification of label dependencies for multi-label classification," in *The 2nd International Workshop on learning from Multi-Label Data*, 2010.

[61] S. Vogrincic and Z. Bosnic, "Ontology-based multi-label classification of economic articles," *Computer Science and Information Systems*, vol. 8, no. 1, pp. 101–119, 2011.

[62] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Cambridge, MA: MIT Press, 2002.

[63] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[64] S. Gao, W. Wu, C.-H. Lee, and T.-S. Chua, "A maximal figure-of-merit learning approach to text categorization," in *Proceedings of the 26th annual international ACM*

SIGIR conference on Research and development in informaion retrieval (SIGIR), 2003, pp. 174–181.

[65] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[66] E. Candès, "Compressive sampling," in *International Congress of Mathematics*, no. 3, Madrid, Spain, 2006, pp. 1433–1452.

[67] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intellegence*, vol. 89, no. 1-2, pp. 31–71, 1997.

[68] Y. Chen and J. Z. Wang, "Image categorization by learning and reasoning with regions," *Journal of Machine Learning Research*, vol. 5, pp. 913–939, 2004.

[69] O. Maron and A. L. Ratan, "Multiple-instance learning for natural scene classification," in *Proceedings of the 15th International Conference on Machine Learning (ICML)*, 1998, pp. 341–349.

[70] Z.-H. Zhou and M.-L. Zhang, "Multi-instance multi-label learning with application to scene classification," in *Advances in Neural Information Processing Systems 20*, 2007, pp. 1609–1616.

[71] P. Tomancak, A. Beaton, R. Weiszmann, E. Kwan, S. Shu, S. Lewis, S. Richards, M. Ashburner, V. Hartenstein, S. Celniker, and G. Rubin, "Systematic determination of patterns of gene expression during *Drosophila* embryogenesis," *Genome Biology*, vol. 3, no. 12, pp. research0088.1–0088.14, 2002.

[72] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.

[73] D. W. Mount, *Bioinformatics: Sequence and Genome Analysis*, 2nd ed. Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press, 2004.

[74] S. M. Weiss, N. Indurkhya, and T. Zhang, *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Berlin: Springer, 2004.

[75] G. Alexander and E. Reiman, *Dementias: Diagnosis, Treatment and Research*, 3rd ed. American Psychiatric Pub, 2003, ch. Neuroimaging.

[76] C. R. Genovese, N. A. Lazar, and T. Nichols, "Thresholding of statistical maps in functional neuroimaging using the false discovery rate," *NeuroImage*, vol. 15, no. 4, pp. 870 – 878, 2002.

[77] M. J. Marton, J. L. DeRisi, H. A. Bennett, V. R. Iyer, M. R. Meyer, C. J. Roberts, R. Stoughton, J. Burchard, D. Slade, H. Dai, D. E. B. Jr., L. H. Hartwell, P. O. Brown, and S. H. Friend, "Drug target validation and identification of secondary drug target effects using dna microarrays," *Nature Medicine*, vol. 11, no. 4, pp. 1293 – 1301, 1998.

[78] D. L. Donoho, "High-dimensional data analysis: the curses and blessings of dimensionality," in *American Mathematical Society Conf. Math Challenges of the 21st Century*, 2000.

[79] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., ser. Springer Series in Statistics. New York, NY: Springer, 2009.

[80] S. Kaski and J. Peltonen, "Dimensionality reduction for data visualization," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 100–104, 2011.

[81] J. Venna and S. Kaski, "Comparison of visualization methods for an atlas of gene expression data sets," *Information Visualization*, vol. 6, pp. 139–154, 2007.

[82] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, "Information retrieval perspective to nonlinear dimensionality reduction for data visualization," *Journal of Machine Learning Research*, vol. 11, pp. 451–490, 2010.

[83] C. Bartenhagen, H.-U. Klein, C. Ruckert, X. Jiang, and M. Dugas, "Comparative study of unsupervised dimension reduction techniques for the visualization of microarray gene expression data," *BMC Bioinformatics*, vol. 11, no. 1, p. 567, 2010.

[84] C. J. C. Burges, "Geometric methods for feature extraction and dimensional reduction - a guided tour," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds. New York, NY: Springer, 2005, pp. 59–92.

[85] M. A. Carreira-Perpinan, "Continuous latent variable models for dimensionality reduction and sequential data reconstruction," Ph.D. dissertation, University of Sheffield, Sheffield, U.K., 2001.

[86] L. Saul, K. Weinberger, J. Ham, F. Sha, and D. Lee, *Semisupervised Learning*. Cambridge, MA: MIT Press, 2006, ch. Spectral methods for dimensionality reduction, pp. 293–308.

[87] J. A. Lee and M. Verleysen, *Nonlinear dimensionality reduction*. New York, NY: Springer, 2007.

[88] I. Jolliffe, *Principal Component Analysis*, 2nd ed., ser. Springer Series in Statistics. New York, NY: Springer, 2002.

[89] R. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, pp. 179–188, 1936.

[90] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. New York, NY: Academic Press, 1990.

[91] L. J. P. van der Maaten, E. O. Postma, and H. J. van den Herik, "Dimensionality reduction: A comparative review," Maastricht University, Tech. Rep., 2007.

[92] M. Balasubramanian, E. L. Schwartz, J. B. Tenenbaum, V. de Silva, and J. C. Langford, "The Isomap Algorithm and Topological Stability," *Science*, vol. 295, no. 5552, p. 7a, 2002.

[93] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[94] H. Hotelling, "Relations between two sets of variables," *Biometrika*, vol. 28, pp. 312–377, 1936.

[95] H. Wold, "Estimation of principal components and related models by iterative least squares," in *Multivariate Analysis*, P. Krishnaiaah, Ed. New York, NY: Academic Press, 1966, pp. 391–420.

[96] J. Wegelin, "A survey of partial least squares (PLS) methods, with emphasis on the two-block case," University of Washington, Tech. Rep., 2000.

[97] R. Rosipal and N. Krämer, "Overview and recent advances in partial least squares," in *Subspace, Latent Structure and Feature Selection Techniques, Lecture Notes in Computer Science*, C. Saunders, M. Grobelnik, S. Gunn, and J. Shawe-Taylor, Eds. Berlin: Springer, 2006, pp. 34–51.

[98] S. Wold and *et al.*, *Chemometrics, mathematics and statistics in chemistry*. Dordrecht, Holland: Reidel Publishing Company, 1984.

[99] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society of Information Science*, vol. 41, pp. 391–407, 1990.

[100] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization." *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[101] S. Sra and I. S. Dhillon, "Nonnegative matrix approximation: Algorithms and applications," Computer Sciences, University of Texas at Austin, Tech. Rep. TR-06-27, 2006.

[102] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[103] F. Bach and M. M. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2003.

[104] E. G. Learned-Miller and J. W. F. III, "ICA using spacings estimates of entropy," *Journal of Machine Learning Research*, vol. 4, pp. 1271–1295, 2003.

[105] T. Cox and M. Cox, *Multidimensional Scaling*, 2nd ed. Boca Raton, FL: Chapman and Hall/CRC, 2000.

[106] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[107] D. L. Donoho and C. Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.

[108] J. Sun, S. Boyd, L. Xiao, and P. Diaconis, "The fastest mixing markov process on a graph and a connection to a maximum variance unfolding problem," *SIAM Review*, vol. 48, no. 4, pp. 681–699, 2006.

[109] K. Weinberger and L. Saul, "Unsupervised learning of image manifolds by semidefinite programming," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2004, pp. 988–995.

[110] Z. Zhang and H. Zha, "Principal manifolds and nonlinear dimensionality reduction via tangent space alignment," *SIAM Journal on Scientific Computing*, vol. 26, pp. 313–338, 2005.

[111] A. Efros, V. Isler, J. Shi, and M. Visontai, "Seeing through water," in *Neural Information Processing Systems 17*, 2004, pp. 393–400.

[112] A. Elgammal and C.-S. Lee, "Separating style and content on a nonlinear manifold," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2004, pp. I–478–485.

[113] R. Pless, "Image spaces and video trajectories: using isomap to explore video sequences," in *Proceedings. of the 9th IEEE International Conference on Computer Vision (ICCV)*, 2003, pp. 1433–1440.

[114] A. Shashua and A. Levin, "Linear image coding for regression and classification using the tensor-rank principle," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001, pp. 42–49.

[115] A. Torralba and A. Oliva, "Semantic organization of scenes using discriminant structural templates," in *The Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV)*, 1999, pp. 1253–1258.

[116] M. Vasilescu and D. Terzopoulos, "Multilinear subspace analysis of image ensembles," in *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2003, pp. 93–99.

[117] J. Yang, D. Zhang, A. Frangi, and J. yu Yang, "Two-dimensional pca: a new approach to appearance-based face representation and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.

[118] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[119] S. Huang, M. Ward, and E. Rundensteiner, "Exploration of dimensionality reduction for text visualization," in *Proceedings of the 3rd International Conference on Coordinated and Multiple Views in Exploratory Visualization (CMV)*, 2005, pp. 63–74.

[120] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, "Multilabel classification of music into emotions," in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, 2008, pp. 325–330.

[121] M. Pintore, H. van de Waterbeemd, N. Piclin, and J. R. Chrétien, "Prediction of oral bioavailability by adaptive fuzzy partitioning," *European Journal of Medicinal Chemistry*, vol. 38, no. 4, pp. 427–431, 2003.

[122] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the 14th International Conference on Machine Learning (ICML)*, 1997, pp. 412–420.

[123] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge University Press, 2008.

[124] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[125] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of Computational and Graphical Statistics*, vol. 15, no. 2, pp. 265–286, 2006.

[126] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal Of the Royal Statistical Society: Series B*, vol. 67, no. 2, pp. 301–320, 2005.

[127] C. H. Park and M. Lee, "On applying linear discriminant analysis for multi-labeled problems," *Pattern Recognition Letters*, vol. 29, pp. 878–887, 2008.

[128] A. Gretton, O. Bousquet, A. J. Smola, and B. Schölkopf, "Measuring statistical dependence with Hilbert-Schmidt norms," in *Algorithmic Learning Theory*, S. Jain, H. Simon, and E. Tomita, Eds. Berlin: Springer, 2005, pp. 63–77.

[129] H. Hotelling, "Relations between two sets of variables," *Biometrika*, vol. 28, pp. 312–377, 1936.

[130] L. Sun, S. Ji, S. Yu, and J. Ye, "On the equivalence between canonical correlation analysis and orthonormalized partial least squares," in *Proceedings of the 21st international jont conference on Artifical intelligence (IJCAI)*, 2009.

[131] L. Sun, S. Ji, and J. Ye, "Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 194–200, 2011.

[132] S. Agarwal, K. Branson, and S. Belongie, "Higher order learning with graphs," in *Proceedings of the 23rd International Conference on Machine Learning (ICML)*, 2006, pp. 17–24.

[133] C. Berge, *Graphs and Hypergraphs*. New York, NY: Elsevier, 1973.

[134] ——, *Hypergraphs: The Theory of Finite Sets*. Amsterdam, Netherlands: North-Holland, 1989.

[135] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.

[136] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*. New York, NY: Halsted Press, 1992.

[137] D. Watkins, *Fundamentals of matrix computations*. New York, NY: John Wiley & Sons, Inc., 1991.

[138] L. Sun, S. Ji, and J. Ye, "A least squares formulation for canonical correlation analysis," in *Proceedings of the 25th International Conference on Machine Learning (ICML)*, 2008, pp. 1024–1031.

[139] ——, "A least squares formulation for a class of generalized eigenvalue problems in machine learning," in *Proceedings of the 26th International Conference on Machine Learning (ICML)*, 2009, pp. 977–984.

[140] L. Sun, B. Ceran, and J. Ye, "A scalable two-stage approach for a class of dimensionality reduction techniques," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2010, pp. 313–322.

[141] A.-L. Boulesteix and K. Strimmer, "Partial least squares: a versatile tool for the analysis of high-dimensional genomic data," *Briefings in Bioinformatics*, vol. 8, no. 1, pp. 32–44, 2007.

[142] H. Wold, *Encyclopedia of Statistical Sciences*. New York, NY: Wiley, 1985, vol. 6, ch. Partial least squares, pp. 581–591.

[143] D. Nguyen and D. Rocke, "Tumor classification by partial least squares using microarray gene expression data," *Bioinformatics*, vol. 18, no. 1, pp. 39–50, 2002.

[144] S. Rännar, F. Lindgren, P. Geladi, and S. Wold, "A PLS kernel algorithm for data sets with many variables and fewer objects. Part 1: Theory and algorithm," *Journal of Chemometrics*, vol. 8, no. 2, pp. 111–125, 1994.

[145] M. Barker and W. Rayens, "Partial least squares for discrimination," *Journal of Chemometrics*, vol. 17, no. 3, pp. 166–173, 2003.

[146] N. Krämer, "An overview on the shrinkage properties of partial least squares regression," *Computational Statistics*, vol. 22, no. 2, pp. 249–273, 2007.

[147] R. Rosipal, L. Trejo, and B. Matthews, "Kernel PLS-SVC for linear and nonlinear classification," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 2003, pp. 640–647.

[148] H. Wold, "Path models with latent variables: The nipals approach," in *Quantitative Sociology: International Perspectives on Mathematical and Statistical Model Building*, B. et al., Ed. Academic Press, 1975, pp. 307–357.

[149] A. Höskuldsson, "PLS regression methods," *Journal of Chemometrics*, vol. 2, no. 3, pp. 211–228, 1988.

[150] K. Worsley, J.-B. Poline, K. J. Friston, and A. Evans, "Characterizing the response of PET and fMRI data using multivariate linear models," *Neuroimage*, vol. 6, no. 4, pp. 305–319, 1997.

[151] P. Sampson, A. Streissguth, H. Barr, and F. Bookstein, "Neurobehavioral effects of prenatal alcohol: Part II. partial least squares analysis," *Neurotoxicology and Teratology*, vol. 11, no. 5, pp. 477 – 491, 1989.

[152] S. De Jong, "SIMPLS: an alternative approach to partial least squares regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 18, no. 3, pp. 251–263, 1993.

[153] X. Huang, W. Pan, S. Park, X. Han, L. Miller, and J. Hall, "Modeling the relationship between LVAD support time and gene expression changes in the human heart by penalized partial least squares," *Bioinformatics*, vol. 20, no. 6, pp. 888–894, 2004.

[154] H. Saigo, N. Krämer, and K. Tsuda, "Partial least squares regression for graph mining," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, 2008, pp. 578–586.

[155] A.-L. Boulesteix and K. Strimmer, "Predicting transcription factor activities from combined analysis of microarray and chip data: a partial least squares approach," *Theoretical Biology and Medical Modelling*, vol. 2, no. 1, p. 23, 2005.

[156] J. Arenas-Garcia and G. Camps-Valls, "Efficient kernel orthonormalized PLS for remote sensing applications," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 10, pp. 2872–2881, 2008.

[157] K. Cao, D. Rossouw, C. Robert-Grani, and P. Besse, "A sparse PLS for variable selection when integrating omics data," *Statistical Applications in Genetics and Molecular Biology*, vol. 7, no. 1, p. 35, 2008.

[158] H. Chun and S. Keles, "Sparse partial least squares regression for simultaneous dimension reduction and variable selection," *Journal of the Royal Statistical Society: Series B*, vol. 72, no. 1, pp. 3–25, 2010.

[159] ——, "Expression Quantitative Trait Loci Mapping With Multivariate Sparse Partial Least Squares Regression," *Genetics*, vol. 182, no. 1, pp. 79–90, 2009.

[160] A.-L. Boulesteix, "PLS dimension reduction for classification with microarray data," *Statistical Applications in Genetics and Molecular Biology*, vol. 3, no. 1, p. 33, 2004.

[161] V. Pihur, S. Datta, and S. Datta, "Reconstruction of genetic association networks from microarray data: a partial least squares approach," *Bioinformatics*, vol. 24, no. 4, pp. 561–568, 2008.

[162] M. Momma and K. Bennett, "Sparse kernel partial least squares regression," in *Proceedings of the 16th Annual Conferemce PM Computational Learning Theory*, 2003, pp. 216–230.

[163] R. Rosipal and L. Trejo, "Kernel partial least squares regression in reproducing kernel Hilbert space," *Journal of Machine Learning Research*, vol. 2, pp. 97–123, 2002.

[164] A. Tenenhaus, A. Giron, E. Viennet, M. Bera, G. Saporta, and B. Fertil, "Kernel logistic PLS: A tool for supervised nonlinear dimensionality reduction and binary classification," *Computational Statistics & Data Analysis*, vol. 51, no. 9, pp. 4083–4100, 2007.

[165] N. Krämer, M. Sugiyama, and M. Braun, "Lanczos approximations for the speedup of kernel partial least squares regression," in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 5, 2009, pp. 288–295.

[166] K. Fukunaga, *Introduction to statistical pattern recognition*, 2nd ed. San Diego, CA: Academic Press Professional, Inc., 1990.

[167] I. Frank and J. Friedman, "A statistical view of some chemometrics regression tools," *Technometrics*, vol. 35, no. 2, pp. 109–135, 1993.

[168] P. Geladi, "Notes on the history and nature of partial least squares (PLS) modelling," *Journal of Chemometrics*, vol. 2, no. 4, pp. 231–246, 1988.

[169] I. Helland, "On the structure of partial least squares regression," *Communications in Statistics - Simulation and Computation*, vol. 17, no. 2, pp. 581–607, 1988.

[170] C. Ter Braak and S. De Jong, "The objective function of partial least squares regression," *Journal of Chemometrics*, vol. 12, no. 1, pp. 41–54, 1998.

[171] A. Phatak and F. de Hoog, "Exploiting the connection between PLS, Lanczos methods and conjugate gradients: alternative proofs of some properties of PLS," *Journal of Chemometrics*, vol. 16, no. 7, pp. 361–367, 2002.

[172] G. Golub and C. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: Johns Hopkins Press, 1996.

[173] C. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

[174] S. De Jong, "PLS shrinks," *Journal of Chemometrics*, vol. 9, no. 4, pp. 323–326, 1995.

[175] O. Lingjaerde, "Shrinkage structure of partial least squares," *Scandinavian Journal of Statistics*, vol. 27, no. 3, pp. 459–473, 2000.

[176] N. Butler and M. Denham, "The peculiar shrinkage properties of partial least squares regression," *Journal of the Royal Statistical Society: Series B*, vol. 62, no. 3, pp. 585–593, 2000.

[177] C. Goutis, "Partial least squares algorithm yields shrinkage estimators," *The Annals of Statistics*, vol. 24, no. 2, pp. 816–824, 1996.

[178] M. Bartlett, "Further aspects of the theory of multiple regression," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 34, no. 1, pp. 33–40, 1938.

[179] Y. Liu and W. Rayens, "PLS and dimension reduction for classification," *Computational Statistics*, vol. 22, no. 2, pp. 189–208, 2007.

[180] J. Gottfries, K. Blennow, A. Wallin, and C. Gottfries, "Diagnosis of dementias using partial least squares discriminant analysis," *Dementia*, vol. 6, pp. 83–88, 1995.

[181] R. Briandet, E. Kemsley, and R. Wilson, "Discrimination of arabica and robusta in instant coffee by fourier transform infrared spectroscopy and chemometrics," *Journal of Agricultural and Food Chemistry*, vol. 44, no. 1, pp. 170–174, 1996.

[182] E. Sksjrvi, M. Khalighi, and P. Minkkinen, "Waste water pollution modelling in the southern area of lake saimaa, Finland, by the simca pattern recognition method," *Chemometrics and Intelligent Laboratory Systems*, vol. 7, no. 1-2, pp. 171 – 180, 1989.

[183] P. Berntsson and S. Wold, "Comparison between x-ray crystallographic data and physicochemical parameters with respect to their information about the calcium channel antagonist activity of 4-phenyl-1,4-dihydropyridines," *Quantitative Structure-Activity Relationships*, vol. 5, no. 2, pp. 45–50, 1986.

[184] S. Delwiche, Y. Chen, and W. Hruschka, "Differentiation of hard red wheat by near-infrared analysis of bulk samples," *Cereal Chemistry*, vol. 72, no. 3, pp. 243–247, 1995.

[185] E. Sundbom, O. Bodlund, and T. Hjerback, "Object relation and defensive operations in transsexuals and borderline patients as measured by the defense mechanism test," *Nordic Journal of Psychiatry*, vol. 49, no. 5, pp. 379–388, 1995.

[186] M. Ortiz, L. Sarabia, C. Symington, F. Santamaria, and M. Inigueze, "Analysis of ageing and typification of vintage ports by partial least squares and soft independent modelling class analogy," *Analyst*, vol. 121, no. 8, pp. 1009–1013, 1996.

[187] K. Iizuka and T. Aishima, "Soy sauce classification by geographic region based on nir spectra and chemometrics pattern recognition," *Journal of Food Science*, vol. 62, no. 1, pp. 101–104, 1997.

[188] R. Vong, P. Geladi, S. Wold, and K. Esbensen, "Source contributions to ambient aerosol calculated by discriminat partial least squares regression (PLS)," *Journal of Chemometrics*, vol. 2, no. 4, pp. 281–296, 1988.

[189] H. D. Vinod, "Canonical ridge and econometrics of joint production," *Journal of Econometrics*, vol. 4, no. 2, pp. 147–166, 1976.

[190] M. Borga, T. Landelius, and H. Knutsson, "A unified approach to PCA, PLS, MLR and CCA," Computer Vision Laboratory, Linkõping University, SE-581 83 Linköping, Sweden, Report, 1997.

[191] R. Johnson, *Applied multivariate statistical analysis*, 4th ed., D. Wichern, Ed. Upper Saddle River, NJ: Prentice Hall, 1998.

[192] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. New York, NY: Cambridge University Press, 2004.

[193] P. Koehn, "Europarl: a parallel corpus for statistical machine translation," in *MT Summit*, 2005, pp. 79–86.

[194] D. R. Hardoon, S. Szedmak, and J. Shawe-taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[195] J.-P. Vert and M. Kanehisa, "Graph-driven feature extraction from microarray data using diffusion kernels and kernel cca," in *Advances in Neural Information Processing Systems 15*, 2003, pp. 1425–1432.

[196] K. Le Cao, P. Martin, C. Robert-Granie, and P. Besse, "Sparse canonical methods for biological data integration: application to a cross-platform study," *BMC Bioinformatics*, vol. 10, no. 1, p. 34, 2009.

[197] P. Rai and H. Daume, "Multi-label prediction via sparse infinite cca," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds., 2009, pp. 1518–1526.

[198] D. Witten, R. Tibshirani, and T. Hastie, "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis," *Biostatistics*, vol. 10, no. 3, pp. 515–534, 2009.

[199] F. Bach and M. Jordan, "A probabilistic interpretation of canonical correlation analysis," University of California, Berkeley, Tech. Rep., 2005.

[200] S. Waaijenborg, P. Verselewel, and A. Zwinderman, "Quantifying the association between gene expressions and dna-markers by penalized canonical correlation analysis," *Statistical Applications in Genetics and Molecular Biology*, vol. 7, no. 1, 2008.

[201] S. Waaijenborg and A. Zwinderman, "Penalized canonical correlation analysis to quantify the association between gene expression and dna markers," *BMC Proceedings*, vol. 1, no. Suppl 1, p. S122, 2007.

[202] E. Parkhomenko, D. Tritchler, and J. Beyene, "Sparse canonical correlation analysis with application to genomic data integration," *Statistical Applications in Genetics and Molecular Biology*, vol. 8, no. 1, p. 1, 2009.

[203] S. Yu, K. Yu, V. Tresp, and H.-P. Kriegel, "Multi-output regularized feature projection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 12, pp. 1600–1613, 2006.

[204] R. R. De Bie T., Cristianini N., *Handbook of Computational Geometry for Pattern Recognition, Computer Vision, Neurocomputing and Robotics*. New York, NY: Springer-Verlag, 2005, ch. Eigenproblems in Pattern Recognition.

[205] J. Kettenring, "Canonical analysis of several sets of variables," *Biometrika*, vol. 58, no. 3, pp. 433–451, 1971.

[206] A. Gifi, *Nonlinear multivariate analysis*, ser. Wiley Series in Probability & Statistics. New York, NY: John Wiley & Sons, 1990.

[207] D. Witten and R. Tibshirani, "Extensions of sparse canonical correlation analysis with applications to genomic data," *Statistical Applications in Genetics and Molecular Biology*, vol. 8, no. 1, p. 28, 2009.

[208] D. L. Donoho, "For most large underdetermined systems of linear equations, the minimal 11-norm near-solution approximates the sparsest near-solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 7, pp. 907–934, 2006.

[209] A. Wiesel, M. Kliger, and A. Hero, "A greedy approach to sparse canonical correlation analysis (in preparation)," 2008.

[210] D. Hardoon and J. Shawe-Taylor, "Sparse canonical correlation analysis," in *Sparsity and Inverse Problems in Statistical Theory and Econometrics Workshop*, 2008.

[211] G. Fyfe and G. Leen, "Two methods for sparsifying probabilistic canonical correlation analysis," in *Neural Information Processing*, 2006, pp. 361–370.

[212] E. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for ℓ_1-minimization: Methodology and convergence," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008.

[213] J. Liu, S. Ji, and J. Ye, *SLEP: Sparse Learning with Efficient Projections*, Arizona State University, Tempe, AZ, 2009.

[214] J. Friedman, T. Hastie, H. Höfling, and R. Tibshirani, "Pathwise coordinate optimization," *The Annals of Applied Statistics*, no. 2, pp. 302–332, 2007.

[215] C. Sigg, B. Fischer, B. Ommer, V. Roth, and J. Buhmann, "Nonnegative cca for audiovisual source separation," in *IEEE Workshop on Machine Learning for Signal Processing*, 2007, pp. 253–258.

[216] Z. Ghahramani, T. Griffiths, and P. Sollich, *Bayesian nonparametric latent feature models*. Oxford: Oxford University Press, 2007, pp. 201–225.

[217] B. Parlett, *The symmetric eigenvalue problem*, ser. Classics in applied mathematics. Philadelphia, PA: SIAM, 1998, vol. 2nd.

[218] D. Zhou, J. Huang, and B. Schölkopf, "Learning with hypergraphs: Clustering, classification, and embedding," in *Advances in Neural Information Processing Systems 19*, 2007, pp. 1601–1608.

[219] J. Ye, "Least squares linear discriminant analysis," in *ICMLProceedings of the 24th International Conference on Machine Learning (ICML)*, 2007, pp. 1087–1094.

[220] D. Cai, X. He, and J. Han, "SRDA: An efficient algorithm for large-scale discriminant analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 1–12, 2008.

[221] C. C. Paige and M. A. Saunders, "LSQR: An algorithm for sparse linear equations and sparse least squares," *ACM Transactions on Mathematical Software*, vol. 8, no. 1, pp. 43–71, 1982.

[222] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. Cambridge, MA: The MIT Press, 2006.

[223] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[224] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, 2001, pp. 849–856.

[225] F. R. K. Chung, *Expanding graphs*, ser. DIMACS Ser. Discrete Math. Theoret. Comput. Sci. Providence, RI: Amer. Math. Soc., 1993, vol. 10, ch. The Laplacian of a hypergraph, pp. 21–36.

[226] S. Rosenberg, *The Laplacian on a Riemannian Manifold: An Introduction to Analysis on Manifolds*. Cambridge, England: Cambridge University Press, 1997.

[227] J. Zien, M. Schlag, and P. Chan, "Multilevel spectral hypergraph partitioning with arbitrary vertex sizes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 9, pp. 1389–1399, 1999.

[228] M. Bolla, "Spectra, Euclidean representations and clusterings of hypergraphs," *Discrete Mathematics*, vol. 117, no. 1-3, pp. 19–39, 1993.

[229] J. Rodriguez, "On the Laplacian eigenvalues and metric parameters of hypergraphs," *Linear and Multilinear Algebra*, vol. 50, no. 1, pp. 1–14, 2002.

[230] ——, "On the Laplacian spectrum and walk-regular hypergraphs," *Linear and Multilinear Algebra*, vol. 51, no. 3, pp. 285–297, 2003.

[231] D. Zhou, J. Huang, and B. Schölkopf, "Beyond pairwise classification and clustering using hypergraphs," Max Plank Institute for Biological Cybernetics, Tubingen, Germany, Technical Report 143, 2005.

[232] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems 13*, 2001, pp. 585–591.

[233] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *Annals of Statistics*, vol. 32, p. 407, 2004.

[234] Y. Saad, *Numerical Methods for Large Eigenvalue Problems*. New York, NY: Halsted Press, 1992.

[235] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, and V. Kumar, "Idr/qr: an incremental dimension reduction algorithm via qr decomposition," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1208–1222, 2005.

[236] T.-K. Kim, S.-F. Wong, B. Stenger, J. Kittler, and R. Cipolla, "Incremental linear discriminant analysis using sufficient spanning set approximations," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8.

[237] H. Zhao and P. C. Yuen, "Incremental linear discriminant analysis for face recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 1, pp. 210–221, 2008.

[238] E. James and S. Annadurai, "Implementation of incremental linear discriminant analysis using singular value decomposition for face recognition," in *The First International Conference on Advanced Computing (ICAC)*, 2009, pp. 172–175.

[239] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions." *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[240] T. N. E. Greville, "Some applications of the pseudoinverse of a matrix," *SIAM Review*, vol. 2, no. 1, pp. 15–22, 1960.

[241] L.-P. Liu, Y. Jiang, and Z.-H. Zhou, "Least square incremental linear discriminant analysis," in *The 9th IEEE International Conference on Data Mining (ICDM)*, 2009, pp. 298–306.

[242] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[243] A. Asuncion and D. Newman, "UCI machine learning repository," 2007.

[244] K. Lang, "Newsweeder: Learning to filter netnews," in *Proceedings of the 12th International Conference on Machine Learning (ICML)*, 1995, pp. 331–339.

[245] J. Carson and *et al.*, "A digital atlas to characterize the mouse brain transcriptome," *PLoS Computational Biology*, vol. 1, no. 4, p. e41, 2005.

[246] E. S. Lein and *et al.*, "Genome-wide atlas of gene expression in the adult mouse brain," *Nature*, vol. 445, pp. 168–176, 2006.

[247] A. A. Samsonova, M. Niranjan, S. Russell, and A. Brazma, "Prediction of gene expression in embryonic structures of *drosophila melanogaster*," *PLoS Computational Biology*, vol. 3, no. 7, pp. 1360–1372, 2007.

[248] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, no. 1-2, pp. 43–72, 2005.

[249] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.

[250] J. Zhou and H. Peng, "Automatic recognition and annotation of gene expression patterns of fly embryos," *Bioinformatics*, vol. 23, no. 5, pp. 589–596, 2007.

[251] S. Schölkopf, K. Tsuda, and J.-P. Vert, *Kernel Methods in Computational Biology*. Cambridge, MA: MIT Press, 2004.

[252] R. Kondor and T. Jebara, "A kernel between sets of vectors," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 2003.

[253] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features," in *The 10th IEEE International Conference on Computer Vision (ICCV)*, 2005, pp. 1458–1465.

[254] ——, "Approximate correspondences in high dimensions," in *Advances in Neural Information Processing Systems 19*, 2007, pp. 505–512.

[255] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *International Journal of Computer Vision*, vol. 73, no. 2, pp. 213–238, 2007.

[256] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine Learning Research*, vol. 5, pp. 27–72, 2004.

[257] A. Zien and C. S. Ong, "Multiclass multiple kernel learning," in *Proceedings of the 24th international conference on Machine learning (ICML)*, 2007, pp. 1191–1198.

[258] G. R. G. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble, "A statistical framework for genomic data fusion," *Bioinformatics*, vol. 20, no. 16, pp. 2626–2635, 2004.

[259] T. De Bie, L.-C. Tranchevent, L. M. M. van Oeffelen, and Y. Moreau, "Kernel-based data fusion for gene prioritization," *Bioinformatics*, vol. 23, no. 13, pp. i125–132, 2007.

[260] K. Grauman and T. Darrell, "The pyramid match kernel: Efficient learning with sets of features," *Journal of Machine Learning Research*, vol. 8, pp. 725–760, 2007.

[261] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006, pp. 2169–2178.

[262] L. Fei-Fei and P. Perona, "A Bayesian hierarchical model for learning natural scene categories," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 524–531.

[263] R. Hettich and K. O. Kortanek, "Semi-infinite programming: Theory, methods, and applications," *SIAM Review*, vol. 35, no. 3, pp. 380–429, 1993.

[264] M. Varma and A. Zisserman, "Texture classification: Are filter banks necessary?" in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003, pp. 691–698.

[265] R. Datta, D. Joshi, J. Li, and J. Z. Wang, "Image retrieval: Ideas, influences, and trends of the new age," *ACM Computing Surveys*, vol. 40, pp. 5:1–5:60, 2008.

[266] C. C. Paige and M. A. Saunders, "Algorithm 583: LSQR: Sparse linear equations and least squares problems," *ACM Transactions on Mathematical Software*, vol. 8, no. 2, pp. 195–209, 1982.

[267] R. Rifkin, G. Yeo, and T. Poggio, "Regularized least-squares classification," in *Advances in Learning Theory: Methods, Model and Applications, NATO Science Series III: Computer and Systems Sciences*. Amsterdam: VIOS Press, 2003, pp. 131–154.

[268] Z. Zhang, G. Dai, and M. Jordan, "A flexible and efficient algorithm for regularized Fisher discriminant analysis," in *Proceedings of the European Conference on Machine Learn-*

ing and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD), 2009, pp. 632–647.

[269] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective and efficient multilabel classification in domains with large number of labels," in *Proceedings of ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD)*, 2008.

[270] C. G. M. Snoek, M. Worring, J. C. van Gemert, J.-M. Geusebroek, and A. W. M. Smeulders, "The challenge problem for automated detection of 101 semantic concepts in multimedia," in *Proceedings of the 14th annual ACM international conference on Multimedia (MULTIMEDIA:)*, 2006, pp. 421–430.

[271] E. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[272] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[273] Y. Eldar, P. Kuppinger, and H. Bolcskei, "Block-sparse signals: Uncertainty relations and efficient recovery," *IEEE Transactions on Signal Processing*, vol. 58, no. 6, pp. 3042–3054, 2010.

[274] S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010, pp. 543–550.

[275] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach, "Proximal methods for sparse hierarchical dictionary learning," in *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010, pp. 487–494.

[276] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397 –3415, 1993.