Mining high-dimensional and graph data using spectral analysis

Li, Wenyan

2007

Li, W. Y. (2007). Mining high-dimensional and graph data using spectral analysis. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/2360>

<https://doi.org/10.32657/10356/2360>

Nanyang Technological University

Downloaded on 26 Apr 2023 20:17:27 SGT

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

MINING HIGH-DIMENSIONAL AND GRAPH
DATA USING SPECTRAL ANALYSIS

**MINING HIGH-DIMENSIONAL AND GRAPH DATA
USING SPECTRAL ANALYSIS**

LI WENYUAN

LI WENYUAN

SCHOOL OF COMPUTER ENGINEERING

2007

2007

Mining High-Dimensional and Graph Data Using Spectral Analysis

Li Wenyuan

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2007

Acknowledgments

Looking back the three years' study in Nanyang Technological University, I amazingly found that I have been quite different from what I was three years ago. The experiences I learned in this period are almost more than those of the past twenty five years. The most valuable is the ideas, advice, help, encouragement and financial support warmly given by a group of wonderful people.

My first thanks go to my advisor, Dr. Ng Wee-Keong. It is him that brought me to the world of research that is full of colorful dream, intriguing exploration, and exciting challenges. Actually I am quite familiar with this world when I studied mathematics and solved difficulty problems in high school. However, such feeling was suddenly cut off during my undergraduate and postgraduate. Therefore, I particularly appreciate what he instructed me. He demonstrated to me that the research can be made in this way. His knowledge and patience are worth learning. Moreover, he is an understanding advisor that is very considerate of my difficulties, and often pull me back on track when I made mistakes.

Dr. Lim Ee-Peng, the head of the information system division, deserves the second thanks. Although he is very busy, he still allocated the time to share his experiences and ideas with me when I discussed with him. Besides, in the last several months of my study, his text mining group provides a warm environment and forum to me for sharing experiences and ideas with each other, and for practicing my presentation techniques. He always provides me enthusiastic help in need. His proactive attitude towards research also deserves my admiration.

In addition, I like to thank Dr. Ong Kok-Leong, my senior, friend and more my mentor. During the whole period of my pursuing Ph.D. degree, Kok-Leong is like my elder brother to give me much help and cares. As my senior, he is more like my mentor. His experiences are my lessons. He taught me much knowledge inside and beyond research. Moreover, I appreciate his great help of polishing my English

writing. In addition, he helped me to gain the opportunities of exposure in conferences by partially funding me.

Special gratitude also goes to my four seniors, David, Aixin, Zehua and Li Zhao. They offered me many helps whatever in research or in everyday life. I would also like to thank all the labmates in CAIS lab. From them, I had the opportunities to share experiences in research. It is them that built a fun environment to work in.

Last but not least, my deepest gratitude to my wife, Zhenghong, my love. She loves, supports me with her whole life. It is her that lights my way in the rest of my life. Without her, I would not survive in this world.

Contents

Acknowledgments	i
List of Figures	vi
List of Tables	ix
List of Algorithms	xi
Abstract	xii
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	4
1.3 Contributions	6
1.4 Organization	7
2 Literature Survey	9
2.1 Basics of Matrix	10
2.1.1 Matrix Representations	10
2.1.2 Matrix Metrics	11
2.2 High-Dimensional Data Analysis	12
2.2.1 Clustering High-Dimensional Data	13
2.2.2 Classification of High-Dimensional Data	16
2.2.3 Dimensionality Reduction and Feature Selection	17
2.2.4 Visualization of High-Dimensional Data	20
2.3 Graph Data Analysis	20
2.3.1 Basics of Graphs	21
2.3.2 Graph-based Data Mining	22
2.3.3 Characteristics and Mechanisms of Complex Networks	25
2.4 Data Analysis by Spectral Techniques	27

2.4.1	Historical Background	27
2.4.2	Preliminaries	28
2.4.3	Intuitions of Spectral Components Underlying its Applications .	29
2.4.4	Applications of Spectral Analysis in Various Domains	31
3	Enhancing the Effectiveness of Clustering with Spectrum Analysis	34
3.1	Preliminaries	36
3.2	Spectrum Analysis by 1-Normalization	38
3.2.1	A Simple Case	40
3.2.2	1-Spectrum Distribution in Large Data sets	42
3.2.3	Empirical Results	44
3.3	p -normalization ($p > 1$) Spectrum Analysis	47
3.3.1	Differential Levels Between Clusters	47
3.4	A Motivating Example	52
3.5	Automatic Detection of k	53
3.6	Additional Empirical Results	54
3.6.1	Text Data	55
3.6.2	Gene Expression Data	57
3.6.3	Spatial Data	63
3.6.4	Performance	65
3.7	Discussion	68
3.8	Related Works	70
3.9	Summary	71
4	Spectral Kernels for Classification	73
4.1	Spectral Graph Analysis of Kernel Matrices	74
4.1.1	Graph Cut Criteria of Kernel Matrices	75
4.1.2	Computing the Spectral Embedding Space	77
4.2	Spectral Kernels	79
4.2.1	Transforming and Spectral Embedding of New Input	80
4.2.2	Computing the Spectral Kernel	81
4.2.3	Relationship to Latent Semantic Kernel	81
4.3	Experimental Results	82

4.3.1	Results on Medline1033	83
4.3.2	Results on Reuters21578	85
4.3.3	Computational Complexity	85
4.4	Summary	86
5	Visual Terrain Analysis of High-dimensional Data by Resonance Model	88
5.1	Problem Formulation of Generic and Basic Terrains	91
5.1.1	Generic Terrain of Data	91
5.1.2	Basic Terrain of Data: Distinct ‘Mountain’	91
5.2	Discovering Basic Terrain of Data	95
5.2.1	Resonance Model	96
5.2.2	Properties of the Model	98
5.2.3	Relationships between Basic Terrain and Typical Patterns . . .	100
5.2.4	Two Real-World Examples for Visually Analyzing Data With Basic Terrain	104
5.3	‘Mountain-Valley’ Terrain and Application to Gene Selection	107
5.3.1	Background and Related Work	107
5.3.2	‘Mountain-Valley’ Terrain of Data	109
5.3.3	Extended Resonance Model	110
5.3.4	Empirical Study	113
5.4	Summary	117
6	Conclusions	118
6.1	Future Works	119
6.2	List of Publications	120
	References	123

List of Figures

3.1	A comparison of two similarity matrices before and after the 3-normalization: (a) $\mathbf{S}_1$ with 2 disjoint clusters – cluster 1 has 30 instances and average similarity of 0.5 and cluster 2 with 70 instances with average similarity of 0.3; (b) $\mathbf{S}_2$ with the same disjoint clusters as $\mathbf{S}_1$ but both clusters have an average similarity of 1; (c) normalized $\mathbf{S}_1$; (d) normalized $\mathbf{S}_2$	37
3.2	The spectrum distribution of the collection in Table 3.1: (a) Spectrum (> 0) of $\mathcal{G}(\mathbf{S})$ and its subgraphs; (b) a graphical representation of $\mathbf{S}$. Note that all grey images in this chapter are not “plots” of the spectrum – They are a graphical way of summarizing the results of clustering for analysis purposes.	41
3.3	The spectrum graphs and the graphical representation of their clustering for all 3 collections: the first two data sets are conceptually disjoint, and the last is conceptually overlapping.	45
3.4	A text collection of 275 new posters selected from newsgroup with four topics that overlap at high-level concepts: (a) gray scale image of similarity matrix $\mathbf{S}$; (b) the topic of each cluster.	48
3.5	The color images of $L_p(\mathbf{S})$ and the α sequences (see Theorem 3.4, $\alpha_i = \frac{\lambda_{i+1}}{\lambda_2}$) induced from their p -spectrum respectively.	49
3.6	The color Images of $L_p(\mathbf{S})$ and the α sequences (refer to Theorem 3.4, $\alpha_i = \frac{\lambda_{i+1}}{\lambda_2}$) induced from their p -spectrum, respectively. The left depicts the color image and α sequences of $L_1(\mathbf{S})$, and the right depicts the color image and α sequences of $L_\infty(\mathbf{S})$: (a) tr12 ; (b) re0 ; (c) hitech ; (d) mm	54
3.7	The Leukemia data set analyzed with the Pearson correlation coefficient: its α sequence of 1-spectrum and ∞ -spectrum.	58

3.8	The budding yeast gene expression data set: (a) α sequence of 1-spectrum; (b) α sequence of ∞ -spectrum; (c) the grey scale plot where (1) is protein degradation, (2) is chromatin structure, (3) is protein synthesis and (4) is glycolysis.	58
3.9	The serum gene expression data set analyzed using different similarity measures: (a) Pearson correlation coefficient; (b) extension of Euclidean distance with $\sigma = 3$	59
3.10	The NCI60 data set: (a) the grey scale plot at http://genome-www.stanford.edu/nci60/figures.shtml ; (b) α sequence of 1-spectrum; and (c) α sequence of ∞ -spectrum.	60
3.11	The monocytes cDNA data set analyzed with different similarity measures. For easy verification, two similarity matrices are reordered by the cluster labels provided in [HSL ⁺ 00] and with the increasing cluster sizes: (a) Pearson correlation coefficient resulting in 8 distinct α sequences in the 1-spectrum and ∞ -spectrum respectively; (b) Processing techniques described in [HSL ⁺ 00] followed resulting in 9 distinct α sequences in the 1-spectrum and 10 distinct α sequences in the ∞ -spectrum.	62
3.12	Simulated two-dimensional data set with 1,500 points. Both (b) and (c) revealed that there are $4 + 1 = 5$ distinct clusters in this data set. .	64
3.13	The data set DS4 with 8,000 points. From (b) and (c), there are clearly $2 + 1 = 3$ obvious clusters; there are $7 + 1 = 8$ clusters that are not obvious and affected by noises, although the 5 small blue α are not distinct enough.	64
3.14	Running time comparison of our method with three typical methods in the very large text collection RCV1-v2.	67
4.1	Spectral embedding in the spectral kernel. Let the projected k -dimensional data space be $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n \in \mathbb{R}^{k \times 1}$, and the first k interested eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ are positive. <i>Note:</i> $\mathbf{v}_j^i$ denotes the i -th coordinate of the eigenvector $\mathbf{v}_j$. $\mathbf{\Lambda}_k$ is the truncated diagonal of $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, its last $(n - k)$ diagonal entries are set 0.	79

4.2	Generalization performance of the SVM classifier with the proposed spectral kernel $SK\langle\Gamma_N, 1(a), S_I\rangle$ and the linear kernel for the Medline1033 data set.	83
4.3	Generalization performance of the SVM classifier with the proposed spectral kernel $SK\langle\Gamma_N, 1(b)2, S_I\rangle$ for the classes of Reuters21578 data set: corn, acq, money-fx, grain, crude, trade, interest, ship, wheat.	84
5.1	The rating matrix for 4 reviewers and 5 movies	90
5.2	A small graph of 40 proteins' interaction network is selected from the core <i>Saccharomyces cerevisiae</i> in Database of Interacting Proteins (DIP, available at http://dip.doe-mbi.ucla.edu). The black points in figures represent non-zeros (or one) in the adjacency matrix, as DIP data is the unweighted graph.	93
5.3	Architecture of Resonance model	98
5.4	Comparison of genes in G_k and biclusters B_k ($1 \leq k \leq 100$) for gene expression data	104
5.5	Visualization of Wisconsin breast cancer data and comparison with outlier methods	105
5.6	Visualization of MovieLens data and a local <i>basic terrain</i> in detail . . .	106
5.7	Statistics of overall <i>basic terrain</i> of MovieLens data	106
5.8	Transformation of the matrix W : the transformed matrix W' has the same structure of submatrix blocks as shown in (a), but with different submatrix $W_i'^{-}$ and $W_i'^{+}$ as listed in (b).	110
5.9	Architecture of extended resonance model	112
5.10	Visualization of the sorted matrix W , sorted transformation matrix $W' = [1 - W_{NC}, 1 - W_{FDR}, W_{ILE}, W_{SLE}]$, and sorted approximation matrix $\mathbf{r}^* \tilde{\mathbf{o}}^{*T} \approx W'$, where $\tilde{\mathbf{o}}^*$ is the concatenation of k vectors: $\tilde{\mathbf{o}}^* = (\tilde{\mathbf{o}}_1^*; \dots; \tilde{\mathbf{o}}_k^*)$	116

List of Tables

3.1	A small text collection taken and modified from [DDL ⁺ 90] for our example. It contains the titles of 12 technical memoranda: 5 about human-computer interaction; 4 about mathematical graph theory; and 3 about clustering. The topics are conceptually disjoint with two assumptions: (i) the italicized terms are the selected feature set; and (ii) the cosine similarity measure is used to compute S	40
3.2	The <i>text</i> collections used in our experiments to estimate k : we selected 4 classes of classic with each class containing 1,000 documents; 5 newsgroups with each newsgroup containing 500 documents; 2 categories of the webset with each category containing 600 documents. . .	44
3.3	Comparison of our approach against 3 well-known indexes: the Calinski and Harabasz (CH) index; Krzanowski and Lai (KL) index; and Hartigan (Hart) index with 3 well-known clustering algorithms: bisecting k -means, graph-based, and hierarchical – a ($\checkmark$) indicates a correct estimation.	46
3.4	The text collections, gene expression and spatial data sets used to compute the p -spectrum.	55
3.5	The same experiment conducted in Table 3.3 on additional data sets, indexes and algorithms. The additional indexes are Davies-Bouldin (DB) index [Gor99], SD index [HBV01] and S_Dbw index [HBV01], and the additional algorithm is gap statistic (GapS) [TWH00]. The subscripts of each data set indicates the possible class numbers, e.g., mm can have 2, 4 or 5 clusters. And the number underlined indicates a correct estimation.	55

3.6	The same experiment conducted in Table 3.3 and Table 3.5 on additional data sets, indexes and algorithms. The additional indexes are the Dunn index and PE index (based on membership matrix of objects computed by Fuzzy k-means clustering algorithm), and the additional algorithms are CLICK, MEC, and MCLUST with k -means (km) and hierarchical (hi). In CLICK, the number in the bracket represents the unclustered singletons. And the number underlined indicates a correct estimation.	63
3.7	The overall procedure of cluster number estimation	70
4.1	Solution to two graph cut criteria used in spectral clustering: (i) the corresponding transformation by eigen-decomposition of the matrix $\Gamma(\mathbf{S})$; (ii) on an incoming data $\mathbf{x}$, the similarity with the training examples is computed as $\mathbf{S}_x$ and its corresponding transformation is $\tau(\mathbf{S}_x)$. <i>Note:</i> d_x is the sum of all elements in $\mathbf{x}$; the original version of the normalized Laplacian matrix should be $\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}$; see Section 4.1.2 and Lemma 4.1.	76
4.2	The similarity measures $s(\mathbf{x}, \mathbf{y})$ used in the computation of spectral kernels.	81
4.3	F_1 using spectral kernel $\text{SK}(\Gamma_N, 1(b)2, S_I)$ and linear kernel on the ten Reuters categories with the best F1 results in bold.	86
5.1	Comparison of frequent itemsets found in D and D' with the same minimum support for the bms-pos data set. l is the length of an itemset; $\#FI_D^l$ is the number of frequent itemsets with length l in D ; $\#FI_D$ is the total number of frequent itemsets, i.e., including all lengths in D . .	101
5.2	LOOCV accuracy rate (%) of the experiment: ALL versus MLL&AML.	114
5.3	LOOCV accuracy rate (%) of the experiment: MLL versus ALL&AML.	114
5.4	LOOCV accuracy rate (%) of the experiment: AML versus ALL&MLL.	115

List of Algorithms

1	(DkA): Detects the number of distinct large eigenvalues	53
2	(ERM): Gene Selection	113

Abstract

Although the field of data mining has seen major advancements in the past fifteen years, algorithms for handling complex data (with high dimensionality or complex graph structures) are only becoming the mainstream in recent years. To address the difficulties of mining complex data, we argue that a right understanding of data characteristics (i.e., the general information of the data that is not particularly designed for any specific data mining task, but might enhance many types of data mining tasks) is important. The objective of this thesis is to study and exploit spectral information to provide quick insights into how data characteristics are beneficial to specific applications. We study issues concerning the design of how spectral information can be integrated into the needs of different types of analysis. We propose (1) a novel method called the **spectrascopy** to determine the number of clusters in very large and high-dimensional data sets for automating the clustering process; (2) a novel type of kernels called **spectral kernels**, which incorporate the latent semantics extracted from spectral clustering into new kernels so that semantic information may enhance data classification; and (3) a novel concept called data **terrain**, a resonance model to facilitate its discovery and its relationships with typical patterns (i.e. frequent item-sets and biclusters). Experiments on three types of analysis yield promising results. **Spectrascopy** improves the automation of clustering process and thus narrows the gap between the user and the available machinery. **Spectral kernels** achieve improvements in classification performance. The data **terrain** has been demonstrated to be useful in visualizing and analyzing complex data sets.

Chapter 1

Introduction

Rapid advances in information and communication technologies in the last thirty years have revolutionized the world. Tremendous amounts of data are generated every day in all facets of human activities. Unlike data in the past, the data sets we collect today are often high in dimensionality (e.g., tens of thousands of dimensions) and possess complex graph structures (e.g., power law distribution of vertex degrees).

In attempting to analyze these data sets, analysts experience the limitations of existing tools. In the context of data mining, algorithms capable of handling such data have only become available in recent years. Indeed, this is not surprising since it has been proven by results in the fields of database systems [HK01] and statistics [Don00] that existing analytical techniques do not work well on such data sets. In addition to the difficulties of analyzing complex data, many of these proposed algorithms require parameters and expertise to operate.

This thesis addresses the above problem through a number of novel algorithms based on spectral techniques. By exploiting the spectral characteristics of the data in question, we observe that it is possible for our algorithms to derive useful insights that will guide an analyst in his/her quest for knowledge from such data.

1.1 Motivation

Nowadays, most data (e.g., scientific and business databases, Web data, autonomous social networks, etc.) carry the hallmark of high dimensionality or graph-based structures. Arguably, such data sets are more complicated than conventional data we see in the early era of data mining.

CHAPTER 1. INTRODUCTION

Possibly, the seriousness of the problem of high dimensionality is commonly described by the phrase “the curse of dimensionality”, which has been coined to reflect the difficulties of analyzing such data sets. To give readers an idea of its impact, Beyer *et al.* [BGRS99] showed that the concept of proximity (or nearest neighbors) may not be meaningful in data sets with high dimensionality; and Donoho [Don00] identified several areas where the curse classically appears.

Besides the dimensionality issue, capturing the characteristics of the data today often requires a graph-based model. Recent research has revealed interesting and surprising theoretical results to reflect the complexity of today’s real-world data. For example, the term “small world” actually embodies a popular law of how entities of this world are connected to one another [Mil67, TM69]. It appears in many ways, including social networks [KSS97, NSP03, FS01, LNO05], Web graphs [Ada99], food and metabolic networks [MS02, WF00], electronic circuits, and software architectures [FJS01, VS03]. Interestingly, recent research [dFCRTB05] discovered that the “small world” phenomenon represents only the tip of the knowledge we have about the complexity of real-world graphs.

While these complex data sets potentially hold more insights than ever, it also brings about greater challenges to data analysts. One of these challenges is the combinatorial explosion problem when searching the high dimensional space (or a large graph). Many irrelevant patterns may be obtained in the searching process. Consequently, one way to manage this large volume of results is to set parameters within the analytical algorithms that require analysts to ‘tune’ the parameters until the desired outcome is obtained. In doing so, the analyst engages in a trial-and-error process that can be very time consuming for these complex data sets. A good example to support our case is the estimation of the possible cluster number in data sets during clustering. More often than not, this is an iterative process where the number of clusters desired is iteratively tuned until the desired clusters are obtained.

Motivated by the above, we argue that it would be a good strategy in practice to consider the characteristics of the data and make the best use of such knowledge in data mining algorithms. Such a strategy, which is domain independent to a certain extent, helps avoid a blind search of patterns as well as clueless optimization of the objective measure. Furthermore, if the strategy can be executed quickly and thus,

CHAPTER 1. INTRODUCTION

saving the number of trials, the discovered insight would bear more value to what might be discovered within the window of opportunity.

An example of how a good strategy affects analysis can be seen in the work of Wang *et al.* [WHP03], where depending on the sparseness of the data, different closed frequent pattern mining strategies are used. Another example is data skewness, a phenomenon that is prevalent in most real-world data sets. In some sense, skewness suggests that the insights about the data set have been effectively reflected or concentrated in a small portion of the data itself. Therefore, any algorithm that exploits this fact would effectively and efficiently find most of the insights by simply analyzing a very small portion of the given data set. Based on this observation, we hypothesize the existence of other data characteristics that can be exploited to improve the accuracy or runtime performance of existing data mining algorithms.

To the best of our knowledge, we observe that there is not much work in this direction. We believe this is where the research conducted in this thesis fills the gap. We studied the characteristics of modern data sets using spectral information ¹. As with existing research that exploited similar observations, their work have shown that spectral information is effective in the understanding of high dimensional and graph-based data. Latent semantic analysis for text [LFL98] and the PageRank algorithm for Web graphs [PBMW98] are good examples. In the spectral analysis of data, we view data in the form of a matrix and obtain meta information through efficient matrix computation techniques. Interestingly, a matrix can be represented as a graph ², and thus graph analysis is closely related to matrix theory. Consequently, the graph representation of a matrix has the potential to yield intuitive observations, results and explanation of matrix computation with the graph concept (refer to spectral graph theory [Chu97]). Best of all, matrix computation techniques also apply to the analysis of graphs ³.

If we take the above view of spectral methods as numerical methods and matrix computation techniques, we see that spectral methods have actually been used in

¹The spectral information obtained from the data can be viewed as meta information and therefore, is not tied to any analytical techniques. In this thesis, we view meta information as the insights for accelerating and improving analysis in data mining.

²A symmetric matrix can be represented by a weighted graph, and a general matrix can be represented as a weighted bipartite graph.

³It may appear to some readers that our work focuses on the analysis of high dimensional data sets. However, if we view that as a kind of graph data analysis by virtue of their interchangeable relationship above, then the motivation of our thesis becomes obvious.

information retrieval, Web pages ranking, collaborative filtering, and so on. Many experimental evidences have shown the usefulness of spectral information with respect to the understanding of data (with high dimensionality or graph structures), and how “meaningful” the patterns are to the structure of data. In addition, numerical methods for spectral computation have been very mature and well-understood. Hence, existing data mining algorithms will benefit from spectral methods in their analytical tasks.

1.2 Objectives

The objective of this thesis is to discover spectral techniques that efficiently and effectively reveal the characteristics of a given data set and to study how those characteristics can be used to assist data mining tasks. We start with a survey of existing literature on high-dimensional and graph data analysis and we select three representative problems where we studied them in detail. Our approach is to use spectral techniques to reveal the characteristics of data sets to users as a guide to the next step of data mining; e.g., clustering, classification, and visualization. Our approach is more convenient, flexible, and interactive than existing data mining methods. It is also easy to be integrated into existing data mining systems. In so doing, we aim to provide insights that will lay the groundwork for further research in this direction. We discuss the three issues and their applications below.

Enhancing Cluster Analysis Clustering has been a topic of extensive research and study in the domain of statistics, pattern recognition, machine learning and data mining. In data mining, one of the research challenges is to cluster large data sets with minimal input parameters while maintaining scalable runtime performance [HK01]. Unfortunately, the state-of-the-art research for clustering focused mainly on performance of large and complex data sets. These algorithms often require input parameters that may influence the quality of analysis and thus, leave users in the time-consuming cycle of trials (and errors) with the parameters. Sometimes, a poor parameter setting may increase the runtime dramatically. What is worse, large complex data set with high dimensionality translates to more difficulties for the analyst who faces the data set for the first time. Therefore, how to enhance cluster analysis in terms of narrowing the gap

between the user and the available machinery is an important and crucial factor to the success of knowledge discovery.

Enhancing Classification In contrast to clustering, which is unsupervised learning, classification is supervised learning where the first step is to build a model using training samples; and in the second step, the model obtained by learning the training samples is used for the test (or previously unseen) data. Especially for text data, an important issue in classification is the ability to incorporate more semantic information into the model for better prediction accuracy. For example, N -grams and phrases extracted from free text, tag features and link information from Web pages are considered as additional useful semantics. However, there are two challenges involved: (i) how do we encode the semantic information to improve prediction accuracy; and (ii) how can we accelerate the task of obtaining the domain knowledge and integrate it into the model? The first challenge can be addressed by a kernel-based learning technique that separates the first step of learning into two sub-steps. In the first sub-step, we map the feature space to the kernel space and train a classifier model in the kernel space [STC04]. Using kernels, we show that it is easier to encode semantic information. In doing so, the second problem can be partially solved by obtaining the knowledge and semantic information from automated discovery algorithms.

Visual Analysis of High-Dimensional Data To visualize high-dimensional data, the first step of most visualization methods is to employ dimensionality reduction techniques to project the data into two or three dimensions. Then the proximity of objects (or features) in data is visualized in this 2D or 3D space where analysts may identify the relationships of objects (or features) by observing their distances. However, only distances of either objects or features in 2D or 3D space often fail to reflect the knowledge of how the data (including both objects, features and their corresponding values) are distributed and how discovered patterns (e.g., clusters, differentially expressed genes in gene expression data) are constructed. Such knowledge facilitates analysts to more deeply understand the high-dimensional data and to analyze the discovered patterns. Therefore, a candidate solution is to provide users an interactive visualization system to mine the high-dimensional data. With a visual overview of data distribution, users may interactively find relevant patterns that interest them.

1.3 Contributions

In a broader sense, spectral technique is a cross discipline of various areas including matrix computation, complex systems and discrete dynamical systems, optimization methods, and graph theories and algorithms. As a result, some methods used in the thesis are variants of conventional spectral techniques (i.e., eigen decomposition). For example, the class of resonance models proposed in Chapter 5 is a form of discrete dynamical system originating from the matrix power method for obtaining eigen decomposition in numerical computations [GL96]. Therefore, all solutions provided in this thesis can be viewed as the spectral analysis of data.

In meeting each of the objectives identified above, this thesis proposes spectral analysis to enhance high-dimensional and graph data mining. They are summarized as follows:

- We propose a novel method called the **spectrascopy** to determine the number of clusters in a very large and high-dimensional data set for automating the clustering process. Our proposal is to perform spectra analysis on the similarity space of the data set by analyzing the eigenvalues (*not* eigenvectors) that encode the answer to our question. As our method does not require the user to specify a range of k to test with a clustering algorithm, the analysis is highly efficient. Recognizing the subjectivity of what constitutes the right value of k , we extend our work to estimate a range of k of a given data set. As the outcome of clustering largely depends on the similarity measure used and the analysis context, computing a rigid value of k may be inappropriate in some situations. We show that our method can be easily extended to handle this real-life issue by further spectra analysis. The proposed method is realized in the context of a system that automates this preprocessing step.
- We propose a novel kernel called **spectral kernels**. The purpose of **spectral kernels** is to incorporate latent semantics extracted from spectral clustering into the new kernels such that the semantic information enhances data classification. One of the first work to exploit this is Latent Semantic Kernels (LSKs) [CSTL01], which we have proved to be a special case of **spectral kernels**. In the other work that uses spectral learning [KKM03], we argue

that it may not be feasible in practice and has not been compared to support vector machines [CST00]. Unlike these previous works, **spectral kernels** are unique not only by virtue of using spectral clustering information but also its ability to support incremental updates to the kernel matrix, thus keeping the cost of training minimal. Moreover, we propose a flexible kernel framework of applying spectral clustering information to data classification; this provides analysts options to tune the performance.

- We propose the concept of data **terrain**, which is discovered through a resonance model. We showed that the data terrain is useful in data analysis as it reveals data distribution by collecting high/low values; i.e., ‘mountain’ or ‘valley’ and thus, help discover patterns and visualize high-dimensional data. The data **terrain** provides a bird’s eye view of the high-dimensional data using a visual landscape where entries of data are ‘places’ and their values become the ‘altitudes’. We argue that if users have knowledge of how data is distributed, it becomes easier to discover interesting patterns. To support our argument, we show how biclusters⁴ and frequent itemsets are related to basic **terrains** with theoretical and empirical evidence, and we apply the ‘mountain-valley’ **terrain** to visualize gene expression data for biomarker discovery. The resonance model is inspired by the resonance phenomenon and simulates its mechanism. In mathematics, such a model is seen as a discrete dynamical system which is a variant and extension of spectral computation [San90].

1.4 Organization

This thesis is organized into the following chapters. Following this Chapter’s introduction, Chapter 2 surveys the literature in spectral techniques, their applications, and data mining and machine learning techniques related to the three representative issues we shall discuss in the thesis. In Chapter 3, we present **spectrascopy** as a preprocessing technique to capture the data characteristic that enhances automation

⁴A bicluster is a set of objects and a set of features in a submatrix that exhibit similar characteristics or trends. In other words, a bicluster is a relevant group of rows and relevant group of columns in a matrix. The term was first introduced by Cheng and Church [CC00] in gene expression analysis, although the technique was originally introduced much earlier by J.A. Hartigan [Har72].

CHAPTER 1. INTRODUCTION

in cluster analysis. For the supervised learning task, we discuss in Chapter 4 the method to encode and integrate the knowledge discovered into the kernels to improve classification. Chapter 5 discusses the last issue, where we show how we can visualize and interact with high dimensional data sets to achieve knowledge discovery. We then conclude our work in Chapter 6 with a discussion of future work.

Chapter 2

Literature Survey

It is common nowadays to find large amount of data, e.g., Web information and bioinformatics, where the data is often high-dimensional or graph-based. Since they share the same mathematical representation - matrix, high-dimensional data sets and graphs are strongly correlated in many aspects. A well-known example is the spectral technique used in the PageRank algorithm for Web graphs [PBMW98] and in the normalized cut algorithm for image segmentation [SM00]. Although there have been a series of works in applying spectral techniques to these two complex types of data, *“its full potential is still unexplored”* as pointed out by Marina Meila. In this chapter, we made an interesting attempt to reveal more potentials of spectral techniques in exploring high-dimensional data and graphs. We present our analysis of the state-of-the-art research from the following three subfields:

- **High-dimensional data analysis:** As each topic discussed in this field can be easily expanded into a chapter, the goal of our analysis in this part is to introduce significant works in the domain. While their features are highlighted, methods proposed in these works will serve as the background of introducing spectral techniques for high-dimensional data analysis.
- **Graph data analysis:** Although graph theory and algorithms have a long history, it is new in data mining where only a few works are present. Yet, there have been a lot of works from other domains in real-world network analysis. To provide a full view of graph data analysis, we shall review graph data analysis by summarizing the essential characteristics of modern graphs, their approaches, and theories from diverse disciplines.

- **Spectral technique:** Given its background from mathematics and physics, and roots in diverse disciplines, the spectral technique has a strong foundation which deserves merit as a platform of data mining research. Unfortunately, it is also the same reason that prevents spectral techniques from gaining the wide uptake within the data mining community. Rather than presenting this as a research topic or technique, our goal is to illustrate the intuition in spectral techniques by discussing them in the context of real-world applications.

2.1 Basics of Matrix

We shall start with an introduction to the matrix representation of the complex data which will lay foundation to our discussion of high-dimensional and graph-based data analysis. Given that matrix is widely used as a method of data representation, its computational techniques became the core component of most scientific tasks, e.g., numerical computation, neural networks, optimization, machine learning, etc. Especially in machine learning, the effectiveness and efficiency of many learning models are fundamentally dependent on matrix computation. A well-known example is the Support Vector Machines (SVMs) [CST00], introduced by Vapnik *et al.* and formulated as a problem of minimizing the empirical classification error and maximizing the geometric margin. It can be solved by the quadratic programming tool, in which a large number of matrix computation operations are involved [Pla99]. Another example is Google’s PageRank algorithm which computes an eigenvector of a matrix with an order of 2.7 billion – the world’s largest matrix computation ¹. Therefore, before our analysis of three subfields listed above, we present the matrix representations for most data sets, and introduce the basic matrix metrics relevant to our work.

2.1.1 Matrix Representations

Most high-dimensional data can be expressed by two elements: objects and attributes. The relationship between any object and any feature is measured by a magnitude. Let O be a set of objects, where $o \in O$ is defined by a set of attributes A . Further, let w_{ij} be the magnitude of o_i over $a_j \in A$. We represent the relationships of all objects

¹MATLAB news and notes at October 2002, available at http://www.mathworks.com/company/newsletters/news_notes/clevescorner/oct02_cleve.html

CHAPTER 2. LITERATURE SURVEY

and their attributes in a matrix $W = (w_{ij})_{m \times n}$ ($m=|O|$ and $n=|A|$), where each row represents an object and each column an attribute.

Given a graph $G(V, E)$ where V is the vertex set and E is the edge set, it can be represented by an adjacency matrix $W_{|V| \times |V|}$ of this graph, where $|V|$ is the number of vertices in G . In this adjacency matrix, the entry $w_{ij} \in \mathbb{R}$ represents the weight of the edge starting from the vertex v_i and ending at the vertex v_j . For an undirected graph, W is a symmetric matrix. For an unweighted graph, W is a binary matrix whose entries take the value of zero or one. Therefore, graphs with different properties lead to their adjacency matrices with different features.

2.1.2 Matrix Metrics

The body of knowledge in linear algebra and matrix computation theory is built on the matrix metrics. In the following, we shall introduce three widely used matrix metrics that are important to the understanding of spectral techniques: “Frobenius norm” and “Rank” of a real matrix, and “Definiteness” of a real symmetric matrix. We begin with some preliminary concepts.

Definition 2.1 (Matrix Norm) *Given a real matrix A , its matrix norm $\|A\|$ is a nonnegative number associated with A having the following properties,*

- (i) $\|A\| > 0$ when $A \neq 0$ and $\|A\| = 0$ iff $A = 0$.
- (ii) $\|kA\| = |k|\|A\|$ for any scalar k .
- (iii) $\|A + B\| \leq \|A\| + \|B\|$.
- (iv) $\|AB\| \leq \|A\|\|B\|$.

The norm of a mathematical object is a quantity that in some (possibly abstract) sense describes the length, size, or extent of the object. Therefore, the matrix norm is a measure to provide a number to quantify each matrix [GL96].

Definition 2.2 (Linearly Independent Vector Set) *Two or more vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$, which are not linearly dependent, i.e., cannot be expressed in the form*

$$\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2 + \dots + \alpha_n \mathbf{x}_n = 0$$

with constants $\alpha_1, \alpha_2, \dots, \alpha_n$ which are not all zero are said to be linearly independent.

The linear independence of a vector set depicts a linear relationship among vectors. It has a foundational geometric background and is a basic concept of understanding many matrix computation methods (including spectral techniques).

Frobenius Norm According to the general definition of the matrix norm, the Frobenius norm is a special norm with the definition $\|W\|_F \equiv \sqrt{\sum_{i=1}^m \sum_{j=1}^n w_{ij}^2}$. It is an important metric to measure the similarity of two matrices, i.e., matrix approximation.

Rank By definition of the linearly independent vector set, a maximal subset of vectors can be selected (from a set of vectors) which are linearly independent. We call the number of vectors in this subset as the rank. A matrix can be decomposed into two sets of vectors: row vector set and column vector set. Consequently, each vector set of the matrix W has its own rank, i.e., “column rank” and “row rank”.

Definiteness Suppose W is a real symmetric matrix. If, for any non-zero vector $\mathbf{x}$, we have $\mathbf{x}^T W \mathbf{x} > 0$, then W is a positive definite matrix; if $\mathbf{x}^T W \mathbf{x} \geq 0$, then W is a positive semi-definite matrix. The definiteness of a real symmetric matrix is an important property that has been used in many areas of mathematics and physics. This is because an undirected graph can be represented as a symmetric adjacency matrix, whose definiteness is essentially related to many properties of this graph. A well-known example is the Laplacian transformation of an adjacency matrix that has been proved to be positive semi-definite [GMS90]. Furthermore, the expression in the definiteness definition, $\mathbf{x}^T W \mathbf{x}$, can be rewritten as $\sum_i \sum_j w_{ij} x_i x_j$, which is a typical quadratic optimization target function in the optimization field. Therefore, the definiteness of an adjacent matrix connects the fields of graph theory and optimization in nature. This close relationship enables these two fields to complement each other.

2.2 High-Dimensional Data Analysis

Regardless, scientific experiments or business transactions often generate very high-dimensional data. For example, a text collection has tens of thousands of terms and

needs to be analyzed through text or Web mining ²; and in bioinformatics, gene expression data can have hundreds or thousands of conditions or samples ³. It is challenging for any learning model and algorithm to analyze such data with the dimensionality in the range of hundreds to hundreds of thousands. In the sections that follow, we discuss four methods of high-dimensional data analysis: clustering, classification, dimensionality reduction and feature selection, and visualization.

2.2.1 Clustering High-Dimensional Data

The curse of dimensionality has brought major challenges to the task of clustering. Many techniques used in low-dimensional data sets are not applicable to their high-dimensional counterparts. For instance, the shape of a cluster does not make as much sense in high-dimensional data sets; and “the distance to the nearest neighbor becomes indistinguishable from the distance to the majority of points” [BGRS99]. Consequently, there seems to be a need to develop new clustering algorithms, in which appropriate similarity/dissimilarity measures or new cluster patterns only appearing in high-dimensional spaces are investigated. Our discussion shall be based upon two categories of data: text and microarrays, because they are publicly available and widely used, and most new clustering algorithms fall into these two categories.

Text data: According to the vector space model of text, each document corresponds to a vector whose elements represent words and the value of each element represents the occurrence magnitude of its corresponding word in the document. Therefore, a text collection can be represented as a word-document matrix. Since only a small portion of the language words are used in each document in the context of a relevant topic, this word-document matrix is often very sparse (i.e., with many zeros). It is one of the distinct features in text data. To analyze text data, a problem naturally arises, “are traditional similarity/dissimilarity measures and clustering algorithms still effective for text?” Strehl *et al.* systematically considered the impact of traditional similarity measures on text data. They concluded that the metric distance functions such as Euclidean distance is not appropriate for high-dimensional, sparse

²In the newly text benchmark collection RCV1-v2, there are 47,236 meaningful terms. Source: Lewis, D. D. *RCV1-v2/LYRL2004: The LYRL2004 Distribution of the RCV1-v2 Text Categorization Test Collection (12-Apr-2004 Version)* at http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm

³<http://bioinformatics.upmc.edu/Help/UPITTGED.html>

CHAPTER 2. LITERATURE SURVEY

domains while the Cosine, correlation and Jaccard functions are most effective to capture the similarities in text [SGM00]. This discovery, in some sense, confirms the need to take the second look at every component of clustering in the high-dimensional space. An intuitive explanation of why Cosine-like measures work in text data is that the similarity of two documents is largely determined by the co-occurrence statistical property of the same subset of words expressing the same topic, and Cosine-like similarity measures are able to capture such co-occurrence property.

Karypis and his group have also reported results in this area recently. A series of works have been done by applying traditional clustering algorithms with different criterion functions to text data. The performances of these algorithms [SKK00, ZK02a, ZK04, ZK02b] have been compared. The empirical study [SKK00] showed that a variant of k -means clustering algorithm, bisecting k -means, works the best for text data among existing clustering algorithms. Although the reinvestigation of traditional measures and clustering algorithms have shown that they work well in text data, the new needs arising from these complex data sets continue to motivate researchers to develop new clustering algorithms. Co-clustering is a good example of a new application need where analysts require the clusters of both documents and words. Dhillon has formulated it as the bipartite graph partitioning problem [Dhi01]. Given that the close relationship between spectral techniques and graphs, Dhillon developed a spectral co-clustering algorithm based on the second left and right singular vectors of an appropriately scaled word-document matrix. Subsequently, several co-clustering algorithms rooted in information theory and fuzzy mathematics were proposed, e.g., information-theoretic method [DMM03] and fuzzy based method [KDK03].

Microarray data: Characterized by its tens of thousands of features but few samples, microarray data analysis is an interesting problem for data mining researchers. Between the tasks of categorizing genes and samples, biologists are more interested in unsupervised learning of genes (i.e., clustering) given the lack of obvious knowledge about the genes that contribute to a particular disease, while supervised learning of samples are more useful to doctors for diagnosis and prognosis.

We shall discuss the second task, supervised learning of samples, later in Section 2.2.2. For now, we discuss the unsupervised learning of genes, in which samples become features. It is often the case that not all the genes are relevant for all the experimental conditions or samples but groups of them are often co-regulated or

co-expressed in specific conditions (or samples). Therefore, traditional clustering algorithms, e.g., hierarchical clustering algorithms, can not meet the requirements of biologist for analyzing genes' functions. Instead, a new cluster pattern, called "bicluster" was proposed, which is defined in a subset of genes that show similar patterns (e.g., co-regulations or coherence) under a subset of samples. In other words, a bicluster corresponds to a submatrix, which shows a special pattern by satisfying a criterion function or metric.

To discover this new cluster pattern, Cheng and Church [CC00] proposed a new type of clustering method, called *biclustering*, by using a new metric *mean squared residue* to measure the coherence of rows and columns in a submatrix. Since then, many biclustering algorithms were developed [YWWY02, YWWY03, WWYY02, LW03, LYW04]. For example, Yang *et al.* proposed the FLOC (FLexible Overlapped biClustering) algorithm to discover overlapping biclusters based on the probabilistic moves of rows and columns [YWWY02, YWWY03]. To obtain larger and better biclusters, Wang *et al.* proposed a new model *pCluster* and a biclustering algorithm based on the sequence mining techniques [WWYY02]. Liu *et al.* redefined a bicluster as an ordered sequence and developed the *OP-Cluster* model based on this new definition [LW03, LYW04]. These algorithms were comparable in terms of both performance and complexity. A systematic survey of biclustering algorithms have been given by Madeira and Oliveira [MO04].

Around the same time before biclustering algorithms were proposed, the KDD community has noticed that many dimensions in the data could mask the clusters from being observed within noisy high dimensional datasets. This motivated *subspace clustering*⁴ where interested clusters are defined to be embedded in the feature subspace. Subspace clustering algorithms localize the search in relevant dimensions to find clusters that exist in multiple and possibly, overlapping subspaces. According to their search strategy, there are two major branches in subspace clustering methods: top-down search iterative methods and bottom-up search grid based methods [PHL04]. The first group uses a top-down approach to find an initial clustering

⁴We use *subspace clustering* to represent this new class of clustering algorithms. In the literature of clustering algorithms, the term *subspace clustering* was used to refer to a narrow range of clustering algorithms, called *projected clustering*. In this thesis, our definition of *subspace clustering* is broad and so the *projected clustering* is just a subset of *subspace clustering*.

in the full set of dimensions and then evaluates the subspaces of each cluster by iteratively improving the results. Representative algorithms include PROCLUS [APW⁺99] and ORCLUS [AY00]. In contrast, the second group employs a bottom-up strategy to build clusters up from dense regions found in low-dimensional space. Examples of algorithms in this group include CLIQUE [AGGR98], DOC [PJAM02], ENCLUS [CFZ99], and their improved cousins such as HARP [YCN04] and EPC [NFW05].

2.2.2 Classification of High-Dimensional Data

Unlike the subfield of clustering high-dimensional data which is emerging in recent several years, classification has been an extensively researched topic for high-dimensional data in machine learning and pattern recognition. A classic case is Text Categorization (TC) that has witnessed a booming interest and becomes the major subfield since the early '90s [Seb02]. Nowadays, TC has contributed many results and experiences in learning high-dimensional data, i.e. text data, to high-dimensional data analysis in the framework of supervised learning. In this section, we follow the structure of the previous section by reviewing classification in text data and microarray data respectively ⁵. However, because there have been classic text books and review papers in introducing classification methods for high-dimensional data, we focused on presenting new results and conclusions. In practice, the high dimensionality of the data significantly impacts classification performance. In many applications, feature selection becomes an important preprocessing step for classifying high-dimensional data sets. However, there have been many evidences in text categorization to show that SVMs perform very well in text data even without feature selection [Joa02]. In fact, it appears that text categorization became successful largely due to SVMs, which was a breakthrough in statistical learning theory [Vap95, Vap98]. Given their success in text data and many other areas, SVMs and kernel-based methods are now the standard solutions to learning high-dimensional data.

Going back to microarray data analysis, one of the analytical goals is to identify a subset of genes (referred to as a *biomarkers*) that are highly correlated to a disease.

⁵In classification tasks of text data and microarray data, words and genes are features respectively. Hence, documents in text data and samples in microarray data are the objects whose class attribute will be predicted by the classifier.

Often, doing so requires interweaving the task of classifying samples with gene selection, which is also the main difference between text and microarray classification. For microarray data, there are no systematic reviews nor empirical studies to compare the available classifiers. However, there have been many experiments that showed the absence of a link between the use of a particular classifier and feature selection method. Recently, in an effort to identify the factors that can significantly distinguish classes, the idea of applying association rule mining to the classification of high-dimensional data was proposed. Given its high efficiency and its rule induction abilities, this new type of classifiers is called a rule-based classification or associative classification [HK01]. Representative works in this case include emerging patterns [DL99], and EP-based classifiers like CAEP [DL99], JEP [LDR00b] and DeEP [LDR00a]; and more recently, CPAR [YH03] and FARMER [CTX⁺04].

2.2.3 Dimensionality Reduction and Feature Selection

Another strategy to reduce the dimensionality is to keep the most informative parts in the new and low-dimensional space. Some domains view feature selection as a data preprocessing step *and* also an important outcome in the analysis. For example, selecting a few genes in microarray data for identifying biomarkers is one of the expected outcomes of analysts. Regardless, dimensionality reduction and feature selection are two important domains in high-dimensional data analysis.

Dimensionality reduction has roots in statistics, neuro-physics, data visualization and image processing. Since useful information embedded in the high-dimensional space has to be compressed into a low-dimensional space with much less volume, how to approximate and summarize the original data set in low-dimensional space is a typical problem in dimensionality reduction. By representing a high-dimensional data set as a matrix, statistics and matrix computation can be applied. Classic dimensionality reduction methods that have been studied in statistics and widely used in pattern recognition can be categorized into linear and non-linear methods.

The linear methods include Principal Component Analysis (PCA), factor analysis, and projection pursuit [FT74]. Although these linear methods can be used to approximate and summarize the original space, they may not work well in complex relationships of objects that exist in high-dimensional space. The “manifold learning” [SL00] is an intuitive example. If we consider the question of how we can map

CHAPTER 2. LITERATURE SURVEY

the relationships of 3-D points in a geometric manifold within a 2-D space, we immediately see the importance of dimensionality reduction where low-dimensional spaces exist in the high dimensional space in a non-linear manner. This led to a series of non-linear methods that have been proposed.

Examples of classic non-linear methods include the generalized principal component analysis [GW69], multidimensional scaling [CC01], and principal curve [HS89] – all of which are extensions of PCA. Of the more recent ones, Locally Linear Embedding (LLE) [RS00] and ISOMAP [TdSL00] were proposed, both of which are methods based on the geometric framework and manifold learning. The developers of these two methods considered the local neighbors of a high-dimensional data point as an important geometric information to be kept in the low-dimensional space. This trick has made their methods very effective in manifold learning, and were thus applied to the visualization and classification in data mining [VDG⁺02]. Non-linear methods have been successfully applied in domains such as similarity search and indexing [CM00, Agg01, Agg02, CTL03].

The feature selection task focuses on selecting the most informative or distinguished features. This makes it suitable for further data analysis, such as classification or clustering. In that sense, most feature selection methods are a kind of supervised learning technique whereby distinct features are identified for distinguishing class labels of samples. Although there are works in unsupervised feature selection, e.g., [DL00, DB04], we shall focus on the commonly used supervised feature selection methods. Generally, feature selection methods fall into two categories: filter methods and wrapper methods [HH03]. Filter methods select features by evaluating the goodness of features based on the intrinsic characteristics, which determines their relevance or discriminant powers with regard to the class labels. So filter methods operate independently of any learning algorithm, i.e., classification methods. Most existing filter methods use statistical tests (e.g., t-test, F-test) [JSR03] and information theory (e.g., mutual information or information gain) [PLD05] to rank features [YP97]. On the other hand, wrapper methods are closely “embedded” in an actual target learning algorithm or a classifier. The goodness and usefulness of a feature subset is evaluated by the estimated accuracy of the classifier, which was trained only with the subset of features.

Comparative studies of feature selection methods in text data were first made by Yang [YP97], and then later by Mladenić [Mla98]. Since the objective is to improve classification accuracy, most feature selection methods used in these two studies are filter methods. The results show that simple filter methods can improve the document classification performance even using a very small portion of words. However, the excellent performance of SVMs seems to indicate that SVMs are not sensitive to high dimensionality in sparse text data, and they often outperform other classifiers. For this reason, there are few new feature selection methods for text data proposed in recent years.

In the case of microarray data, the gene selection task is an active topic, where an eventual outcome of gene selection is to discover a biomarker by finding a minimal subset of genes that are not only differentially expressed across different sample classes, but also most irrelevant without redundancy. These two characteristics distinguish the task of gene selection from common feature selection tasks. The “biomarker” can be used for disease classification and the discovery of structure of the genetic network. The common way of filter gene selection methods is to rank genes and then choose the top-ranked genes, say top 50 [Ge99] or 150 genes [LZO04]. The number of the genes selected is subjectively determined with the trial-and-error process. Since ranking genes is based on a univariate scoring metric, the ranking value of a gene is computed in isolation from all other genes, or at most in combinations of pairs of genes [DUdA06]. As a result, the genes selected could be highly correlated among themselves, which raises the issue of “redundancy” in selected features [DUdA06, PLD05].

Filter methods are independent of any learning method and therefore, they have better generalization properties [PLD05, YL04]. Wrapper methods can derive a gene subset with a very small number of non-redundant genes [PLD05]. Because the characteristics of the selected genes match that of the classifier, wrapper methods often give higher classification/prediction accuracy. However, wrapper methods are computationally expensive for data sets with large number of features. More importantly, they behave like a black-box that screens analysts from details of the analysis process. Therefore, they are not extensively used in microarray data analysis [CGFW05, IBL02], although they are popular in other machine learning applications. Rather, microarray data analysis uses filter methods most of the time in favor of its computational efficiency and intuitive process. Comparative studies have been

conducted to evaluate different gene selection methods [CD04, LZO04, LLW02]. Their results show that there are no obvious winner among filter and wrapper methods.

2.2.4 Visualization of High-Dimensional Data

Data visualization has always been a significant technique of exploratory and intelligent data analysis since the '80s. International conferences, such as *International Conference Information Visualization*, *Conference on Visualization and Data Analysis*, *IEEE Visualization* and so on, have advanced this technique. The importance of this field can not be undermined and its significance can be seen in the establishment of its own domain called visual data mining.

There are several established types of visualization techniques. They include (i) geometric methods, such as scatterplots [Cle93], landscape [Wri95], parallel coordinates [ID90], projection pursuit [Hub85], and hyperslice [WL93]; (ii) icon-based methods, such as stick figures [PG88], and color icons [KK94]; (iii) hierarchical methods, such as cone trees [RMC91], and treemap [Shn92]; and (iv) pixel-oriented methods including recursive patterns [KKA95], and circle segments [AKK96]. Instead of any attempt to discuss their details, the reader may refer to [Kei02] for more information.

Instead, what we want to highlight is that dimensionality reduction needs to be done first so that any of the aforementioned visualization techniques can work in high-dimensional data sets [SL04]. Therefore, most dimensionality reduction techniques reviewed in Subsection 2.2.3 are used prior to visualization. The commonly used methods for visualizing high-dimensional data sets include PCA and MDS. In any case, since visualization seeks to provide an interactive environment for human exploration of the data, effective integration of human expertise into the visualization is important. This is done in a number of ways such as easy and intuitive navigation support of the interface [WR02], and real-time interactivity such as zooming, e.g., IVEE [AW95], details on demand, e.g., magiclens [ASP⁺93], and projection layout using hyperbolic planes and hyperspheres [WR02, GZRL05].

2.3 Graph Data Analysis

In recent years, graph data analysis techniques developed from other domains have become the mainstream of mining complex data within the KDD community. This

shift towards graph-based data mining is natural since most complex systems, such as social, biological and communication systems, are made up of the systems' components and their complex interactions which can be well captured by graph models. More and more complex networks emerge in recent years, such as World-Wide Web, Internet, metabolic and ecological networks, and many other virtual complex graphs obtainable through data transformation and mapping [New03, Alb01]. As this shift towards graph-based data mining takes central stage, in this section, we shall review graph models and techniques studied in other fields, like physics, mathematics, artificial intelligence and so on, which may inspire novel graph data analysis techniques to be developed in the KDD community.

2.3.1 Basics of Graphs

Mathematically, a graph G is represented as a triple $G = (V, E, f)$, where V is a set of vertices, E is a set of edges connecting some vertex pairs in V , f is a mapping $f : E \rightarrow V \times V$. With different settings of V and E , we can get different graphs, such as tree (also called acyclic graph), bipartite graph, complete graph, regular graph and some special graphs [BLS99]. There are many classic graph problems in graph theory. The solutions of these problems advance the development of graph theory and eventually form basic blocks of many graph applications (see [Sed01]).

Generally, a graph is represented as an adjacency matrix. It is a $|V|$ -by- $|V|$ array of boolean (or real) values with the entry in row u and column v defined to be 1 (weight of an edge) if there is an edge connecting the vertex u and the vertex v in the graph, and to be 0 otherwise. As a result of this representation, graphs are closely related to matrix theory. For some applications of graphs, there is a need to compare the similarity of two graphs. Hence, several similarity measures have been considered in the literature. Many of them involve edge transformations, such as edge rotations, edge slides, and edge jumps, where the class of graphs involved is that of a fixed size or a subclass thereof [CKS98]. There are also some meaningful metrics for characterizing graph structures that emerged from the field of social network analysis that further provides practical insights into the nature of graphs. They include centrality, betweenness, reciprocity, reachability, prestige, balance, cohesiveness, and equivalence.

2.3.2 Graph-based Data Mining

The advances in data mining and the rising needs of modern applications helped researchers realize the limitations of traditional attribute-value and item-set representations in domains such as networks, Web analysis, text mining, biology, and chemistry. Consequently, this motivated new directions in data mining with emphasis on alternative (and more expressive) representations. In mathematics, graph is one of the most generic topological structures that has, in the last few years, become the basis of graph mining – an active research direction within the KDD community. Without any surprise, the emerging complexity of data in the domains that we cited above is why graphs are becoming the preferred representation. It has the following advantages:

Expressiveness of real-life data sets Most data in the real world can be expressed naturally as graphs. Examples include hypertexts from World Wide Web, Internet networks, and social networks (e.g., networks for fund-transfers in stock markets, switches in telephone-calls, relationships between actors and their co-starred movies, scientists and their coauthored papers, maps from city streets, biological networks from gene sequences and metabolism, circuits and even program structures constructed by the compiler, etc.). These examples demonstrate the range of applications for which the graph representation is the appropriate abstraction.

Profound foundations Graph theories have been extensively studied for hundreds of years in mathematics, and graph analysis techniques have been explored in artificial intelligence since the early days of computer science. As a consequence, many important and useful properties of graphs were proven. In recent years, there is also an increasing interest in understanding statistical properties of real-world networks in the field of statistical physics. The theories developed in these fields eventually built profound foundations for graph mining research.

Given that most graph analysis techniques developed are “problem-driven” and hence are mostly dependent on applications or domains, we shall organize and introduce these techniques by their origins and application domains. This is different from

the survey made by Washio and Motoda [WM03]. Our presentation is more meaningful for readers to understand how these techniques originate, how their nature is, and what potential values they possess. In addition, we attempt to introduce some potential methods that have not been applied to any real-world problem, as we believe they would be promising research directions for graph mining in the near future.

Graph theories in mathematics Spectral graph theory is a component of spectral analysis that originates from mathematics. It is the study of relationships between a graph and the eigenvalues of matrices (such as adjacency matrix) that are naturally associated to this graph [Chu97] (refer to Section 2.4.4 in this chapter for more details). The set of eigenvectors of a graph is another component of spectral analysis, which has been successfully applied to link analysis of the World Wide Web for improving information retrieval [PBMW98, Kle99] and to data clustering [SM00, NJW01]. While several applications of spectral analysis have been introduced [DHZ01a], there remain many open problems in understanding networks with large-scale and random properties. Many of these problems, we believe, may be solved by random graph theory.

Graph theories in physics Models and theories from statistical physics have also provided abundant materials and fertile sources of new ideas to solving complex network problems. Here, we introduce three theories from this domain that may be significant contributors to solving complex network problems. Mean Field (MF) methods are deterministic methods that make use of tools such as Taylor expansions and convex relaxations to approximate or bound quantities of interest. Their main idea is to focus on one particle and assume that the most important contribution to the interactions of such particle with its neighboring particles is determined by the mean field due to the neighboring particles. The Cellular Automata (CA) model was conceived by Ulam and Von Neumann in the 1940s to provide a formal framework for investigating the behavior of complex and extended systems [Neu66]. A CA consists of a grid of cells, each of which can be in one of a finite number of k possible states, updated in discrete time steps according to a local interaction rule. This rule (also known as the transition function or interaction rule) normally operates on each cell. Hence, the state

of any single cell is determined by previous states of the cells in its neighborhood [Wol84, TM87]. The Markov Random Field (MRF) theory is a branch of probability theory for analyzing spatial or contextual dependencies of physical phenomena. It has been used in visual labeling to establish probabilistic distributions of interacting labels. MRF provides a convenient and consistent way of modeling spatially correlated features. This is achieved through characterizing mutual influences among such entities using MRF probabilities. MRF has been extensively applied in image processing and computer vision [Li95]. All these methods have potential values in establishing insights to the understanding of graphs. To date, MRF has been applied in graph-based data mining [DR01].

Traditional Artificial Intelligence (AI) Ever since the beginning of computer science, artificial intelligence investigates intelligent search algorithms in graphs. These search algorithms are categorized into three different types based on their strategies: (1) Branch and bound strategy – includes best-first search, depth-first and branch-and-bound algorithms; (2) Greedy and local strategy – designed to iteratively improve the objective function by searching for better solutions in a local neighborhood of the current solution; (3) Stochastic-based strategy – includes genetic algorithms and simulated annealing algorithms. Most classic graph problems are typically solved by these search algorithms. Another important research field in AI is the Neural Network (NN), which itself is a graph-based model. There are many types of neural networks, such as back propagation NN, Bidirectional Associative Memory (BAM) and so forth. NN has been applied in many fields, including data mining.

Data mining and machine learning Kernel methods, especially SVMs, have become popular tools in machine learning and data mining. They are computationally attractive because of the low cost in computing the feature map. Research in kernel methods has recently turned towards kernel functions defined on graphs [Gär03]. This is a new direction for knowledge discovery in graphs. In traditional data mining, there are major works on discovering useful patterns or knowledge in large and real-world graphs. An example is the Apriori-based Graph Mining (AGM) system [IWM03]. The basic principle of AGM is similar to the Apriori algorithm for basket analysis. It finds frequent substructures,

and thus the problem of finding frequent itemsets is generalized to frequent subgraphs. After AGM, a family of graph-based data mining algorithms have been proposed. They include the SUBDUE system [CH00], a faster version of SUBDUE called FARMER [NK01], the MolFea approach to find characteristic paths from graphs [RK01], and the ANF [PGF02], which is a fast and scalable tool for analyzing massive graphs in the real world.

Social Network Analysis (SNA) Aimed at uncovering interaction patterns of people, and based on the intuition that these patterns are important features of the lives of the individuals who display them, academics in social science observe networks of people using sociometric techniques and have been developing a set of techniques to provide both visual and mathematical analysis of human relationships. It is now becoming mainstream due to its better techniques for tracing relationships. Research in this area now focuses on mining graphs extracted from relationships among customers for marketing. Relationships among customers are of potential marketing tools in many businesses. Algorithms of discovering potential customers for effective business objectives and network values for virus marketing were proposed in the last three years [DR01, ONL02]. Models and processes in analysis of the customer relationships have also been proposed, including the MRF model, independent cascade model, and linear threshold model [DR01, KKT03].

2.3.3 Characteristics and Mechanisms of Complex Networks

The main contributors of graph theories are mathematics and physics. Mathematicians consider mathematical aspects of graphs purely in its abstract and mathematical representation, whereas physicists focus mainly on the mechanics of complex networks in the real world, as in how they emerge and evolve, and impact the understanding of complex systems.

There is a long history of graph theory in mathematics that originates from the 18th century in the work of Leonard Euler's work. Since the middle of the 20th century, graph theory has evolved greatly from a statistical and algorithmic standpoint. The theory of random graphs was founded by Paul Erdős and Alfréd Rényi, after Erdős discovered that probabilistic methods are useful in tackling problems in

CHAPTER 2. LITERATURE SURVEY

graph theory [ER59, ER60], leading to the construction of the Erdős-Rényi model. However, the real-world data is usually difficult to describe using random graphs as they often exhibit the small world and power law phenomena. Since random graphs do not capture these characteristics, a different model known as the power-law random graph (a.k.a. scale-free random graph) [Bol01, ACL00] was proposed. In this model, the degree distribution is a constraint, but all other respects are random. In other words, although the edges connect randomly selected vertices, but the degree distribution is restricted by the power law. A comprehensive theory with given degree distribution was developed by Newman, Watts and Strogatz using a generating function to calculate a variety of quantities on a large graph with an arbitrary degree distribution [New01]. Some of its properties were discussed in [GMS03, CLV03].

Graph theory also has close relationships with other mathematical areas, two of which are matrix theory and linear algebra. They have been effective methods for the analysis of adjacency matrices of graphs. As a matter of fact, both matrix theory and linear algebra form the basis of spectral graph theory. In spectral graph theory, the eigenvalues are closely related to almost all major invariants of a graph, linking one extremal property to another. Therefore, eigenvalues play a central role in our fundamental understanding of graphs [Chu97].

As aforementioned, various networks from the real world are natural prototypes of graphs. Therefore, it is natural that there have been several works in physics reporting the mechanics of real-world networks that later form an important component of graph theories. In the last several years, the field of statistical physics has made several important observations on some massive networks. The study of statistical mechanics of complex networks soon became an active field responsible for understanding the nature of these networks. Two amazing phenomena in complex networks as the hallmark of complex systems, i.e., the power law and small world phenomena, were extensively studied in this field. They are pervasive in complex network of all forms, for example, human language [FS01], networks of scientific collaborators [New01], WWW [WS98], metabolic network [WF00], software architecture [VS03], and electronic circuits [FJS01]. Recently, the analysis of co-authorship relationships from notable conferences (SIGMOD, PODS, VLDB and ICDE) also showed that this graph is small world [NSP03] (<http://database.cs.ualberta.ca/coauthorship>). The

implication of these works revealed that the small world phenomenon has great importance in many applications.

One of them is the observation that individuals in the network have only limited local information about the global network, and therefore, finding short paths between two individuals is a nontrivial decentralized search effort [Kle00, WDN02, DMW03, Gra03]. Moreover, recent research shows that the small world and power law phenomena are two coexisting properties of complex networks [SCMV02, RB03]. The hierarchical organization model was then proposed to simultaneously capture these two properties [RB03, BDR⁺04]. One of applications of this observation is the growing and navigating of the Web by local content [Men02]. The other application is the halting of virus spreading in such networks [DB02]. A more direct application is the link analysis of graphs with these two properties [Hac03].

2.4 Data Analysis by Spectral Techniques

In a broader sense, spectral computation is a cross-discipline technique with multiple foundations in computer science including matrix computation and linear algebra, complex systems and discrete dynamical system, and graph theories and algorithms. Despite its long history, it is still a very powerful data analysis technique as researchers from different domains have shown in recent years. In particular, we saw new data analysis techniques being developed for high-dimensional data and graphs where spectral techniques are fused with those from machine learning, data mining, information retrieval, image processing, and statistical mechanics. While many insights of spectral techniques have been revealed in data analysis, we agree with Marina Melia (as what she wrote at <http://www.stat.washington.edu/mmp>) that the “full potential of” spectral techniques “is still unexplored”. In this section, we discuss spectral techniques from a number of perspectives.

2.4.1 Historical Background

Most readers may be familiar with Principal Component Analysis (PCA; see Section 2.2.3) and Singular Value Decomposition (SVD) in linear algebra, which belong to spectral techniques. They are correlated to matrix decomposition operations, where after decomposition, its leading eigenvalues and eigenvectors can be used to construct

the embedding space that can approximate the original space. This is the case for most readers who are familiar with statistics and linear algebra. However, we believe that spectral techniques reveal more than that and whatever meaning statistics and linear algebra give to spectral techniques is simply the tip of the iceberg. In fact, we argue that there is more to be explored as reflected by a large number of applications in different domains (see Section 2.4.3). In the following, we shall introduce the mathematical preliminaries needed to understand spectral techniques, after which, the intuitions in spectral techniques are discussed. In the end, we discuss the well-known applications that result from exploiting those intuitions.

2.4.2 Preliminaries

Let the data of interest be represented by the matrix A . In high-dimensional data, rows of A represent objects and columns correspond to attributes. In the case where the data is a graph, A is the adjacency matrix of this graph. We define two spectral operations on A , i.e., decompositions of a matrix.

EIG(A): Given a real symmetric matrix $A_{n \times n}$, if there is a vector $x \in \mathbb{R}^n \neq 0$ such that $Ax = \lambda x$ for some scalar λ , then λ is called the eigenvalue of A with corresponding (right) eigenvector x . **EIG(A)** is the operation to compute all eigenvalues and their corresponding eigenvectors of A . All eigenvalues and eigenvectors are real, which can be guaranteed by the real Schur decomposition theorem [GL96].

SVD(A): Given a real matrix $A_{m \times n}$, there always exist two orthogonal matrices $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ (where $U^T U = I$ and $V^T V = I$) to decompose A to the form $A = USV^T$, where $S = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{r \times r}$, $r = \text{rank}(A)$ and $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r = \dots = \sigma_n = 0$. Here, the σ_i are the singular values of A and the first r columns of U and V are the left and right (respectively) singular vectors of A . **SVD(A)** is called Singular Value Decomposition of A [GL96].

Usually, the set of eigenvalues (or singular values) is called the spectrum of A . Also, eigenvectors (or singular vectors) are important components of spectral methods. They are principal tools in spectral methods for data analysis.

2.4.3 Intuitions of Spectral Components Underlying its Applications

With mathematical preliminaries, we start to discuss intuitions of spectral techniques and consequently, the insights provided by these intuitions help understanding well-known applications of spectral techniques to be introduced next.

Spectral Graph Theory It historically rooted from the fact that fundamental properties of a graph can be obtained from spectral components that are decomposed from the graph's adjacency matrix. In particular, eigenvalues are closely related to almost all major invariants of graphs and thus, are fundamental to the understanding of graphs. This spurs many recent developments in spectral graph theory [Chu97].

Optimization From a mathematical perspective, spectral decomposition of a matrix is essentially an optimization process as shown by some notable theorems below.

Theorem 2.1 (Eckart-Young, see [GR71]) *Given a matrix $A \in \mathbb{R}^{n \times m}$ (supposing $n > m$), and its singular value decomposition, $A = U\Lambda V^T$, where $U \in \mathbb{R}^{n \times m}$, $\Lambda \in \mathbb{R}^{m \times m}$ is a diagonal matrix of singular values, and $V \in \mathbb{R}^{m \times m}$, such that $U^T U = I$ and $V^T V = V V^T = I$, with singular values arranged in decreasing sequence, $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_m \geq 0$. Then among all $n \times m$ matrices B of rank k ($k < m$), A_k is the one that minimizes $\|A - B\|_F^2$, where $\|\cdot\|_F$ is the Frobenius form of a matrix⁶, and $A_k = U\Lambda_k V^T$ (Λ_k is the $m \times m$ diagonal matrix with diagonal elements as $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_k > \lambda'_{k+1} = \lambda'_{k+2} = \cdots = \lambda'_m = 0$).*

Geometrically, the Eckart-Young Theorem states that the least square approximation in k dimensions of the matrix A can be found by replacing the smallest $(m - k)$ roots of Λ with zeroes and re-multiplying $U\Lambda V^T$. Such low-rank approximation of a high-dimensional data set has been widely accepted as a general solution to capture the true degree of freedom (or the meaningful dimensions) in the data [AM01, FKV04, Ye04]. The Latent Semantic Indexing (LSI) is a well-known application [DDL⁺90].

⁶ $\|X\|_F = \sqrt{\sum_{ij} X_{ij}^2}$

Theorem 2.2 (Courant-Fischer Min-Max, see [GL96]) *Let A be real symmetric. Then for $k = 1, 2, \dots, n$,*

$$\lambda_k(A) = \max_{\dim(S)=k} \min_{0 \neq y \in S} \frac{y^T A y}{y^T y} \quad (\text{Eq. 2.1})$$

where $\lambda_k(A)$ represents the k -th largest eigenvalue of A , S is a certain subspace, and $\dim(S)$ is its number of dimensions.

The connection between a graph problem and a linear (or quadratic) programming problem has been established in graph theory. As a matter of fact, the numerator in the right term of Eq. 2.1 is the standard quadratic expression that corresponds to a metric of the graph represented by the matrix A (see Section 5.1.2). Hence, the Courant-Fischer Min-Max Theorem can be applied to graph data analysis. The Normalized Cuts (NCuts) algorithm, a well-known image segmentation technique, is a notable application [SM00]. In addition, both the Google's PageRank algorithm [PBMW98] and Hyperlink-Induced Topic Search (HITS) algorithm [Kle99] in Web information retrieval are related to this theorem.

Discrete Dynamical System (DDS) It is the study of how things change over time [San90]. The typical process of DDS includes a network with initial link/node weights (a.k.a. the status of the system) and the iterative rules of how these weights change over time. It is interesting that the mathematical representation of DDS is often in the form of linear or quadratic expression (if functions of these rules are linear). In general, neural network is a classic DDS. The study of DDS shows that the convergence or stability of DDS's status over time is closely related to spectral components of the DDS's network [San90]. This has been demonstrated in two categories: (i) the classic Hopfield network [Hop82] and the excitatory-inhibitory network [SRLH97], whose dynamics are related to spectral components of network's weights; and (ii) mutually reinforcing relationships between hub vertices and authority vertices in the HITS algorithm [Kle99], two of whose extensions include a non-linear DDS [SRLH97] and a categorical data clustering algorithm proposed by Gibson et al. [GKR00].

2.4.4 Applications of Spectral Analysis in Various Domains

Considering the history of spectral techniques, we see that each application unveils an aspect of insights of spectral techniques. To better understand intuitions introduced above, applications of spectral techniques in various domains are discussed below.

Spectral Clustering Pioneering works in spectral clustering are spectral graph partitioning algorithms based on the Laplacian transformation ⁷ of the graph's adjacency matrix [Fie73, DH73]. Subsequently, those based on the Laplacian transformation, e.g., ratio-cut clustering algorithms [HK92, CSZ94] and the normalized Laplacian transformation, e.g., the NCuts algorithm [SM00] were proposed. The NCuts algorithm caught the interests of the KDD community in recent years. For example, the NCuts algorithm was extended to partition bipartite graphs [ZHD⁺01, Dhi01]. In addition to the NCuts algorithm which is characterized by the recursive way (refers to the use of only one eigenvector to recursively bisecting the data), spectral clustering algorithms based on multi-way (refers to the use of k eigenvectors as the new space and clustering the points in this space with general clustering algorithms) were proposed, including PDDP [Bol98], MMC [DHZ⁺01b], NJW [NJV01], KVV [KVV00], and Meila-Shi [MS01]. These algorithms are reviewed in [Din04, VM03]. In addition, recently many new properties of spectral components have been proven, e.g., random walks [MS01], self-aggregation [DHZS02], relationship with k -means algorithms [ZDG⁺01], and semi-definite relaxation [XJ03].

Spectral Analysis of Graphs The advancements in spectral graph theory [Chu97] and algebraic graph theory [RG01], have made spectral computation one of the most important graph data analysis techniques in discovering combinatorial patterns in large-scale graphs. As a matter of fact, most spectral clustering algorithms, such as NCuts [SM00], MMC [DHZ⁺01b], and KVV [KVV00], can be viewed as graph partitioning algorithms. In recent years, more algorithms based on spectral computation were developed for efficiently mining massive graphs, e.g., a separation method applied to Web graph analysis [DHZ01a], a SVD-based method for clustering graphs [DFK⁺04], and the

⁷Laplacian and normalized Laplacian are transformations of a symmetric matrix with positive values (see Chapter 3).

revelation of relationships between PCA and spectral clustering algorithms on graphs [SFYD04]. Similar works were also done in other areas, such as Internet topologies [VHE01, MGZ03], cell-graphs of cancer [DGY04], social network analysis [SR03], and the structure understanding of complex networks [dABY05].

Application to Data Classification Given the good performance of spectral analysis of high-dimensional data and graph data, spectral techniques also found their applications in classification. In transductive learning [Vap98], Joachims introduced a new method to overcome the problems brought by greedy search. It is based on spectral graph partitioning that was used to arrive at the global optimum of the problem formulated in transductive learning [Joa03]. With more results providing insights of spectral analysis of data, there is an increasing interest to incorporate knowledge obtained from spectral analysis into learning models to improve classification. Recently, several works have made efforts. An example is the latent semantic kernels that integrates semantics discovered from latent semantic analysis of text into kernels [CSTL01]. After that, a spectral classifier incorporating spectral clustering information from unlabeled and labeled data was also proposed [KKM03].

Application to Information Retrieval Deerwester *et al.* [DDL⁺90] introduced Latent Semantic Indexing (LSI) in 1990. Based on the SVD operation of the word-document matrix, LSI projects the original space to the spectral subspace which can reveal more latent semantics among terms and documents than the original space ⁸. Subsequently, Hofmann proposed the Probabilistic LSI (PLSI) which is an automated document indexing approach based on the statistical and generative latent class model [Hof99]. In contrast to traditional information retrieval mainly indexing free text, the Internet era has provided opportunities to integrate more useful information into information retrieval (such as links of Web pages), and to revolutionize indexing techniques (such as HITS [Kle99] and

⁸For example, if two documents with the same topic are expressed by different words, their cosine similarity value will be zero when using the word vectors to express these two documents. However, LSI can provide a high similarity value to these two documents by investigating the whole text collection. A small example in [LFL98] illustrates this point. A theoretical and probabilistic explanation of LSI was given by Papadimitriou *et al.* [PTRV98]

PageRank [PBMW98]). Spectral computation has become an important component of these novel Web information retrieval techniques [KT99]. For example, the ranking framework [JK01] introduced by the HITS algorithm has generated a set of algorithms called Link Analysis Ranking (LAR). Most LAR algorithms are essentially built upon DDS. Those using linear DDS correspond to spectral computation [AFKM01, NZJ01], while the LAR algorithm making use of non-linear DDS [Tsa04] can be viewed as an extension of spectral computation.

Application to Dimensionality Reduction and Similarity Search As more and more high-dimensional data sets, e.g., images and multimedia data, are available nowadays, efficient operations associated with these databases (e.g., query processing) became important. Efficient retrievals of such data require a technique called similarity search (a.k.a. nearest neighbor search) where there is a need to perform dimensionality reduction. Techniques used for dimensionality reduction include PCA and several non-linear methods recently proposed, such as Isomap [TdSL00], Locally Linear Embedding (LLE) [RS00], and Clustering with Singular Value Decomposition (CSVD) [CTL03]. All of them explicitly or implicitly used spectral computation in their algorithms. A theoretical discussion [HLMS04] provides a thorough and comprehensive analysis of these techniques in the view of kernel-based methods.

Application to Data Compression and Approximation Recalling the Eckart-Young Theorem, SVD operations on a matrix is equivalent to a global optimum approximation in the low-rank subspace. Such approximation is a kind of data compression by capturing principal signals or information. This result has been widely used in image processing for image compression and noise reduction [YL95, WR97]. In fact, the HITS algorithm is theoretically a 1-rank approximation of the adjacency matrix of the Web graph. The other example is the PROXIMUS algorithm [KG03] that modifies SVD and thus is a non-orthogonal matrix decomposition method based on SVD. It can be applied to data compression, clustering and pattern discovery in high-dimensional and discrete-attribute data sets [KGR05, KGR].

Chapter 3

Enhancing the Effectiveness of Clustering with Spectrum Analysis

The bulk of data mining research is devoted to the development of techniques that solve a particular problem. Often, the focus is on the design of algorithms that outperform previous techniques either in terms of speed or accuracy. While such effort is a valuable endeavor, the overall success of knowledge discovery, i.e., the larger context of data mining, requires more than just algorithms for the data. With an exponential increase in the complexity and volume of data in recent years, an important and crucial factor to the success of knowledge discovery is to narrow the gap between the user and the available machinery.

A good example to argue a case for the above is clustering. In clustering, there is usually a requirement to set some parameters. Often, these parameters directly or indirectly control the number of clusters, i.e., k , to return. In the presence of different data characteristics and analysis contexts, it is often difficult for the user to determine the correct value of k in the data set [SC03, TWH00]. Therefore, setting these parameters require either detailed pre-existing knowledge of the data, or the time-consuming trial and error. Moreover, in the latter case, the user also needs sufficient knowledge to know what is a good clustering. Worse, if the data set is very large or has a high dimensionality, the trial and error process becomes very inefficient.

Furthermore, certain algorithms require a good estimate of the input parameters. For example, the EM [DLR77] algorithm is known to perform well in image segmentation [EAd03] when k and the initialization parameters are close to their true values. Yet, one reason that limits its application is the difficulty of estimating a k that is

close to this true value. Likewise, a poor parameter setting (which indirectly determines k) in CLOPE [YGY02] can dramatically increase its runtime. In all cases above, the user is likely to devote more time in parameter tuning than knowledge discovery. Clearly, this is undesirable.

To further strengthen the case, many analytical applications, e.g., information retrieval and pattern recognition, etc., require various forms of data preparation such as data cleaning. One of these approaches in data cleaning is to use clustering algorithms [AC03, KN98] to remove noise (or outliers). Again, if the user is unfamiliar with the nature of the data set in question, deciding a value for k becomes a daunting task that can affect the effectiveness of data cleaning [CFT03]. Since data preparation is such a critical step to effective analysis, getting the right value of k to start the process is important.

In this chapter, we provide a concrete instance of the above problem by studying this issue in the context of complex data sets, e.g., text collections, images, biological data, etc. Such data sets are inherently large in size and have dimensionality in magnitude of hundreds to several thousands. And considering the domain specificity of the data, getting the user to set a value for k becomes a challenge. In this case, we argue that a good starting point is to initialize k to the possible number of clusters. This gives rise to the fundamental question: *How can we effectively estimate the possible number of clusters for a given data set?* In our attempt to answer this question, we made the following contributions in this chapter.

- We develop an accurate and efficient method to determine k of a given data set — Our solution is to perform a spectrum¹ analysis on the similarity space of the data set by analyzing the eigenvalues (*not* eigenvectors) that encode the answer to our question. As our method does not require the user to specify a range of k to test with a clustering algorithm, the analysis is also highly efficient. We present the detail in Sections 3.1 and 3.2.
- Recognizing the subjectivity of clusters, we extend our work to estimate a range of k of a given data set — Since the outcome of clustering depends largely on the similarity measure used and the analysis context, computing a rigid value

¹Given a real symmetric matrix, spectrum is the set of all its eigenvalues.

of k may be inappropriate in some situations. We show that our method can be easily extended to handle this real-life issue by further spectrum analysis. This is discussed in depth in Section 3.3.

- We demonstrate how our solution can be realized in the context of a larger system by presenting an algorithm to automate this preprocessing — If the ultimate goal is to narrow the gap between the user and the available machinery, our solution should be automated so that it can be implemented on real-life systems in a non-intrusive manner. Fundamental to this is an algorithm that exploits our method to determine k . We introduce this algorithm in Section 3.5 together with a motivating example in Section 3.4.
- We report a wide range of experimental results that empirically support the effectiveness of our method on real-life data sets. In all cases, our approach performed well in terms of accuracy and speed. Sections 3.2.3, and 3.6 report these results.

3.1 Preliminaries

Most algorithms perform clustering by embedding the data in some similarity space determined by some similarity measures, e.g., cosine similarity [JMF99]. Let $\mathbf{S} = (s_{ij})_{n \times n}$ be the similarity space matrix, where $0 \leq s_{ij} \leq 1$, $s_{ii} = 1$ and $s_{ij} = s_{ji}$, i.e., $\mathbf{S}$ is symmetric. Further, let $\mathcal{G}(\mathbf{S}) = \langle V, E, \mathbf{S} \rangle$ be the graph of $\mathbf{S}$, where V is the set of n vertices and E is the set of weighted edges. Each vertex v_i of $\mathcal{G}(\mathbf{S})$ corresponds to the i -th column (or row) of $\mathbf{S}$, and the weight of each edge $\widehat{v_i v_j}$ corresponds to the non-diagonal entry s_{ij} . For any two vertices (v_i, v_j) , a larger value of s_{ij} indicates a higher connectivity between them, and vice versa. For ease of exposition, we also refer $\mathbf{S}$ as the adjacency matrix of $\mathcal{G}(\mathbf{S})$ when $s_{ii} = 0$.

In cases where a single similarity measure is used, it is actually possible to analyze the spectrum distribution of $\mathcal{G}(\mathbf{S})$ directly. However, there are occasions where multiple similarity measures need to be considered due to different analysis contexts. In such a situation, there is a need to first perform normalization of the different adjacency matrices to ensure comparability. As an example, Figure 3.1(a) and (b) shows two different similarity matrices on the same data set. While the similarity measure

Figure 3.1: A comparison of two similarity matrices before and after the 3-normalization: (a) $\mathbf{S}_1$ with 2 disjoint clusters – cluster 1 has 30 instances and average similarity of 0.5 and cluster 2 with 70 instances with average similarity of 0.3; (b) $\mathbf{S}_2$ with the same disjoint clusters as $\mathbf{S}_1$ but both clusters have an average similarity of 1; (c) normalized $\mathbf{S}_1$; (d) normalized $\mathbf{S}_2$.

is different, the clustering result turns out to be the same. The only difference is that the first similarity matrix $\mathbf{S}_1$ has a totally different spectrum (as represented by the color) from that of $\mathbf{S}_2$ because of the different range of similarities within each cluster. Since they have the same clustering, it is therefore necessary to normalize the adjacency matrices $\mathbf{S}_1$ and $\mathbf{S}_2$ so that they also have similar spectrum of $\mathcal{G}(\mathbf{S}_1)$ and $\mathcal{G}(\mathbf{S}_2)$ respectively.

Definition 3.1 (Normalization of $\mathbf{S}$) Given a similarity matrix $\mathbf{S} = (s_{ij})_{n \times n}$, the normalization of $\mathbf{S}$ by p -norm (or p -normalization) is denoted as $L_p(\mathbf{S}) = (\ell_{ij})_{n \times n}$ where

$$\ell_{ij} = \frac{s_{ij}}{\sqrt{\|\vec{s}_i\|_p \|\vec{s}_j\|_p}}$$

and $\|\vec{s}_i\|_p = (\sum_j s_{ij}^p)^{\frac{1}{p}}$ with $p > 0$.

The p -norm presents some interesting properties. One of them is that $L_p(\mathbf{S})$ remains symmetric and its diagonal entries are set to 0. Furthermore, regardless of the value of p , the p -norm on $\mathbf{S}$ has the property of standardizing the adjacency matrix without affecting the underlying clustering structure in $\mathcal{G}(\mathbf{S})$. Continuing our example, Figure 3.1(c) and (d) shows the normalized $\mathbf{S}_1$ and $\mathbf{S}_2$ after applying a 3-normalization, i.e., $p = 3$, where a similar result can also be obtained with other values of p . Theorem 3.1 gives this formal guarantee.

Theorem 3.1 (Properties of $L_p(\mathbf{S})$) Let $\mathbf{S} = (s_{ij})_{n \times n}$ with $0 \leq s_{ij} \leq 1$. Then, the p -norm of $\mathbf{S}$, i.e., $L_p(\mathbf{S}) = (\ell_{ij})_{n \times n}$ satisfies (i) $\ell_{ij} = \ell_{ji}$; (ii) $\ell_{ii} = 0$; (iii) $0 \leq \ell_{ij} \leq 1$.

Proof: The p -norm of a vector $\vec{x} = (x_1, \dots, x_n)$ [GL96] has the property $\|\vec{x}\|_1 \geq \|\vec{x}\|_2 \geq \dots \geq \|\vec{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$, and therefore we have

$$\ell_{ij} = \frac{s_{ij}}{\sqrt{\|\vec{s}_i\|_p \|\vec{s}_j\|_p}} \leq \frac{s_{ij}}{\sqrt{\|\vec{s}_i\|_\infty \|\vec{s}_j\|_\infty}} = \frac{s_{ij}}{\sqrt{\max_{1 \leq j \leq n} |s_{ij}| \max_{1 \leq i \leq n} |s_{ij}|}} \leq 1 \quad (\text{Eq. 3.1})$$

In addition to comparability reason, $L_p(\mathbf{S})$ is also closely related to the field of spectral graph theory. In particular, when $p = 1$, the spectrum of $L_p(\mathbf{S})$ reveals the possible number of clusters in $\mathcal{G}(\mathbf{S})$; and when p is increased beyond 1, further clustering information can also be discovered.

Once normalized, we can obtain the spectrum of $L_p(\mathbf{S})$ by computing the eigenvalues as follows. Notably, we can utilize many mature efficient solutions in matrix computation by this formulation. Let $\mathbf{A}$ denote a symmetric $n \times n$ real matrix, λ an eigenvalue of $\mathbf{A}$ that satisfies $\mathbf{A}\vec{x} = \lambda\vec{x}$ where $\vec{x}$ is a vector. It is known [GL96] that all eigenvalues of $\mathbf{A}$ are real-numbers and that each of the n eigenvalue corresponds to a unique $\vec{x}$. Without loss of generality, these n eigenvalues are denoted as $\text{eig}(\mathbf{A}) = \{\lambda_1(\mathbf{A}), \lambda_2(\mathbf{A}), \dots, \lambda_n(\mathbf{A})\}$ that we refer to as the spectrum distribution. These eigenvalues are assumed to be in decreasing order, i.e., $\lambda_1(\mathbf{A}) \geq \lambda_2(\mathbf{A}) \geq \dots \geq \lambda_n(\mathbf{A})$, where $\lambda_k(\mathbf{A})$ denotes the k -th largest eigenvalue of $\mathbf{A}$. Specific to our work, we are interested in $\text{eig}(L_p(\mathbf{S}))$, which is the p -spectrum of $\mathcal{G}(\mathbf{S})$.

Once we establish this relationship between the structural properties of $\mathcal{G}(\mathbf{S})$ and its spectrum, the answer to our fundamental question is now mapped to a matter of how to analyze $\text{eig}(L_p(\mathbf{S}))$. This approach is particularly attractive in comparison to existing methods of estimating k in terms of runtime and simplicity. More importantly, as we demonstrate in the subsequent sections, our proposal has better flexibility in terms of accommodating the different analytical contexts of the users, e.g., the use of a different clustering algorithm or evaluation index, on the same data set.

3.2 Spectrum Analysis by 1-Normalization

In spectral graph theory, the weighted Laplacian matrix is commonly used. For ease of analysis, let $\mathbf{L} = \mathbf{T}^{-1/2}(\mathbf{T} - \mathbf{S})\mathbf{T}^{-1/2}$ be the weighted Laplacian matrix of $\mathcal{G}(\mathbf{S})$ where $\mathbf{T}$ is the diagonal matrix, and $\text{diag}(d_i)$ and $d_i = \sum_j s_{ij}$ the degree of vertex v_i in $\mathcal{G}(\mathbf{S})$. By Definition 3.1, $\text{eig}(L_1(\mathbf{S})) = \{1 - \lambda \mid \lambda \in \text{eig}(\mathbf{L})\}$ and therefore, the

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

1-spectrum of $\mathcal{G}(\mathbf{S})$ maintains the same conclusions and properties of those found in $\mathbf{L}$. By this, we can establish the following three basic facts drawn from spectral graph theory. Among them, the third fact about $\mathcal{G}(\mathbf{S})$ is an important property that we exploit – it depicts the relationship between the spectrum of the disjoint subgraphs $\mathcal{G}(\mathbf{S}_i)$ and the spectrum of $\mathcal{G}(\mathbf{S})$.

Theorem 3.2 *For a graph $\mathcal{G}(\mathbf{S})$ and its 1-spectrum $\text{eig}(L_1(\mathbf{S})) = (\{\lambda_1, \lambda_2, \dots, \lambda_n\})$, where $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, we have: (i) $-1 \leq \lambda_i \leq 1$ for $i = 1, 2, \dots, n$ where $\sum \lambda_i = 0$, and $\lambda_1 = 1$; (ii) if $\mathcal{G}(\mathbf{S})$ is connected, then $\lambda_2 < 1$; and (iii) the 1-spectrum of $\mathcal{G}(\mathbf{S})$ is the union of the 1-spectrum of its disjoint subgraphs $\mathcal{G}(\mathbf{S}_i)$.*

Proof: As shown in [Chu97, GL96].

We will begin with the spectrum properties of a conceptually disjoint data set, whose chosen similarity measure achieves a perfect clustering. From this simple case, we then extend our observations to real-world data sets, and show how the value of k can be obtained.

Before the discussion of p -spectrum analysis, a standard clustering structure in the similarity matrix $\mathbf{S}$ is denoted as follows. It shall facilitate our discussion. The similarity matrix $\mathbf{S}$ is reordered by its clustering structure (k clusters) and be presented as the form of the block submatrices in Eq. 3.2.

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{11} & \cdots & \mathbf{S}_{1k} \\ \vdots & \ddots & \vdots \\ \mathbf{S}_{k1} & \cdots & \mathbf{S}_{kk} \\ n_1 & \cdots & n_k \end{bmatrix} \quad \begin{matrix} n_1 \\ \vdots \\ n_k \end{matrix} \quad (\text{Eq. 3.2})$$

with the properties: each diagonal block submatrix $\mathbf{S}_{jj}$ ($1 \leq j \leq k$) represents the intra-similarity matrix within the j -th cluster. Thus a non-diagonal block submatrix $\mathbf{S}_{ij}$ shows the inter-similarity matrix between the i -th and j -th clusters. We denote n_j as the number of rows or columns of $\mathbf{S}_{jj}$, and say $l \in \mathbf{S}_{jj}$ if $\sum_{i=1}^{j-1} n_i < l \leq \sum_{i=1}^j n_i$. Therefore, if $i \in \mathbf{S}_{jj}$, then $\vec{s}_i = (s_{1i}, s_{2i}, \dots, s_{ni})^T$, the i -th column vector of $\mathbf{S}$, belongs to the j -th cluster. And we say that $\vec{s}_i$ is in the area of $\mathbf{S}_{jj}$.

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Table 3.1: A small text collection taken and modified from [DDL⁺90] for our example. It contains the titles of 12 technical memoranda: 5 about human-computer interaction; 4 about mathematical graph theory; and 3 about clustering. The topics are conceptually disjoint with two assumptions: (i) the italicized terms are the selected feature set; and (ii) the cosine similarity measure is used to compute $\mathbf{S}$.

c1	<i>Human machine interface</i> for ABC computer applications
c2	A survey of user opinion of computer system response time
c3	The <i>EPS</i> user interface management system
c4	System and human system engineering testing of <i>EPS</i>
c5	Relation of user perceived response time to error measurement
m1	The generation of random, binary, ordered trees
m2	The intersection graph of paths in trees
m3	Graph minors IV: Widths of trees and well-quasi-ordering
m4	Graph minors: A survey
d1	Linguistic features and clustering algorithms for topical document clustering
d2	A comparison of document clustering techniques
d3	Survey of clustering Data Mining Techniques

3.2.1 A Simple Case

Assume that we have a conceptually disjoint data set, whose chosen similarity measure achieves a perfect clustering. In this case, the similarity matrix $\mathbf{A}$ will have the structure given by Eq.(Eq. 3.2) with the properties: all entries in each diagonal block matrix $\mathbf{A}_{ii}$ of $\mathbf{A}$ are 1; and all entries in each non-diagonal block matrix $\mathbf{A}_{ij}$ in $\mathbf{A}$ are 0. From this similarity matrix, we can get the corresponding graph $\mathcal{G}(\mathbf{A})$ and obtain its 1-spectrum in decreasing order [Chu97], i.e.,

$$\lambda_i(L_1(\mathbf{A})) = \begin{cases} 1, & 1 \leq i \leq k \\ 0, & k < i \leq n \end{cases} \quad (\text{Eq. 3.3})$$

Lemma 3.1 *Given a similarity matrix $\mathbf{S}$ as defined in in Eq. 3.2, where $n_1 + \dots + n_k = n$; where each diagonal submatrix $\mathbf{S}_{ii}$ satisfies $0 < n_i - \|\mathbf{S}_{ii}\|_F < \delta (\delta \rightarrow 0)$; and where each non-diagonal submatrix $\mathbf{S}_{ij}$ satisfies $\|\mathbf{S}_{ij}\|_F \rightarrow 0$ ($\|\cdot\|_F$ is the Frobenius norm), then $\mathbf{S}$ achieves a perfect clustering of k clusters. At the same time, the 1-spectrum of $\mathcal{G}(\mathbf{S})$ exhibits the following properties:*

$$\begin{aligned} \lambda_i(L_1(\mathbf{S})) &\rightarrow 1 \quad (i = 1, \dots, k \text{ and } 0 < \lambda_i \leq 1) \\ |\lambda_i(L_1(\mathbf{S}))| &\rightarrow 0 \quad (i = k + 1, \dots, n) \end{aligned} \quad (\text{Eq. 3.4})$$

Proof: Let $\mathbf{E} = L_1(\mathbf{S}) - L_1(\mathbf{A})$ and $\mathbf{E}' = \mathbf{S} - \mathbf{A}$, where $\mathbf{A}$ is as defined in Equation (Eq. 3.2). From definitions of $\mathbf{A}$ and $\mathbf{S}$, we obtain the following:

$$\left. \begin{aligned} 0 < n_i - \|\mathbf{S}_{ii}\|_F < \delta (\delta \rightarrow 0), \quad \|\mathbf{A}_{ii}\|_F = n_i \\ \|\mathbf{S}_{ij}\|_F \rightarrow 0, \quad \|\mathbf{A}_{ij}\|_F = 0 \end{aligned} \right\} \Rightarrow \|\mathbf{E}'\|_F \rightarrow 0 \quad (\text{Eq. 3.5})$$

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.2: The spectrum distribution of the collection in Table 3.1: (a) Spectrum (> 0) of $\mathcal{G}(\mathbf{S})$ and its subgraphs; (b) a graphical representation of $\mathbf{S}$. Note that all grey images in this chapter are not “plots” of the spectrum – They are a graphical way of summarizing the results of clustering for analysis purposes.

When $\|\mathbf{E}'\|_F \rightarrow 0$, this also means that $L_1(\mathbf{S})$ is much closer to $L_1(\mathbf{A})$ and thus we have:

$$\|\mathbf{E}\|_F \rightarrow 0 \quad (\text{Eq. 3.6})$$

By the property of the Frobenius norm, and the p matrix norm (where $p = 2$ [GL96]), we have:

$$\|\mathbf{E}\|_2 \leq \|\mathbf{E}\|_F \quad (\text{Eq. 3.7})$$

and

$$|\lambda_i(L_1(\mathbf{A}) + \mathbf{E}) - \lambda_i(L_1(\mathbf{A}))| \leq \|\mathbf{E}\|_2, \quad (i = 1, \dots, n) \quad (\text{Eq. 3.8})$$

where $\|\cdot\|_2$ is the $p = 2$ matrix norm. Equation (Eq. 3.7) states that the Frobenius norm of a matrix is always greater than or equal to the p matrix norm at $p = 2$, and Equation (Eq. 3.8) defines the distance between the eigenvalues in $L_1(\mathbf{A})$ and its perturbation matrix $L_1(\mathbf{S})$. In addition, the sensitivity of the eigenvalues in $L_1(\mathbf{A})$ to its perturbation is given by $\|\mathbf{E}\|_2$. Hence, from Equations (Eq. 3.5), (Eq. 3.6) (Eq. 3.7), and (Eq. 3.8), we can conclude that:

$$\lambda_i(L_1(\mathbf{S})) \rightarrow \lambda_i(L_1(\mathbf{A})), \quad (i = 1, \dots, n) \quad (\text{Eq. 3.9})$$

which when we combine with Equation (Eq. 3.3), we arrive at Lemma 3.1.

Simply put, when the 1-spectrum distribution satisfies Equation (Eq. 3.4), then $\mathbf{S}$ shows a good clustering, i.e., the intra-similarity approaches 1, and the inter-similarity approaches 0. As an example, suppose we have a collection with 3 clusters as depicted in Table 3.1. The 3 topics are setup to be conceptually disjoint, and the similarity measure as well as the feature set are selected such that the outcome produces 3 distinct clusters. In this ideal condition, the spectrum distribution (as shown in Figure 3.2) behaves as per Equation in (Eq. 3.4).

Of course, real-world data sets that exhibit perfect clustering are extremely rare. This is especially the case for data whose dimensionality is large but the data itself is sparse, e.g., *text* collections. In this case, most similarity measures do not rate two data points as distinctively similar, or different. If we perform a spectrum analysis on the data set, we will end up with a 1-spectrum of $\mathcal{G}(\mathbf{S})$ that is very different from our example in Figure 3.2. As we will see next, this 1-spectrum distribution is much more complex.

3.2.2 1-Spectrum Distribution in Large Data sets

Point (iii) of Theorem 3.2 offers a strong conclusion between $\mathcal{G}(\mathbf{S})$ and its subgraphs. However, real-world data sets often exhibit a different characteristic. If we examine their corresponding $\mathcal{G}(\mathbf{S})$, we will see that the connections between $\mathcal{G}(\mathbf{S})$ and its subgraphs are weak, i.e., Lemma 3.1 no longer holds (from the proof of Lemma 3.1, we can see that $\|E'\|_F$ will become larger and thus $L_1(\mathbf{S})$ will not be closer to $L_1(\mathbf{A})$ any more.).

Fortunately, we can still judge the cluster quality and estimate the number of clusters with spectrum analysis. In this section, we present the proofs that lead to the conclusion about cluster quality and k . But first, we need introduce the Cheeger constant. Let $SV \subset V$ of $\mathcal{G}(\mathbf{S})$. We define the volume of SV as:

$$\text{vol}(SV) = \sum_{v \in SV} d_v \quad (\text{Eq. 3.10})$$

where d_v is the sum of all weighted edges containing vertex v . Further, let $E(\delta SV)$ be the set of edges, where each edge has one of its vertices in SV but not the other, i.e., $\overline{SV}$. Then, its volume is given by:

$$|E(\delta SV)| = \sum_{v_i \in SV, v_j \notin SV} \text{weight}(v_i, v_j) \quad (\text{Eq. 3.11})$$

and by Equations (Eq. 3.10) and (Eq. 3.11), we derive the Cheeger constant:

$$h(\mathcal{G}) = \min_{SV \subset V} \frac{|E(\delta SV)|}{\min(\text{vol}(SV), \text{vol}(\overline{SV}))} \quad (\text{Eq. 3.12})$$

which measures the optimality of the bipartition in a graph. The magnitude $|E(\delta SV)|$ measures the connectivity between SV and $\overline{SV}$ while $\text{vol}(SV)$ measures the density of SV against V .

Since SV enumerates all subsets of V , $h(\mathcal{G})$ is a good measure that finds the best bipartition, i.e., $\langle SV, \overline{SV} \rangle$. Perhaps, what is more interesting is the observation that

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

no other bipartition gives a better clustering than the bipartition determined by $h(\mathcal{G})$. Therefore, $h(\mathcal{G})$ can be used as an indicator of cluster quality, i.e., the lower its value, the better the clustering.

Theorem 3.3 *Given the 1-spectrum of $\mathcal{G}(\mathbf{S})$ as $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, if $\lambda_2 \rightarrow 1$, then there exists a good bipartition for $\mathcal{G}(\mathbf{S})$, i.e., a good cluster quality.*

Proof: From [Chu97], we have the Cheeger inequality: $\frac{(1-\lambda_2)}{2} \leq h(\mathcal{G}) < \sqrt{2(1-\lambda_2)}$ that gives the bound of $h(\mathcal{G})$. By this inequality, if $\lambda_2 \rightarrow 1$, then $h(\mathcal{G}) \rightarrow 0$. And since $h(\mathcal{G}) \rightarrow 0$ implies a good clustering, we have the above.

For a given similarity measure, Theorem 3.3 allows us to get a “feel” of the clustering quality without actually running the clustering algorithm (see Subsection 3.7 for more detail). This saves computing resources and reduces the amount of time the user waits to get a response. By minimizing this “waiting time” during initial analysis, we promote interactivity between the user and the clustering algorithm. In such a system, Theorem 3.3 can also be used to help judge the suitability of each supported similarity measure. Once the measure is decided, the theorem to be presented next, provides the user a starting value of k .

Theorem 3.4 *Given the 1-spectrum of $\mathcal{G}(\mathbf{S})$ as $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, $\exists k \geq 1$ such that when $i \leq k$ there are $\alpha_i \rightarrow 1$ and $\alpha_i - \alpha_{i+1} > \delta$ ($0 < \delta < 1$) for the sequence $\alpha_i = \frac{\lambda_{i+1}}{\lambda_2}$, ($i \geq 1$), where δ is a predefined threshold to measure the first large gap between α_i ; and $k + 1$ is the number of clusters in the data set.*

Proof: Since Theorem 3.3 applies to both $\mathcal{G}(\mathbf{S})$ and its subgraphs $\mathcal{G}(\mathbf{S}_{ii})$, we can estimate the cluster quality of the bipartition in $\mathcal{G}(\mathbf{S}_{ii})$ (as well as its subgraphs). Combined with Point (iii) of Theorem 3.2, we can conclude that the number of eigenvalues in $\mathcal{G}(\mathbf{S})$ (that approaches 1 and has large eigengaps) gives the value of $k + 1$, i.e., the number of clusters.

By Theorem 3.4, we shall use the α value sequence for analysis in subsequent experiments except we clearly mention to use eigenvalues. To cite an example for the above, we revisit Table 3.1 and Figure 3.2. By the Cheeger constant of $\mathcal{G}(\mathbf{S})$, $SV = \{c_1, c_2, c_3, c_4, c_5\}$ and $\overline{SV} = \{m_1, m_2, m_3, m_4, d_1, d_2, d_3\}$ produce the best bipartition. Thus, $\mathbf{S}_{11}$ represents the inter-similarities in SV and $\mathbf{S}_{22}$ represents inter-similarities

Table 3.2: The *text* collections used in our experiments to estimate k : we selected 4 classes of **classic** with each class containing 1,000 documents; 5 **newsgroups** with each newsgroup containing 500 documents; 2 categories of the **webset** with each category containing 600 documents.

Collections	Source	# Classes	# Documents
classic	ADI/CACM/CISI/CRAN/MED	5	5559
newsgroup	UseNet news postings	17	7473
webset	Categories in Yahoo [SC02]	10	6607

in $\overline{SV}$. From Theorem 3.3, we can assess the cluster quality of $\mathcal{G}(\mathbf{S})$'s bipartition by λ_2 . Also, we can recursively consider the bipartitions of the bipartitions of $\mathcal{G}(\mathbf{S})$, i.e., $\mathcal{G}(\mathbf{S}_{11})$ and $\mathcal{G}(\mathbf{S}_{22})$. Again, the Cheeger constant of $\mathcal{G}(\mathbf{S}_{22})$ shows that $\mathcal{G}(\mathbf{S}_{22}(1))$ and $\mathcal{G}(\mathbf{S}_{22}(2))$ are the best bipartition in the subgraph $\mathcal{G}(\mathbf{S}_{22})$. Likewise, the λ_2 of $\mathcal{G}(\mathbf{S}_{11})$, $\mathcal{G}(\mathbf{S}_{22})$, $\mathcal{G}(\mathbf{S}_{22}(1))$, and $\mathcal{G}(\mathbf{S}_{22}(2))$ all satisfy this observation.

In fact, this recursive bisection of $\mathcal{G}(\mathbf{S})$ is a form of clustering using the Cheeger constant – the spectrum of $\mathcal{G}(\mathbf{S}_{22})$ contains the eigenvalues of $\mathcal{G}(\mathbf{S}_{22}(1))$ and $\mathcal{G}(\mathbf{S}_{22}(2))$, and $\mathcal{G}(\mathbf{S})$ contains the eigenvalues of $\mathcal{G}(\mathbf{S}_{11})$ and $\mathcal{G}(\mathbf{S}_{22})$ respectively (despite with some small “fluctuations”). As shown in Figure 3.2(a), λ_2 of $\mathcal{G}(\mathbf{S})$ gives the cluster quality of the bipartition $\mathcal{G}(\mathbf{S}_{11})$ and $\mathcal{G}(\mathbf{S}_{22})$ in $\mathcal{G}(\mathbf{S})$; and λ_3 of $\mathcal{G}(\mathbf{S})$, which corresponds to λ_2 of $\mathcal{G}(\mathbf{S}_{22})$, gives the cluster quality indicator for the bipartition $\mathcal{G}(\mathbf{S}_{22}(1))$ and $\mathcal{G}(\mathbf{S}_{22}(2))$ in $\mathcal{G}(\mathbf{S}_{22})$, and so on. Therefore, if there exist k distinct and dense diagonal squares (i.e., $\mathbf{S}_{ii}$ where $1 \leq i \leq k$) in the matrix, then λ_i of $\mathcal{G}(\mathbf{S})$ will be the cluster quality indicator for the i -th bipartition ($2 \leq i \leq k$), and the largest k eigenvalues of $\mathcal{G}(\mathbf{S})$ give the estimated number of clusters in the data.

3.2.3 Empirical Results

We digress for a moment to a set of experiments to provide some evidence on the viability of the 1-spectrum of $\mathcal{G}(\mathbf{S})$ to estimate k . The text data sets used are of high dimensionality and sufficiently large to reflect the real-world situation. The details of these data sets are given in Table 3.2. To verify our results, we used the gray-scale of $\mathbf{S}$ reordered by the true cluster labels. The similarity measure used is the cosine function.

In practice, we can estimate k by using Theorem 3.4. However, for the purpose of illustration, we will walk through the analysis by using eigenvalues, not α sequence. Since λ_1 is always 1, our analysis begins from λ_2 . In Figure 3.3, we have marked out

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.3: The spectrum graphs and the graphical representation of their clustering for all 3 collections: the first two data sets are conceptually disjoint, and the last is conceptually overlapping.

the large eigenvalues whose gap is larger than the rest. This gap can be best identified by the big stair steps among the eigenvalues. According to Theorem 3.4, the number of such eigenvalues (including λ_1) gives the number of clusters. We can verify this by analyzing their corresponding grey images in the same figure.

Figure 3.3(a) shows the **web2** collection with just 2 class labels and their topic being completely disjoint: *finance* and *sport*. In this case, notice that λ_2 has a higher value than the others. Since the remaining eigenvalues fall along a smooth curve, this phenomenon conforms to Theorem 3.3 and 3.4. In this case, we therefore conclude $k = 2$. At the same time, the high value of λ_2 indicates that this is a good clustering by the similarity measure used.

In the second case, **classic4** has 4 topics from scientific abstract from different research domains: *computing algorithms*, *information retrieval*, *aerodynamics* and *medicine*. They are conceptually disjoint, which can be observed from Figure 3.3(b) where there are 4 distinctive diagonal squares. From its spectrum graph, we observe that λ_2, λ_3 and λ_4 show higher values and wider gaps than other eigenvalues. Again by the same theorem, our method obtains the correct number of clusters, i.e., $k = 4$.

The third collection is the most challenging. There are 5 topics: *atheism*, *comp.sys*, *comp.windows*, *misc.forsale* and *rec.sport*. Unlike the previous two collections, the topics are not disjoint. In this case, both *comp.sys* and *comp.windows* belong to the broader topic of *comp* in the newsgroup. Therefore, the graphical representation in

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Table 3.3: Comparison of our approach against 3 well-known indexes: the Calinski and Harabasz (CH) index; Krzanowski and Lai (KL) index; and Hartigan (Hart) index with 3 well-known clustering algorithms: bisecting k -means, graph-based, and hierarchical – a ($\checkmark$) indicates a correct estimation.

	Bisecting k -means			Graph-based			Hierarchical		
	web2	classic4	news5	web2	classic4	news5	web2	classic4	news5
CH	5	2	3	3	3	3	7	2	5 ($\checkmark$)
KL	29	17	22	27	21	22	22	21	9
Hart	6	13	4 ($\checkmark$)	6	10	4 ($\checkmark$)	1	2	1

Figure 3.3(c) does not show a set of distinctive squares along its diagonal. When we apply our analysis, only λ_2, λ_3 , and λ_4 have a higher value and a wider gap than the others. So by our theorems, $k = 4$. This conclusion is actually reasonable since *comp* is more different from the other topics. If we observe the grey image in Figure 3.3(c), we see that the second and third squares appear to “meshed” together – an indication of similarity between *comp.sys* and *comp.windows*.

Furthermore, *comp.sys*, *comp.windows* and *misc.forsale* can also be viewed as one topic. This is because *misc.forsale* has many postings on buying and selling of computer parts. Again, this can be observed in the grey image. On the spectrum graph, λ_4 is much lower than λ_2, λ_3 , and closer to the remaining eigenvalues. Therefore, λ_4 may not be counted as the number of clusters. Thus, it is possible to conclude $k = 3$ by Theorem 3.4. Strictly speaking, this is also an acceptable estimation. Thus, the onus is on the user to judge the actual value of k , which is really problem-specific as illustrated in the Section 3.4.

There are many methods to estimate the number of clusters in a data set. To date, most require to choose an appropriate clustering algorithm, e.g., k -means, where it is run multiple times with predefined cluster numbers from 2 to k_{max} . The optimum k is then obtained by an internal index based on the clustering outcome. The key difference between these methods is in the index used. In this experiment, we compared our results to 3 widely used statistical methods [Gor99] on 3 well-known clustering algorithms (which are implemented in CLUTO toolkit [Kar02]. See Table 3.3).

From the table, we found that all 3 indexes managed to get only 1 out of the 3 estimations right. Although the Hart index correctly estimated 4 clusters in **news5**, it fails to handle the conceptually disjoint **web2** and **classic4**. Worse, most of the estimation are way off-track. For example, the KL index predicted $k = 29$ on **web2**! Furthermore, to the best of our knowledge, many of these methods report results

on data sets with low dimensionality. Hence, our experiment reveals how sensitive these indexes are to the choice of clustering algorithms and the dimensionality of the data set. In comparison, our approach (based on the 1-spectrum of $\mathcal{G}(\mathbf{S})$) remarkably outperforms all 3 methods in terms of accuracy.

3.3 p -normalization ($p > 1$) Spectrum Analysis

Although the 1-spectrum of $\mathcal{G}(\mathbf{S})$ gives an estimate of k for a given data set and similarity measure, this information with the single k may be restrictive for users with different analytical requirements. Due to subjectivity or domain expertise, it is possible that some overlapping clusters are viewed as one big clusters on one instance but not on another. In this case, providing a single value for k may be inappropriate.

As an example, Figure 3.4 shows a text collection with 4 topics. At the high-level, there are two conceptually disjoint clusters, i.e., $\langle \{T_1\}, \{T_2, T_3, T_4\} \rangle$, and among the three topics within the second cluster, each topic can be viewed as being overlapping or completely disjoint. On some instances, users may choose $k = 2$ as the “right” cluster numbers when looking at the data from a high-level concept. On another occasion, the user may perceive $k = 4$ (i.e., $\langle \{T_1\}, \{T_2\}, \{T_3\}, \{T_4\} \rangle$) as the “right” value when the data is dealt with at a low-level concept.

Given this situation, we now have a variation of our fundamental question: *In light of different analytical contexts, how can we suggest an appropriate value of k ?* Our solution in this case is to provide users two possible values of k that is based on different perspectives of the data set. To our knowledge, previous works rarely address this issue.

3.3.1 Differential Levels Between Clusters

Since the cause of ambiguity in the value of k lies in the overlapping of concepts, we need a method to characterize the degree of overlapping between clusters. To do so, we first define two measures that are used to compute this. Assume a similarity matrix $\mathbf{S}$ with the structure given in Eq.(Eq. 3.2), we define the *signal* and *noise* level of an i -th column vector $\vec{s}_i$ below.

Definition 3.2 (Signal and noise level of $\vec{s}_i$) *Given a similarity matrix $\mathbf{S}$ with structure similar to Eq.(Eq. 3.2) where the i -th column vector $\vec{s}_i = (s_{1i}, \dots, s_{ni})^T$*

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.4: A text collection of 275 new posters selected from **newsgroup** with four topics that overlap at high-level concepts: (a) gray scale image of similarity matrix $\mathbf{S}$; (b) the topic of each cluster.

is in the area of $\mathbf{S}_{jj}$ ($1 \leq j \leq k$), then all s_{li} ($l \in \mathbf{S}_{jj}$) are signals that contribute to the structure $\mathbf{S}_{jj}$ while all s_{li} ($l \notin \mathbf{S}_{jj}$) are noise that interfere with the structure $\mathbf{S}_{jj}$. We denote the signal level of $\vec{s}_i$ as $\text{sig}(\vec{s}_i)$ and the noise of $\vec{s}_i$ as $\text{noi}(\vec{s}_i)$ which are respectively defined in Eq.(Eq. 3.13) and therefore, $\|\vec{s}_i\|_p = (\text{sig}^p(\vec{s}_i) + \text{noi}^p(\vec{s}_i))^{\frac{1}{p}}$ and typically $\text{sig}(\vec{s}_i) \gg \text{noi}(\vec{s}_i)$.

$$\text{sig}(\vec{s}_i) = \left(\sum_{l \in \mathbf{S}_{jj}} s_{li}^p \right)^{\frac{1}{p}}, \quad \text{noi}(\vec{s}_i) = \left(\sum_{l \notin \mathbf{S}_{jj}} s_{li}^p \right)^{\frac{1}{p}} \quad (\text{Eq. 3.13})$$

Lemma 3.2 (Properties of $\text{sig}(\vec{s}_i)$) Let n_j be the size of signals (i.e., the number of columns of $\mathbf{S}_{jj}$) of $\mathbf{S}_{jj}$, we define $\overline{\text{sig}(\vec{s}_i)} = \frac{1}{n_j} \sum_{l \in \mathbf{S}_{jj}} s_{li}$ as the average of signals in $\vec{s}_i$, and $\max(\text{sig}(\vec{s}_i)) = \max_{l \in \mathbf{S}_{jj}} \{s_{li}\}$ as the maximum of signals in $\vec{s}_i$. We have the following properties: (i) $\|\vec{s}_i\|_p \approx \text{sig}(\vec{s}_i)$; (ii) when $p \rightarrow 1$, $\|\vec{s}_i\|_p \rightarrow \overline{\text{sig}(\vec{s}_i)} n_j$; (iii) when $p \rightarrow \infty$, $\|\vec{s}_i\|_p \rightarrow \max(\text{sig}(\vec{s}_i))$.

Proof: According to the properties of p -norm of vectors, and because $\|\vec{s}_i\|_p = (\text{sig}^p(\vec{s}_i) + \text{noi}^p(\vec{s}_i))^{\frac{1}{p}}$ and $\text{sig}(\vec{s}_i) \gg \text{noi}(\vec{s}_i)$, when $p \rightarrow \infty$ then $\|\vec{s}_i\|_p \rightarrow \text{sig}(\vec{s}_i)$ and $\text{sig}(\vec{s}_i) \rightarrow \max(\text{sig}(\vec{s}_i))$. Therefore, we get the above conclusions.

Notice that the signal level of the i -th column vector actually describes the cluster $\mathbf{S}_{jj}$ where $\vec{s}_i$ resides. This conclusion can be drawn by the observation where nearly all column vectors in the same cluster exhibit a similar signal level. Hence, extending Definition 3.2 gives us the signal level of a cluster.

Definition 3.3 (Signal level of a cluster) Given a similarity matrix $\mathbf{S}$ with a structure similar to Eq.(Eq. 3.2), the signal level of a cluster $\mathbf{S}_{jj}$ ($1 \leq j \leq k$) is given as

$$\text{sig}(\mathbf{S}_{jj}) = \frac{1}{n_j} \sum_{i \in \mathbf{S}_{jj}} \text{sig}(\vec{s}_i) \quad (\text{Eq. 3.14})$$

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.5: The color images of $L_p(\mathbf{S})$ and the α sequences (see Theorem 3.4, $\alpha_i = \frac{\lambda_{i+1}}{\lambda_2}$) induced from their p -spectrum respectively.

The definition of $\text{sig}(\mathbf{S}_{jj})$ is straightforward. We simply take the mean of the signal levels of all the column vectors in $\mathbf{S}_{jj}$. This is possible because the signal level of a cluster can be approximated by its column vectors. Based on this concept, we derive the following lemma that tells us how L_p normalization makes $\mathbf{S}$ meaningful in subsequent spectrum analysis.

Theorem 3.5 (Differential level of L_p normalization) *Let a and c be entries of $\mathbf{S}$ with structure similar to Eq.(Eq. 3.2). We let a and c represent two possible cases – whether the entry is a diagonal or non-diagonal sub-matrix. Further, suppose a is in $\mathbf{S}_{11}$ and c is in $\mathbf{S}_{12}$. Without loss of generality, we permute $\mathbf{S}$ to make the sub-matrices containing a and c in the place of $\mathbf{S}_{11}$ and $\mathbf{S}_{12}$. Then, we select one of the entries in $\mathbf{S}_{22}$ which we denote as b . After p -normalization of $\mathbf{S}$, a' , b' and c' in $L_p(\mathbf{S})$ corresponds to a , b and c in $\mathbf{S}$ respectively. We therefore have the following conclusions: (i) if $\text{sig}(\mathbf{S}_{11}) \approx \text{sig}(\mathbf{S}_{22})$, then $a' : b' : c' \approx a : b : c$; (ii) if $\text{sig}(\mathbf{S}_{11}) \ll \text{sig}(\mathbf{S}_{22})$, then $\frac{a'}{b'} \gg \frac{a}{b}$, $\frac{c'}{a'} \ll \frac{c}{a}$ and $\frac{c'}{b'} \gg \frac{c}{b}$.*

Proof: According to Definition 3.1,

$$a' = \frac{a}{\sqrt{\|\vec{s}_{a1}\|_p} \sqrt{\|\vec{s}_{a2}\|_p}}, b' = \frac{b}{\sqrt{\|\vec{s}_{b1}\|_p} \sqrt{\|\vec{s}_{b2}\|_p}}, c' = \frac{c}{\sqrt{\|\vec{s}_{c1}\|_p} \sqrt{\|\vec{s}_{c2}\|_p}} \quad (\text{Eq. 3.15})$$

where $\vec{s}_{a1}$, $\vec{s}_{b1}$ and $\vec{s}_{c1}$ represent the column vectors containing the entries a , b and c respectively; likewise for representation for column vectors $\vec{s}_{a2}$, $\vec{s}_{b2}$ and $\vec{s}_{c2}$. By Lemma 3.2(1) and the fact that we can approximate $\text{sig}(\vec{s}_i)$ by $\text{sig}(\mathbf{S}_{jj})$, we have

$$a' = \frac{a}{\sqrt{\text{sig}(\mathbf{S}_{11})}\sqrt{\text{sig}(\mathbf{S}_{11})}}, b' = \frac{b}{\sqrt{\text{sig}(\mathbf{S}_{22})}\sqrt{\text{sig}(\mathbf{S}_{22})}}, c' = \frac{c}{\sqrt{\text{sig}(\mathbf{S}_{11})}\sqrt{\text{sig}(\mathbf{S}_{22})}}$$

then

$$\frac{a'}{b'} = \frac{a}{b} \cdot \frac{\text{sig}(\mathbf{S}_{22})}{\text{sig}(\mathbf{S}_{11})}, \frac{c'}{b'} = \frac{c}{b} \sqrt{\frac{\text{sig}(\mathbf{S}_{22})}{\text{sig}(\mathbf{S}_{11})}}, \frac{c'}{a'} = \frac{c}{a} \sqrt{\frac{\text{sig}(\mathbf{S}_{11})}{\text{sig}(\mathbf{S}_{22})}} \quad (\text{Eq. 3.16})$$

and hence, we can easily get conclusions when $\text{sig}(\mathbf{S}_{11}) \approx \text{sig}(\mathbf{S}_{22})$ and $\text{sig}(\mathbf{S}_{11}) \ll \text{sig}(\mathbf{S}_{22})$.

Corollary 3.1 (Differential level of L_1) *In the same context as Theorem 3.5, when $p \rightarrow 1$: (i) if $n_1 \approx n_2$, then $a' : b' : c' \approx a : b : c$; (ii) if $n_1 \ll n_2$, then $\frac{a'}{b'} \gg \frac{a}{b}$, $\frac{c'}{a'} \ll \frac{c}{a}$ and $\frac{c'}{b'} \gg \frac{c}{b}$; where n_1 and n_2 are the size of $\mathbf{S}_{11}$ and $\mathbf{S}_{22}$ respectively.*

Corollary 3.2 (Differential level of L_∞) *Again from Theorem 3.5, we have the following when $p \rightarrow \infty$: (i) if $\overline{\text{max}(\text{sig}(\mathbf{S}_{11}))} \approx \overline{\text{max}(\text{sig}(\mathbf{S}_{22}))}$, then $a' : b' : c' \approx a : b : c$; (ii) if $\overline{\text{max}(\text{sig}(\mathbf{S}_{11}))} \ll \overline{\text{max}(\text{sig}(\mathbf{S}_{22}))}$, then $\frac{a'}{b'} \gg \frac{a}{b}$, $\frac{c'}{a'} \ll \frac{c}{a}$ and $\frac{c'}{b'} \gg \frac{c}{b}$; where $\overline{\text{max}(\text{sig}(\mathbf{S}_{11}))} = \frac{1}{n_1} \sum_{i \in \mathbf{S}_{11}} \text{max}(\text{sig}(\vec{s}_i))$ is the average of all maximums of column vectors in $\mathbf{S}_{11}$, and likewise for $\overline{\text{max}(\text{sig}(\mathbf{S}_{22}))}$.*

From Corollary 3.1 and 3.2, we can now quantify the degree of overlapping in the clusters by applying the L_1 and L_∞ operations. To better understand this, we revisit our example in Figure 3.4. Let us first look at the results of $L_1(\mathbf{S})$ shown in Figure 3.5(a). Here, we have opted to display the figures in color to better illustrate the differential levels between clusters. If we observe $\mathbf{S}$ in Figure 3.4, the last 3 topics (i.e., T_2 , T_3 and T_4) are highly overlapping and being rather disjoint against T_1 . Therefore, it is possible to view them as 1 unique cluster T_{234} . This point can also be observed by the differentiation level between T_1 and T_{234} in $L_1(\mathbf{S})$, which reflects the differentiation property of L_1 . Since the size of T_1 is much less than T_{234} , we have Point 2 of Corollary 3.1 where nearly all entries of T_1 (which are close to 1) in $L_1(\mathbf{S})$ are greater than those of T_{234} – Theorem 3.1. This implies that the differentiation level of T_1 and T_{234} (shown by their color in Figure 3.5(a)) has “dramatically” changed leading to a signal for T_1 being greater than T_{234} after applying the L_1 operation to $\mathbf{S}$.

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

To see the differentiation level between T_2 , T_3 and T_4 , we recursively apply Corollary 3.1 to T_{234} . Since they have similar sizes, they satisfy Point 1 of Corollary 3.1. Thus, all entries in and between T_2 , T_3 and T_4 of $L_1(\mathbf{S})$ maintain a similar level of differentiation as they are in $\mathbf{S}$. When rendering the differentiation levels, this small level of differentiation results in a similar color tone for most entries in T_{234} as Figure 3.5(a) depicts along with its α value curve of $L_1(\mathbf{S})$, which indicates 2 clusters. Therefore, the L_1 operation magnifies the differentiation levels between disjoint clusters, while “downplaying” those clusters that overlap by obscuring their signal and noise levels, as the total size of overlapping clusters is often larger than the size of single disjoint cluster. Therefore, the 1-spectrum has the ability to find the number of clusters that the user can use to have an insight on the number of disjoint clusters in the data. In this example, examining the collections in T_2 , T_3 and T_4 will reveal that they belong to the newsgroup “comp” which shares many of the keywords that are not in T_1 . Depending on the purpose of analysis, $k = 2$ can be the “right answer”.

So that the users can count the number of overlapping clusters overlooked by the 1-spectrum, we have the p -spectrum ($p \gg 1$). Using the same example but with the p -normalization ($p > 1$), we will show how Corollary 3.2 works. The color images of $L_5(\mathbf{S})$ and $L_\infty(\mathbf{S})$ are shown in Figure 3.5(b) and (c). Recall that in L_p operation, the cluster signal is determined by the average of all maximums of its column vectors. Therefore, $L_\infty(\mathbf{S})$ reveals the differentiation level between clusters by the maximum entry of each cluster. In this way, we can achieve a higher differentiation level to reveal smaller but conceptually more cohesive clusters. As Figure 3.5(c) shows, the differentiation level of T_2 , T_3 and T_4 under $L_\infty(\mathbf{S})$ is more distinctive than under $L_1(\mathbf{S})$ in Figure 3.5(a).

For completeness, Figure 3.5(b) shows the effect of applying a $L_5(\mathbf{S})$ on the same data. The result of $L_5(\mathbf{S})$ gives a higher differentiation level to T_2 , T_3 and T_4 as compared to $L_1(\mathbf{S})$ but less than that of a $L_\infty(\mathbf{S})$ operation. This gradual change from $p = 1$ to $p = \infty$ validates the effectiveness of 1-spectrum and ∞ -spectrum for our purpose of estimating k under different analysis context. We will report more empirical results in Section 3.6 to support this argument.

3.4 A Motivating Example

In this section, we discuss an example of how the theoretical observations discussed earlier work to close the gap between the user and the machinery. For illustration, we assume that the user is given some unknown collection.

If a user does not have pre-existing knowledge of the data, there is a likelihood of not knowing where to start. In particular, all clustering algorithms directly or indirectly require the parameter k . Without spectrum analysis, the user is either left guessing what value of k to start with; or expend time and effort to find k using one of the existing estimation algorithm. In the case of the latter, the user has to be careful in selecting a clustering algorithm and in setting k_{max} (see Sub-Section 3.2.3) – if it is set too high, the estimation algorithm takes a long time to complete; if it is set too low, the user risks missing the actual value of k .

In contrast, our proposal allows the user to obtain an accurate value of k without setting k_{max} . And in situations where the data is sufficiently complex, our proposal can also suggest alternate values of k depending on the user’s judgement and the data in question – if the data has some highly overlapping clusters, our proposal can provide a macro-view and micro-view of the data by the 1-spectrum and ∞ -spectrum. In either case, the performance of our approach is almost instantaneous when compared to methods that require a clustering algorithm. Once an initial value of k is decided, the user can commence clustering. Unfortunately, this is not the end of clustering in real-life.

Upon obtaining the outcome, the user usually faces another question: *What is the quality of this clustering?* In our opinion, there is no knowledge discovery when there is no means to judge the outcome. As a result, it is also at this stage where interactivity becomes important. On this issue, some works propose the use of constraints. However, it is difficult to formulate an effective constraint if the answer to the above is unknown. This is where spectrum analysis plays a part. By Theorem 3.3, the user is given feedback about the cluster quality. At the same time, grey images (e.g., Figure 3.2(b)) can also be constructed to help the user gauge the outcome.

Depending on the feedback, the user may then wish to adjust k , or use another similarity measure. In either case, the user is likely to make a better decision with this assistance. Once the new parameters are decided, another run of the clustering

Algorithm 1 (DkA): Detects the number of distinct large eigenvalues

Input : $\lambda_2, \dots, \lambda_{l+1}$, the sequence with ℓ elements, i.e., the p -spectrum
Output : k , the number of distinct largest eigenvalues

```

1: generate  $\alpha$  sequence  $\alpha_i = \frac{\lambda_{i+1}}{\lambda_i}$  where  $(1 \leq i \leq \ell)$ 
2: select  $\alpha_{\mathcal{O}} = \alpha_1$ ,  $\alpha_{\mathcal{N}} = \alpha_l$  from sequence  $\alpha$  to represent the outliers and normal sets respectively
3: repeat
4:   for all remaining  $\alpha_i$  do
5:     assign  $\alpha_i$  to the nearest representative, i.e., make  $\alpha_i \in \mathcal{O}$  or  $\alpha_i \in \mathcal{N}$ 
6:   end for
7:   select a non-representative  $\alpha_{nr}$  at random
8:   compute the total costs  $S_{\mathcal{O}}$  of swapping  $\alpha_{\mathcal{O}}$  with  $\alpha_{nr}$  and  $S_{\mathcal{N}}$  of swapping  $\alpha_{\mathcal{N}}$  with  $\alpha_{nr}$ 
9:   if  $\min(S_{\mathcal{O}}, S_{\mathcal{N}}) < 0$  then
10:    swap the representative whose  $S = \min(S_{\mathcal{O}}, S_{\mathcal{N}})$  with  $\alpha_{nr}$  to form the new  $\alpha_{\mathcal{O}}$  or  $\alpha_{\mathcal{N}}$ 
11:   end if
12: until no further changes in  $\alpha_{\mathcal{O}}$  and  $\alpha_{\mathcal{N}}$ 
13: return  $k = |\mathcal{O}| + 1$ , where  $|\mathcal{O}|$  is the number of elements in the set  $\mathcal{O}$ 

```

algorithm begins. Our proposal would then kick in at the end of each run to provide the necessary feedback to the user via Theorem 3.3. This interaction exists because different clustering objectives can be formulated on the same data set. At some point, the user may group overlapping concepts in one class. Other times, the user may prefer to separate them. In this aspect, our approach is non-intrusive and works in tandem with the user’s intentions.

3.5 Automatic Detection of k

Until now, we have shown how to determine k by counting the number of large eigenvalues after applying $L_p(\mathbf{S})$. In practice, our approach should operate automatically in the background so that the user can focus on the actual analysis. Therefore, it is important that our proposed method can be automated such that k is provided to the user in a non-intrusive fashion. Fundamental to this is an algorithm that can decide the value of k once the analysis is carried out.

Recall from Theorem 3.4, the first k largest α values (or the first $k+1$ largest eigenvalues) have two properties. First, they form a large gap from the rest of the eigenvalues which are almost equal in their magnitude. Second, the number of large eigenvalues is always smaller than the number of remaining small eigenvalues since it is practically possible to assume that the number of clusters is always less than the number of distinct data points. From these two properties, we can actually perceive the first k large α values as “outliers” from the rest of the α values in the p -spectrum sequence. Then, a simple strategy can be done to identify these “outliers”. We first choose two representatives: $\alpha_{\mathcal{O}}$ to represent the “outliers” and $\alpha_{\mathcal{N}}$ to represent the remaining eigenvalues. We then assign the remaining α values to either $\alpha_{\mathcal{O}}$ or $\alpha_{\mathcal{N}}$

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.6: The color Images of $L_p(\mathbf{S})$ and the α sequences (refer to Theorem 3.4, $\alpha_i = \frac{\lambda_{i+1}}{\lambda_2}$) induced from their p -spectrum, respectively. The left depicts the color image and α sequences of $L_1(\mathbf{S})$, and the right depicts the color image and α sequences of $L_\infty(\mathbf{S})$: (a) `tr12`; (b) `re0`; (c) `hitech`; (d) `mm`.

depending on the α value's distance to a representative. We repeat this process until $\alpha_{\mathcal{O}}$ and $\alpha_{\mathcal{N}}$ reaches equilibrium, as shown in Algorithm 1.

In practice, this algorithm is efficient because ℓ is far less than the number of rows or columns of the matrix n . This is because most α values in the tail of the descending spectrum curve are redundant to the detection of k . Hence, the number of clusters would be far less than the number of the data points and hence, setting ℓ as 100 would be sufficient for detection in most cases.

3.6 Additional Empirical Results

So far, we have provided some initial empirical results to support our proposed method. In this section, we report further results on more data sets from different sources. These data sets are more complex and varied in domain to demonstrate

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Table 3.4: The text collections, gene expression and spatial data sets used to compute the p -spectrum.

Name	Text source	#Classes	# Doc.
tr12	TREC	8	313
re0	Reuters	13	1504
hitech	TREC	6	2301
mm	CLUTO	2	2521

Name	Gene source	#Genes	#Cond.	#Classes
Leukemia	Study of cancer [ea99]	7129	72	2
Yeast	Budding yeast [ESBB98]	64	79	4
Serum	Study of serum [SAS03]	517	12	6
NCI60	National Cancer Institute [ea00]	1161	60	4
Monocytes	Peripheral blood monocytes [HSL ⁺ 00]	2329	139	18

Name	Data source	# Points	# Dimension	# Classes
Simulation	Simulation	1500	2	5
DS4	Widely used in spatial data clustering [KHK99]	8000	2	8

(a) text data sets

(b) gene expression data sets

(c) spatial data sets

Table 3.5: The same experiment conducted in Table 3.3 on additional data sets, indexes and algorithms. The additional indexes are Davies-Bouldin (DB) index [Gor99], SD index [HBV01] and S_Dbw index [HBV01], and the additional algorithm is gap statistic (GapS) [TWH00]. The subscripts of each data set indicates the possible class numbers, e.g., **mm** can have 2, 4 or 5 clusters. And the number underlined indicates a correct estimation.

	Bisecting k -means							Graph-based							Hierarchical						
	CH	KL	Hart	DB	GapS	SD	S_Dbw	CH	KL	Hart	DB	GapS	SD	S_Dbw	CH	KL	Hart	DB	GapS	SD	S_Dbw
tr12 ₈	5	28	1	31	6	20	4	10	23	2	31	7	11	2	4	28	1	26	4	2	13
re0 _{5,6,13}	2	<u>13</u>	8	3	16	7	14	23	11	2	3	<u>13</u>	7	3	<u>6</u>	18	2	3	2	2	2
hitech _{2,3,6}	<u>2</u>	23	22	9	14	<u>2</u>	<u>2</u>	<u>2</u>	18	12	30	11	21	7	<u>3</u>	28	1	11	<u>3</u>	<u>2</u>	<u>2</u>
mm _{2,4,5}	<u>5</u>	20	19	<u>4</u>	15	7	7	6	28	10	21	11	6	20	3	25	1	3	3	<u>4</u>	<u>4</u>

the effectiveness of our approach. Their characteristics are summarized in Table 3.4. In this section, we continued to use the gray-scale or color image of $\mathbf{S}$ or $L_p(\mathbf{S})$ re-ordered by the true cluster labels to verify our results. Also the similarity measures used in this section is cosine function for text data and shall be introduced in detail for gene expression data respectively. Particularly, preprocessing of gene expression data shall be elucidated as well.

3.6.1 Text Data

Our first test is to use the text collections (listed in Table 3.4) generated by the CLUTO toolkit [Kar02]. This text collection is more complex than the one used in Table 3.2 in terms of the cluster size and the hierarchical structures embedded among clusters. Furthermore, this collection is interesting in the sense that the number of classes provided is actually different from the natural and actual number of clusters – a reflection of the subjectivity of k in the real-world.

Figure 3.6(a_1) and (a_∞) shows the spectrum analysis on the **tr12** collection. For this collection, there are 8 predefined classes and that tallies with the results of our analysis using both L_1 and L_∞ operations (note that the 8th class at the lower right

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

of the image is very small). In this case, there is no ambiguity in the number of clusters since there is no obvious hierarchical structure present in the image of $\mathbf{S}$, despite some noise being detected. Therefore, we can conclude that **tr12** is likely to be conceptually cohesive even without examining the documents. For situations like this, the analysis would be relatively straightforward.

The **re0** collection is more interesting than **tr12**. Although **re0** has 13 predefined classes, most of the clusters are small with some having less than 20 documents while a few classes (*money*, *trade* and *interest*) made up 76.2% of documents in **re0**, i.e., the remaining 10 classes contain 23.8% of the documents. This is made more challenging when the 10 classes are highly related. Hence, our analysis only shows 2 clusters for these 10 classes giving us a total of 5 clusters for the similarity matrix used. Furthermore, our analysis also reveals no obvious hierarchical structures in the collection. Unlike our study of **tr12**, **re0** goes on to show the possibility where the analysis may differ from the predefined class labels. Of course, the crucial point here is to appreciate that both are valid because of different analysis contexts, and that this should only serve as a reference to the actual analysis that the user is conducting. Figure 3.6(b_1) and 3.6(b_∞) shows the image and α sequences of the 1-spectrum and ∞ -spectrum in which we derive the above conclusions.

There are 6 predefined classes in the **hitech** collection namely, *computer*, *electronics*, *medical*, *health*, *research*, and *technology*. The results of the L_1 operation show 3 clusters as shown in Figure 3.6(c_1) while in Figure 3.6(c_∞), we have the results of the L_∞ spectrum showing the second clusters to contain 3 highly correlated topics, i.e., *computer*, *electronics* and *technology*. For simplicity, we collectively refer these 3 topics as “cet” and can, as a matter of fact, be viewed as a single cluster. Likewise, topics like *health* and *medical* are also highly correlated and therefore, can be viewed as another cluster which we shall refer to as “mh”. Thus, if we now look at the images in Figure 3.6(c_∞), we can see a clear hierarchical structure consisting of the class “mh” and *research*. If we look at them closely, we can conclude that they are 2 distinct clusters when taking the 1-normalization on $\mathbf{S}$, and when ∞ -normalization on $\mathbf{S}$ is applied, $L_\infty(\mathbf{S})$ concludes a single cluster. Thus, our method signals the existence of a clear hierarchical structure within this collection. Given that the other classes do not exhibit an obvious hierarchical structure, the possible number of clusters can be

2 or 3 depending on the analysis requirements. Again, this result differs from that of what was given in the predefined class labels.

The last text collection (`mm`) that we looked at contains 2 predefined classes: *music* and *movie*. The interesting result here is that we obtained 4 distinct clusters from our analysis instead of the predefined 2. When we did further checks, we found a clear hierarchical structure embedded in the image of **S** (enclosed in white squares in the figure). In this case, we demonstrate how our method can be used to reveal a hierarchical structure that is otherwise “hidden” by predefined classes given under a different context.

We shall check the accuracy of our method similar to what we did in Section 3.2.3. From Table 3.5, we see that although `tr12` is a small data set containing only 313 documents. None of the indexes tested can correctly estimate the cluster number. We attribute the rationale of this behavior to the high level of noise among clusters as we can see in Figure 3.6. On the other data sets, some of the indexes perform well in their estimation only if the right algorithm is used in conjunction. In all cases, none of the indexes worked correctly with all the algorithms on all the data sets. The best is the CH index on `hitech`, which managed to estimate k to be either 2 or 3, and SD and S_Dbw indexes on `hitech` and `mm` by hierarchical clustering algorithm which give the results that agree with our method and analysis. Interestingly, the CH index estimates $k = 3$ when the hierarchical clustering algorithm was used and $k = 2$ on the other partition-based algorithms. This reinforces our earlier observation of a hierarchical structure in `hitech` that our method reveals.

3.6.2 Gene Expression Data

In this set of experiments, we selected 5 gene expression data sets to demonstrate the effectiveness of our approach. For ease of comparison, we preprocess all the data sets by standardizing the object vectors with a mean of 0 and a standard deviation of 1. We also used similarity measures to compute the similarities between the genes or conditions. In most cases, we found that the Pearson correlation coefficient is sufficient to develop the similarity matrix for effective clustering. If higher accuracy is desired, our experiments show that an extension of the Euclidean distance² can be

² $\exp(-\frac{\|\vec{x}-\vec{y}\|^2}{\sigma})$, where σ is a positive number.

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.7: The **Leukemia** data set analyzed with the Pearson correlation coefficient: its α sequence of 1-spectrum and ∞ -spectrum.

Figure 3.8: The **budding yeast** gene expression data set: (a) α sequence of 1-spectrum; (b) α sequence of ∞ -spectrum; (c) the grey scale plot where (1) is protein degradation, (2) is chromatin structure, (3) is protein synthesis and (4) is glycolysis.

used as the similarity measure. Of course, this conclusion comes easily since we can substitute different similarity measures within the spectrum analysis.

Leukemia Data

The Leukemia data set [ea99] comes from a study of gene expression of two types of acute Leukemias: Acute Lymphoblastic Leukemia (ALL) and Acute Myeloid Leukemia (AML). The data comprises 47 samples of ALL (of which 38 are B-Cell ALL and 9 are T-Cell ALL) and 25 samples of AML. From the above, we expect a good clustering algorithm to output 2 (ALL, AML) or 3 (ALL B-Cell, ALL T-Cell, AML) clusters. we followed the experimental setup and preprocessing in [ea99]. The results, shown in Figure 3.7, give the correct answer that we expected. Meanwhile, the large gap between the first and second α values in Figure 3.7(b) suggests that it is also possible to have 2 clusters.

Budding Yeast Data

This data set is obtained from a study of gene expression in the *Saccharomyces cerevisiae* budding yeast during a diauxic shift [ESBB98] where each gene has 79 arrays.

Figure 3.9: The **serum** gene expression data set analyzed using different similarity measures: (a) Pearson correlation coefficient; (b) extension of Euclidean distance with $\sigma = 3$.

For our experiment, we selected 4 clusters (64 genes in total) sharing similar expression patterns, and are annotated along the same biological pathway. The 4 clusters are *protein degradation* (cluster C), *chromatin structure* (cluster H), *protein synthesis* (cluster F), and *glycolysis* (cluster E). For simplicity of experimental setup, but without loss of accuracy, we do not impute the cells without data in the gene expression matrix with zeros. Again, we used the Pearson correlation coefficient as the measure of similarity. From these preprocessing, we obtained the corresponding spectrum in Figure 3.8. By Theorem 3.4, the spectrum graph suggests $k = 4$ in this particular data set, which corresponds to our experimental setup. At the same time, the image in Figure 3.8(b) suggests that it is also possible to have 2 clusters, i.e., $k = 2$. This possibility is correct since we specially selected the 4 clusters such that it is possible to put *protein synthesis* and *glycolysis* in one class and the other 2 into another.

Serum Data

This data set comprises the transcriptional response of human fibroblasts to serum [SAS03]. We selected 517 genes whose expressions vary in response to serum concentration in human fibroblasts as did in [SAS03]. Sharan et al. (2003) showed that this data can be well described with 6 clusters [SAS03]. There are no missing values and therefore,

Figure 3.10: The **NCI60** data set: (a) the grey scale plot at <http://genome-www.stanford.edu/nci60/figures.shtml>; (b) α sequence of 1-spectrum; and (c) α sequence of ∞ -spectrum.

we simply standardize the mean and deviation. The similarity matrix is built using the Pearson correlation coefficient, and the results are shown in Figure 3.9(a).

Interestingly, the analysis reveals only 3 clusters, and there is no obvious hierarchical structure present. Given that there should be 6 clusters, we suspect that the Pearson correlation coefficient may be the cause. We therefore replaced it with the extension of the Euclidean distance. Performing the analysis with this measure, we have the results in Figure 3.9(b). This time round, we see that the number of clusters reported with 1-spectrum is 5, and the ∞ -spectrum, 6, suggesting the presence of a hierarchical structure. This experiment therefore conveyed an important point: without sufficient domain knowledge, the user may have to use a number of similarity measure to compare the results before furthering other cluster analysis, or conclusions.

NCI60 Data

The cDNA microarrays was used to examine the variation in gene expression among the 64 cell lines from the National Cancer Institute's (NCI) anti-cancer drug screen [ea00]. This time, we followed experiment setup in [ea00] to select a subset of 1,161 cDNAs measured across 64 cell lines. For convenience, we reproduced the part of this information in Figure 3.10. As with the previous tests, we performed the same standardization and used the Pearson correlation coefficient as the similarity measure. The 1 and ∞ -spectrum of this data set are shown in Figure 3.10(b) and (c). Once again, the estimation from our analysis agrees with that given by Figure 3.10(a) and its discussion in the original paper.

Peripheral Blood Monocytes Data

Our last data set contains the hybridization fingerprints of 2329 cDNAs obtained from 139 oligonucleotide probes [HSL⁺00]. The true clustering of these 2329 cDNAs is known from back hybridization experiments. – it contains 18 gene clusters varying in sizes from 709 to 1 as shown in [HSL⁺00]³. Due to its wide variation of the cluster sizes and many singletons (data that do not belong to any compact clusters), this data set has more complex clustering structure than all of the previous gene expression data sets. As a matter of fact, clusters with too small sizes (e.g. 1 or 2) do not really constitute to a cluster per se. The analysis in [HSL⁺00] concludes the first 8 small clusters (with sizes from 1 to 43) as invalid clusters. Therefore, the actual possible cluster numbers range between 10 and 16.

In this experiment, we first used the Pearson correlation coefficient to compute a similarity matrix from the standardized input matrix (for cDNAs). And we conducted the analysis using this matrix to obtain the results in Figure 3.11(a). However, this experiment was made in the case we do not have the domain knowledge. Next, we considered the domain expertise from [HSL⁺00] and followed its preprocessing and construction method of similarity matrix to get the result in Figure 3.11(b). Comparing these two figures, it implies that the similarity matrix encoding domain knowledge is likely to give the best result.

From Figure 3.11(a), we see that the Pearson correlation coefficient generates a high level of noise. Nevertheless, we can still detect 9 distinct clusters using our analysis, where the last two largest true clusters are blurred and can only be viewed as one class. The optimality of our approach is demonstrated when we incorporate multiple similarity measures in our analysis as shown in Figure 3.11(a) and (b). In this case, the value of k estimated ranges from 10 to 12 which agrees with our analysis above. Some readers may find that the largest α values are not obvious in Figure 3.11. If we inspect the image rendered, we can see that some of the clusters are not “well-formed” and that leads to obscured α sequences. Even in this situation, our proposed detection algorithm manages to detect the largest α sequences correctly.

³The data set and the class labels for each cDNA are available at <http://www.cs.tau.ac.il/~rshamir/expander>

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

Figure 3.11: The **monocytes** cDNA data set analyzed with different similarity measures. For easy verification, two similarity matrices are reordered by the cluster labels provided in [HSL⁺00] and with the increasing cluster sizes: (a) Pearson correlation coefficient resulting in 8 distinct α sequences in the 1-spectrum and ∞ -spectrum respectively; (b) Processing techniques described in [HSL⁺00] followed resulting in 9 distinct α sequences in the 1-spectrum and 10 distinct α sequences in the ∞ -spectrum.

Comparison with Other Methods

We conclude our experiments by showing the effectiveness of our approach on the data sets in Table 3.4 as what we did in Section 3.2.3 and Section 3.6.1. We have also increased the number of validation methods to 10 (added Dunn index [Gor99] and Partition Entropy Coefficient (PE) index [HBV01], CLICK [SAS03] MCLUST [FR98], Minimum Entropy Clustering (MEC) [LZJ04]). The results are shown in Table 3.6. Among these methods, MEC and PE are based on information theory.

In this table, all data sets were preprocessed according to the papers studying them. Not surprising, most indexes managed to successfully estimate the number of clusters in the Leukemia and Budding Yeast data sets because of the small number of data points (72 and 64 respectively), and their low dimensionality (for Leukemia data set, only small number of gene features were selected according to preprocessing in [ea99]). Yet, there are also indexes that failed to give a reasonable estimate, e.g., the KL index. Worse, all the indexes failed to give a correct estimate when we scaled out test to larger and higher dimensionality data sets. For example, none of the indexes managed to give a correct value of k for the Serum and NCI60. For the Monocytes

Table 3.6: The same experiment conducted in Table 3.3 and Table 3.5 on additional data sets, indexes and algorithms. The additional indexes are the Dunn index and PE index (based on membership matrix of objects computed by Fuzzy k-means clustering algorithm), and the additional algorithms are CLICK, MEC, and MCLUST with k -means (km) and hierarchical (hi). In CLICK, the number in the bracket represents the unclustered singletons. And the number underlined indicates a correct estimation.

	CH		KL		Hart		DB		Dunn		GapS		CLICK	MEC	PE	MCLUST
	km	hi	km	hi	km	hi	km	hi	km	hi	km	hi				
Leukemia _{2,3}	<u>2</u>	<u>2</u>	27	25	<u>2</u>	<u>2</u>	<u>3</u>	7	<u>2</u>	<u>2</u>	<u>2</u>	<u>2</u>	1 (0)	<u>2</u>	<u>2</u>	4
Yeast ₄	<u>4</u>	<u>4</u>	25	1	<u>4</u>	<u>4</u>	6	<u>4</u>	2	2	<u>4</u>	<u>4</u>	3 (3)	3	2	1
Serum ₆	3	3	30	2	10	10	7	8	3	3	7	3	4 (10)	4	3	1
NCI60 ₄	2	2	28	22	7	9	2	2	2	2	11	3	6 (521)	5	2	1
Monocytes ₁₀₋₁₆	3	3	28	18	18	<u>14</u>	7	3	2	2	30	3	<u>16</u> (60)	7	3	1

data set, only the Hart index managed to get the acceptable value of k when the hierarchical algorithm is used.

We also note that some of the indexes are unstable in the sense that there is a large difference in the estimate when used with different algorithms. For example, both KL and DB are highly sensitive in our tests. Our approach being independent of the clustering algorithm, does not fluctuate as much despite being dependent on the preprocessing and similarity measure used. Finally, none of the algorithms in our test are effective in estimating k - CLICK manages only one right estimate in Monocytes while MCLUST failed in all cases.

3.6.3 Spatial Data

The spatial data is different from text data and gene expression in terms of domain knowledge, e.g., the shape of cluster, which is not meaningful in high-dimensional data. Therefore, to incorporate the knowledge of the cluster shape in the similarity measure, we used the Connectivity Kernel (CK) proposed by [FRB03]. As the CK is a distance-based measure, we employed the Gaussian function to transform CK from distance values to similarity values. Given a spatial data set with n points, the CK obtained is denoted as $D = (d_{ij})_{n \times n}$ ($d_{ij} \geq 0$), where d_{ij} is the path-based distance between the i -th and j -th points. We then apply the Gaussian function to each d_{ij} as $s_{ij} = \exp(-d_{ij}/\sigma)$, where σ is a positive scalar number. In this way, a similarity matrix based on CK is constructed $S = (s_{ij})_{n \times n}$. Next we can apply our method to S which encodes the domain knowledge of cluster shapes in the similarity matrix S . We conducted the experiments in both the simulation data and the DS4 data set which is widely used in spatial data clustering.

Figure 3.12: Simulated two-dimensional data set with 1,500 points. Both (b) and (c) revealed that there are $4 + 1 = 5$ distinct clusters in this data set.

Figure 3.13: The data set DS4 with 8,000 points. From (b) and (c), there are clearly $2 + 1 = 3$ obvious clusters; there are $7 + 1 = 8$ clusters that are not obvious and affected by noises, although the 5 small blue α are not distinct enough.

Simulation Data

We begin our test with the circular cluster shapes. A 2-dimensional data set with five clusters of different circular sizes was simulated as shown in Figure 3.12(a). After a run of CK and Gaussian function with $\sigma = 0.2$, we obtained the similarity matrix whose 1-spectrum and ∞ -spectrum are shown in Figure 3.12(b) and (c). The result clearly reflects the correct number of clusters in this data set.

DS4 Data

The DS4 data set is widely used in spatial data clustering [KHK99]. In our next experiment, we repeat the test using the DS4 data set used in [KHK99]. This particular data set has arbitrary cluster shapes and contains much noise as shown in Figure 3.13(a). Clearly, the amount of noise in DS4 poses a challenge for analysis since some of the noise actually connect two clusters together. The CK may not be

robust to such spatial data with high level of noisy points connecting clusters. Thus, it may be expected that using CKs may wrongly identify two close clusters as one in the presence of much noise. The similarity matrix of DS4 was obtained by running CK and Gaussian function with $\sigma = 10$. Its 1-spectrum and ∞ -spectrum are shown in Figure 3.13(b) and (c). This result verifies our surmise. There are only 3 distinct clusters, i.e., there are 2 red large circles in Figure 3.13(b) that was inferred from the DS4's similarity-based connectivity kernel. In Figure 3.13(a), we identified 3 regions. Within each region, there is a higher degree of noise (in Figure 3.13(a), they are labeled by the yellow rectangles) causing clusters within the region to be seen as one by the CK. When observing Figure 3.13(b) and (c) carefully, we find that our spectrum analysis reveals five more α s which corresponds to the five small blue circles in Figure 3.13(b). These 5 small α s show the impact of noise in DS4. So if we count these five α s, The total 8 clusters in DS4 can be estimated.

3.6.4 Performance

In cases where the similarity matrix is dense, the computational and storage cost will be high during similarity computation, transformation, and eigenvalues retrieval. To minimize cost, we adopted a graph sparsification technique proposed in [FK81] to speed up our proposal. The technique works by randomly omitting entries in the symmetric matrix $\mathbf{S}$ to obtain the associated matrix of the sparse graph. It is formally stated below,

$$\hat{\mathbf{S}}_{ij} = \hat{\mathbf{S}}_{ji} = \begin{cases} 0 & \text{with probability } 1 - p \\ \frac{1}{p}\mathbf{S}_{ij} & \text{with probability } p \end{cases} \quad (\text{Eq. 3.17})$$

where the probability p indicates the dense degree of the graph. The effectiveness of graph sparsification is guaranteed by the analysis of the error symmetric matrix $\Delta = \hat{\mathbf{S}} - \mathbf{S}$. Δ whose entries are independent random variables with mean $E(\Delta) = 0$ and variance $(\frac{1}{p} - 1)\mathbf{S}_{ij}^2 \leq (\frac{1}{p} - 1)$. It is easy to show that the error matrix Δ with small matrix norms shall make $\hat{\mathbf{S}}$ maintain almost the same spectrum structure of $\mathbf{S}$. Theorem 3.6 shows the error quantification of sparsification process stated in (Eq. 3.17) and thus guarantees the process not destroying the spectral structure of $\mathbf{S}$. Generally, every matrix with Frobenius norm $\|\cdot\|_F = \left[\sum_{i=1}^n \sum_{j=1}^n s_{ij}^2\right]^{1/2} = \sigma n$ has 2-norms at least $\sigma\sqrt{n}$. Therefore, we see that the 2-norms error introduced by $\hat{\mathbf{S}}$

is within a factor of 4 of the 2-norms error associated with any modification to $\mathbf{S}$ that has the same entry mean squared error.

Theorem 3.6 (Füredi and Komlós [FK81]) *Let Δ be an $n \times n$ symmetric matrix whose entries are independent random variables with mean zero, variance bounded above by σ^2 , and magnitude bounded by $\frac{\sigma\sqrt{n}}{\log^3 n}$. With probability $1 - 2\exp(-\frac{\sigma^2 n}{8})$, we have $|\Delta|_2 \leq 4\sigma\sqrt{n}$, where $|\cdot|_2$ is 2-norms of a matrix that is also known as the spectral norm [GL96].*

In practice, the entries need not be multiplied by $\frac{1}{p}$ and this further improves performance. Therefore, the sparsification process greatly reduces the time and space requirements to store the similarity matrix. Consequently, steps (transformation and computation of extreme eigenvalues) that follow also become efficient. Our experimental results in Section 3.2.3 and 3.6 show the effectiveness of using this technique. In most cases, $p = \frac{\log^2(n)}{n}$ is sufficient especially when n is large ($n > 10,000$). Therefore, during construction of $\hat{\mathbf{S}}$, the space used to store $\hat{\mathbf{S}}$ is $O(pn^2) = O(n\log^2(n))$. If data set is sparse, e.g., text collections, the time complexity is $O(n\log^2(n)\bar{r})$, where $\bar{r}$ is the average number of non-zeros in each object vector of data set. Otherwise, its time complexity is $O(n\log^2(n)d)$, where d is the dimension of data set.

For $\hat{\mathbf{S}}$, the time complexity of performing a p -normalization from $\hat{\mathbf{S}}$ to $L_p(\hat{\mathbf{S}})$ would be $O(n\log^2(n) + n)$. The study of how to effectively compute the largest eigenvalues has been well-developed within the framework of symmetric generalized eigen problems that arise from structural analysis in physics and computational chemistry. The class of **Lanczos** methods is especially efficient both in space and time to compute the extreme eigenvalues of large-scale sparse symmetric matrix. It possesses the remarkable convergence property with $i \ll n$ iterations [GL96]. Since the complexity of each iteration is $O(n\log^2(n) + n)$ and the eigenvalue computation of the final tridiagonal matrix is $O(nk)$, where k is the number of eigenvalues to be computed, the final complexity of this method is therefore $O(i(n\log^2(n) + n)) + O(nk)$. If we ignore the very small i in real computations, the complexity of our method becomes $O(n\log^2(n) + n) + O(nk)$. There are fast **Lanczos**-based packages (e.g., ARPACK [LSY98], LANSO [LAN]) for such computation. If using ARPACK package, additional space storage is $nO(k) + O(k^2)$ [LSY98].

Figure 3.14: Running time comparison of our method with three typical methods in the very large text collection RCV1-v2.

In summary, total time cost when graph sparsification is used (construction of similarity matrix, transformation, and eigenvalues retrieval) is $O(n \log^2(n)\bar{r}) + O(n \log^2(n) + n) + O(nk)$ when data set is sparse, or $O(n \log^2(n)d) + O(n \log^2(n) + n) + O(nk)$ when data set is dense; and the total space cost is $nO(k) + O(k^2) + O(n \log^2(n))$ in memory and $O(n \log^2(n))$ in disk. If data set is small (in the scale of thousands), e.g., biological data, the time and space required is small without needing sparsification.

In practice, we did the empirical experiments on a real-life and massive data set to test the efficiency and scalability of our method. We used the text collection RCV1-v2⁴ with 804,414 documents and 47,236 terms. We selected three typical methods (DB index, S_Dbw index and gap statistic with the very efficient text clustering algorithm bisecting k-means implemented in CLUTO toolkit [Kar02]) to compare with our method. They are implemented in C and run on a computer with Pentium 4 CPU 3.4GHZ, 1.00 GB of RAM and Windows XP. For two methods compared, we set $k_{max}=21$ and thus run the clustering algorithm 20 times as a round. Particularly, we set the number of reference data sets randomly generated as 3 for gap statistic. The result was shown in Figure 3.14. The curve of our method has indicated how the efficiency and scalability of our method is with the size of data set. Meanwhile, the curves of two typical methods compared verified they are tasks with intensive computation.

⁴It is available at http://www.jmlr.org/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm

3.7 Discussion

As we conclude, we would like to discuss a number of issues that may confuse the readers, and to summarize the empirical results and the algorithms we presented so far.

The first issue is with the definition of what constitutes a cluster. Clearly, several works have attempted to give a definition under different problem contexts and their results would vary under different clustering algorithms. For example, some clusters are defined based on their spatial relationships in the Euclidean space, e.g., DBSCAN [Ber02] and OPTICS [Ber02], while others are defined by their co-occurrence property.

We can conclude from these examples that it is principally the domain knowledge that influences the definition of the cluster in the given data set. This is also the case when we looked at how the true number of clusters is determined. According to [JMF99], the clustering process consists of three key steps: pattern representation, inter-pattern similarity, and grouping. In this framework, the domain knowledge, i.e., subjectivity of clusters, is encoded in the first two steps. This idea is similar to the principles of kernel-based learning methods. In fact, our method can be considered kernel-based by viewing the similarity matrices as kernels. Likewise, the true number of clusters can be related by the graph and its similarity matrix (which encodes the domain knowledge), and thus allow the clusters to be discovered.

In exploiting this characteristic, our method becomes generic and independent of the clustering algorithms used. A well-understood example is the application of our method to the spatial data by means of utilizing the connectivity kernel which can effectively capture the clusters of spatially elongated shapes [FRB03] and therefore, estimate the number of such spatial clusters by the connectivity kernel. Therefore, our proposal is efficient and effective in different domains despite differences in their data characteristics. Recently, two clustering validity indexes SD [HBV01] and S_Dbw [HBV01] were proposed but they are valid only within the spatial domain as [HBV01]’s empirical results show, i.e., they are inefficient and ineffective in large high-dimensional data sets.

As our method is developed from spectral graph theory, from the aspect of graph cut, there is a general concern of relationships between our method with other graph

cut models recently proposed, e.g. ratio cut [Alo86], min-max graph cut [DHZ⁺01b] and normalized cut [SM00]. Although these graph cut models are closely related to spectral graph theory, several differences distinguish our method from theirs. Firstly, the 1-spectrum of our method is clearly based on the normalized Laplacian and Cheeger constant which is a min cut of a graph. This is an extensively studied graph cut model in mathematics and has been mathematically proved and studied (please refer to the monograph by Chung [Chu97]). More important, Chung pointed out that the eigenvalues of the normalized Laplacian (i.e., 1-spectrum) relate well to the graph invariants and topology, while eigenvalues of the Laplacian have failed to do. Meanwhile, although other graph cut models are effective in clustering graphs, they have not been extensively studied theoretically and empirically in the application of inferring the cluster number by only investigating their related eigenvalues. This makes 1-spectrum in our method different from others. Secondly, the adoption of 1-spectrum to our method consequently enables us to easily extend 1-spectrum to p -spectrum. This is the benefit of 1-spectrum which other graph cut models may not provide. In this way, p -spectrum together with 1-spectrum provides users more flexible and reasonable options with a range of possible cluster numbers. This is a solution any existing cluster estimation methods can not obtain. However, due to the complexity of clusters with nested and overlapping structures, different graph cut models may generate different clustering results given the same cluster number as input parameter. We realized the complexity of this problem and therefore our research gave an attempt to provide users a more flexible solution (a range of possible cluster number) from the benefit and advantages of kernels and spectral graph either 2, 4, or 5.

All in all, this result is still good with only a small fluctuation on the cluster number in complicated cases after introducing sparsification. While these fluctuations are acceptable, we further analyzed these 3 data sets to find out the cause. We discovered that there were some small clusters in these data sets. And after sparsification of their similarity matrices, those small clusters no longer appear to be “good clusters” while the big clusters were largely unaffected by the process. On the other hand, the random omitting strategy may cause some of the high similarity values to be omitted causing the small clusters to become sparse and therefore, excluded in the sparsified similarity matrix. While both affects the outcome of small clusters in the data set,

Table 3.7: The overall procedure of cluster number estimation

Input:	(1) D , the data set (suppose $\mathcal{O}$ is the set of objects, and $ \mathcal{O} $ is the number of objects); (2) $f : (o_1, o_2) \in \mathcal{O} \times \mathcal{O} \longrightarrow \mathbb{R}$, the similarity measure between any two objects.
Output:	$[k_{\min}, k_{\max}]$, the range of the cluster number in the data set.
1:	if $ \mathcal{D} < n_{threshold}$, compute the pairwise similarity matrix, S .
2:	else compute the sparse pairwise similarity matrix, S , by randomly calculating $p \mathcal{O} ^2$ pairwise similarities.
3:	normalize S with 1-normalization and ∞ -normalization to get $L_1(S)$ and $L_\infty(S)$.
4:	find the largest 100 eigenvalues of $L_1(S)$ and $L_\infty(S)$ respectively, denoted as $\text{eig}(L_1(S))$ and $\text{eig}(L_\infty(S))$.
5:	if users would like to find the first k largest eigenvalues by eyes,
6:	plot the $\text{eig}_{2-100}(L_1(S))$ and $\text{eig}_{2-100}(L_\infty(S))$ and find k_1 and k_∞ by eyes.
7:	else automatically compute $k_1 = \text{DkA}(\text{eig}_{2-100}(L_1(S)))$ and $k_\infty = \text{DkA}(\text{eig}_{2-100}(L_\infty(S)))$
8:	$k_{\min} = \min(k_1, k_\infty)$, and $k_{\max} = \max(k_1, k_\infty)$.
9:	return the range of the cluster number, $[k_{\min}, k_{\max}]$.

the estimation results are fine as we have tested in practice. Even with the loss of small clusters, the ‘big picture’ remains intact. In cases where absolute precision is needed, we can omit sparsification. As shown in Section 4.3.3, the complexity of our method is still much smaller than that of existing methods.

Finally, Table 3.7 gives the overall procedure of cluster number estimation using our method.

3.8 Related Works

Underlined by the fact that there is no clear definition of what is a good clustering [TWH00, KVV00], the problem of estimating the number of clusters in a large data set with high dimensionality is arguably a difficult one. Over the years, several approaches to this problem have been suggested. Among them, the more well-known ones include cross-validation [Smy96], penalized likelihood estimation [HY01], resampling [RLBB02], finding the ‘knee’ of an error curve [TWH00, TWBB01], gap statistics [TWH00], validity indexes [Gor99, HBV01] under the framework of cluster

validation, and criteria such as the Bayesian Information Criterion (BIC) for model selection capability and likelihood-based clustering algorithms, e.g., MCLUST [FR98].

The problem with these techniques is that they either make a strong parametric assumption, or they are computationally expensive. For example, both cross-validation and penalized likelihood estimation require some form of input parameters, e.g., the number of cross validations or the MML. On the other hand, techniques such as resampling and finding the ‘knee’ of an error curve are examples of CPU-intensive methods. In resampling, the number of clusters are discovered by repeated clustering of samples drawn from the original data set; while in the case of finding the ‘knee’ of an error curve, each potential value of k requires a run of the clustering algorithm. In cluster validity indexes and gap statistics, runs of the algorithm on the target data (and in the case of gap statistics, reference data as well) with a range of k are needed making them computationally expensive as our experiments show. Worse, some of them depend on specific clustering algorithms as reported in [HBV01].

Unfortunately, these methods become undesirable when the objective of estimating k is to facilitate productive cluster analysis. What is really needed is a “roadmap” containing the necessary information for the user to perform the knowledge discovery. In our case, this “roadmap” is the result of analyzing the spectrum of the *text* collection. More importantly, this result is easy to obtain as neither are parameters needed (from the user’s perspective, δ is unknown) nor computationally intensive. Furthermore, the spectrum contains other information besides k – it also provides a means to estimate the cluster quality of the data set in question.

3.9 Summary

In this chapter, we demonstrate a concrete case of our argument on the need to close the gap between the user and the available machinery. We support our argument by studying a well-known issue in clustering which is faced by every user during analysis: *What value of k should we select so that analysis converges quickly to the desired outcome?*

We answered this question with spectrum analysis within the context of *text* collections. We show, both argumentatively and empirically, that if we are able to provide a good estimate to the value of k (or values of k if p -norm is used over 1-norm), then

CHAPTER 3. ENHANCING THE EFFECTIVENESS OF CLUSTERING WITH SPECTRUM ANALYSIS

we will have a good starting point for analysis. Once this “ground” is known, data mining can proceed by changing the value of k incrementally from the starting point. This is often better than the trial and error approach. In addition, we also show that our proposal can be used to estimate the quality of clustering. As part of the analytical processing, this is equally important to the success of knowledge discovery. Our proposal contributes in part to this insight.

In general, the results shown here demonstrate the feasibility to study techniques that bridge the user with the available machinery – in this case, the clustering algorithm. We strongly believe that this endeavor will play a pivotal role to the advancement of knowledge discovery. In particular, as data takes a paradigm shift into continuous and unbounded form, users will no longer be able to devote time in tuning parameters. Rather, their time should be spent on interacting with the available machinery such as what we have demonstrated in this chapter.

Chapter 4

Spectral Kernels for Classification

Kernel-based learning first appeared in the form of Support Vector Machines (SVMs), and readily became the state-of-the-art for learning algorithms. The framework of kernel-based learning methods (KM) is also known as kernel-based analysis of data in both supervised and unsupervised learning [CST00, SSM98, BJ03]. Within this framework, kernels encode all the information required by the learning machinery, and act as the interface between the data and the learning modules [STC04]. Hence, these kernels are implicitly high-dimensional spaces that contain more information than the original explicit feature space. The advantage of this is that once the kernel is obtained, kernel algorithms can perform analysis without further information from the original data set.

In text categorization, the kernel was used to capture a semantic network of terms to better compute the similarities between documents [SdB00]. In natural language learning, subparse trees are taken into consideration in the semantic kernel to improve the accuracy of classifying predicative arguments [MB04]. And in image retrieval, the knowledge about users' queries are encoded in the kernel to improve query accuracy [GC04]. While domain knowledge is usually encoded in the kernel by expert users, they can also be obtained from automated discovery algorithms. The pioneering attempt to integrate unsupervised discovery, in the form of kernels, for supervised learning is Latent Semantic Indexing (LSI). It has been shown [CSTL01] that the Latent Semantic Kernel (LSK) benefits from the automated discovery of latent semantics that aid the task of classification. In fact, the semantics uncovered in LSI are simply the 'tip of the iceberg' of spectral graph analysis on kernel matrices. Under spectral graph theory, there has been active research on the use of latent semantics for

clustering. This research, known as *spectral clustering*, is a method that uses spectral information to assist clustering algorithms.

Obtaining the spectral information of a data set is a three step process: (i) compute the similarity matrix $\mathbf{S}$ from the data; (ii) transform $\mathbf{S}$ to another matrix $\Gamma(\mathbf{S})$; and finally (iii) perform an eigen-decomposition on $\Gamma(\mathbf{S})$. Our analysis of this process led to an important discovery — if certain matrix transformation (e.g., normalized Laplacian) is performed, we can observe some interesting latent clustering semantics in the eigenvalues and eigenvectors of $\Gamma(\mathbf{S})$ [Chu97, Din04, SM00, MS01, NJW01] that can be used in the kernel for the task of classification. Our observation, and hence the main contribution of this chapter, led to our proposal of the *Spectral Kernel*. The spectral kernel combines two state-of-the-art learning algorithms: kernel-based learning and spectral clustering; and introduces a mechanism that supports the spectral embedding of new input data into the kernel to improve classification precision.

We present our proposal as follows. The next section provides the background about kernels and spectral clustering. In doing so, we provide the theoretical analysis and examples to demonstrate the steps to compute the spectral embedding space. We then present in Section 4.2, the steps to update the kernel values as new input arrives — a differentiation of our approach from other spectral learning methods. We then provide empirical results in Section 4.3 to support the feasibility of our proposal. Finally, we conclude our work in Section 4.4.

4.1 Spectral Graph Analysis of Kernel Matrices

To facilitate understanding of our proposal, as well as the analysis and proofs presented in the later sections of this chapter, we first introduce some basic facts of kernel matrices and its mathematical foundation [Chu97].

Given a set of data points $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ and a kernel function $\kappa(\cdot, \cdot)$, the kernel matrix $\mathbf{K} = (\mathbf{K}_{ij})_{i,j=1}^n$ is defined as $\mathbf{K}_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{K}$ is symmetric and usually positive semi-definite (see Section 2.1 for more detail). By operating on $\mathbf{K}$, we can easily recode the data in a manner suitable for the learning module. A simple and widely used κ is the inner product $\kappa_I(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$. And if we have $\kappa_1(\mathbf{x}, \mathbf{z})$ as a kernel, we can construct new kernels using other kernel functions, e.g., exponential kernel

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

$\kappa_E(\mathbf{x}, \mathbf{z}) = \exp(\kappa_1(\mathbf{x}, \mathbf{z}))$; polynomial kernel $\kappa_P(\mathbf{x}, \mathbf{z}) = (\kappa_1(\mathbf{x}, \mathbf{z}) + d)^p$ with positive coefficients; and Gaussian kernel $\kappa_G(\mathbf{x}, \mathbf{z}) = \exp(-(\kappa_1(\mathbf{x}, \mathbf{x}) + \kappa_1(\mathbf{z}, \mathbf{z}) - 2\kappa_1(\mathbf{x}, \mathbf{z})) / (2\sigma^2))$.

In many cases, κ does not need to be an explicit function if the kernel matrix can be given directly. Examples of that include the Latent Semantic Kernel and our proposed Spectral Kernel. The idea is that while some kernels can be represented using explicit functions, many are implicitly represented without one. Regardless of whether κ is an explicit function, the matrix serves as the underlying representation of a kernel capturing all the information required for supervised or unsupervised learning. More interesting perhaps, is that the underlying idea found in spectral embedding and clustering methods (proposed in recent years) coincides with that of kernel-based methods, i.e., a symmetric matrix is used in the analysis. As a result, there are some interesting properties that we can learn about kernels through spectral properties.

A symmetric matrix $\mathbf{S} = (\mathbf{S}_{ij})_{n \times n}$ (where $\mathbf{S}_{ij} = \mathbf{S}_{ji}$) is naturally mapped to an undirected graph $\mathcal{G}(\mathbf{S})$, where its adjacency matrix is $\mathbf{S}$. In spectral graph theory, the spectral component of the transformed $\mathbf{S}$ has a natural relationship with the structure and properties of the graph $\mathcal{G}(\mathbf{S})$ [Chu97]. Furthermore, let $\mathcal{G}(\mathbf{S}) = \langle V, E, \mathbf{S} \rangle$ be the graph of $\mathbf{S}$, where V is the set of n vertices and E is the set of weighted edges. Each vertex i of $\mathcal{G}(\mathbf{S})$ corresponds to the i -th column (or row) of $\mathbf{S}$, and the weight of each edge $\widehat{ij}$ corresponds to the non-diagonal entry $\mathbf{S}_{ij}$. For any two vertices (i, j) , a larger value of $\mathbf{S}_{ij}$ indicates a higher connectivity, and vice versa. From the above, we have the following interesting spectral properties:

Eigenvalues The spectrum of the Normalized Laplacian transformation of $\mathbf{S}$ reveals the embedding clustering structure of $\mathcal{G}(\mathbf{S})$ with different global bisection (or cut) criteria [Chu97].

Eigenvectors Correspondingly, the i -th *eigenvector* naturally explains the meaning of the i -th *eigenvalue*. This led to the development of spectral clustering [SM00, MS01, NJW01].

4.1.1 Graph Cut Criteria of Kernel Matrices

In spectral clustering, since $\mathbf{S}$ is actually an adjacency matrix of the weighted graph $\mathcal{G}(\mathbf{S})$, finding the clustering structure of $\mathbf{S}$ can be transformed into the problem of

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

Table 4.1: Solution to two graph cut criteria used in spectral clustering: (i) the corresponding transformation by eigen-decomposition of the matrix $\Gamma(\mathbf{S})$; (ii) on an incoming data $\mathbf{x}$, the similarity with the training examples is computed as $\mathbf{S}_x$ and its corresponding transformation is $\tau(\mathbf{S}_x)$. *Note:* d_x is the sum of all elements in $\mathbf{x}$; the original version of the normalized Laplacian matrix should be $\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}$; see Section 4.1.2 and Lemma 4.1.

		Average Volume	Normalized Cut
	Criterion	$\max \frac{\text{vol}(A)}{ A } + \frac{\text{vol}(B)}{ B }$	$\min \frac{\text{cut}(A,B)}{\text{vol}(A)} + \frac{\text{cut}(A,B)}{\text{vol}(B)}$
	Solution to criterion	$\mathbf{S}\mathbf{x} = \lambda\mathbf{x}$	$(\mathbf{D} - \mathbf{S})\mathbf{x} = \lambda\mathbf{D}\mathbf{x}$
(i)	Transformed $\mathbf{S}$	$\Gamma_I(\mathbf{S}) = \mathbf{S}$	$\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$
(ii)	Transformed $\mathbf{S}_x$	$\tau_I(\mathbf{S}_x) = \mathbf{S}_x$	$\tau_N(\mathbf{S}_x) = \mathbf{D}^{-1/2}\mathbf{S}_x d_x^{-1/2}$

finding an optimum graph cut in $\mathcal{G}(\mathbf{S})$. Notably, a different graph cut criterion leads to a different solution of $\mathcal{G}(\mathbf{S})$.

In the case of Table 4.1, the criterion is to find an optimal cut of $\mathcal{G}(\mathbf{S})$ such that we have two non-overlapping subsets $A, B \subseteq V$ satisfying the conditions $A \cap B = \emptyset$ and $A \cup B = V$, where $|A|$ is the number of vertices or data points; $\text{vol}(A) = \sum_{i \in A} d_i$ is the volume with $d_i = \sum_{j \in V} \mathbf{S}_{ij}$ being the degree of the vertex i ; $\text{cut}(A, B) = \sum_{i \in A, j \in B} \mathbf{S}_{ij}$ is the cut between A and B ; and $\mathbf{D}$ is the diagonal matrix formed from the degrees of the vertices. In both solutions, the second largest eigenvalues (a.k.a. interested eigenvalues) and the corresponding eigenvectors relating to the equations in Table 4.1 provide the global optimum.

Clearly, the idea of bisecting the kernel matrix can be extended to the first k largest eigenvalues or eigenvectors. And this observation leads to the construction of multi-way spectral clustering. In the criterion of *average volume*, if we consider the inner product matrix of the term-document matrix (without normalization) as $\mathbf{S}$, then its first k largest eigenvectors is used in LSI for information retrieval. In the criterion of *normalized cut*, the second largest eigenvector of $\Gamma_N(\mathbf{S})$ is also the clustering information used in the normalized cut image segmentation algorithm [SM00]. Finally, we have the NJW clustering algorithm [NJW01] when we consider the first k largest eigenvectors of $\Gamma_N(\mathbf{S})$.

We can thus conclude the following. First, the criterion determines the solution and transformation of the kernel matrix that in turn, affects the behavior in the learning module. Second, the type of application to be delivered by the kernel is determined by how the eigenvalues and/or eigenvectors are used. The spectral kernel, presented next, is the result of exploiting these observations.

4.1.2 Computing the Spectral Embedding Space

Among the different spectral clustering techniques proposed in the literature, e.g., [SM00, MS01, NJW01], an analysis of the underlying mathematical representation suggests that with appropriate transformations, they are essentially reduced to a common representation. This observation motivates the first contribution of the spectral kernel – a unifying framework which by means of different parameters, creates different kernel instances that exhibit different behaviors. We will first prove the existence of this framework, and then show how the spectral embedding is computed.

From Table 4.1, we see that it is easy to mathematically transform the solutions into a standard eigendecomposition problem of symmetric matrices, i.e., $\Gamma(\mathbf{S})\mathbf{x} = \lambda\mathbf{x}$. To do so however, requires the transformation of $\mathbf{S}$, and this is dependent on the solution to the cut criteria. In Table 4.1, $\Gamma_I(\mathbf{S})$ represents the original matrix while $\Gamma_N(\mathbf{S})$ is the normalized Laplacian matrix. From spectral graph theory [Chu97], $\mathbf{K}_1 = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{S})\mathbf{D}^{-1/2}$ is actually the normalized Laplacian matrix. Notably, $\mathbf{K}_1$ has the same eigenvectors as $\mathbf{K}_2 = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$ and the eigenvalues are related by $\text{eig}(\mathbf{K}_1) = \{1 - \lambda | \lambda \in \text{eig}(\mathbf{K}_2)\}$, where $\text{eig}(\cdot)$ is the set of eigenvalues of a symmetric matrix. Furthermore, the interested eigenvalues change from the smallest in $\mathbf{K}_1$ to the largest in $\mathbf{K}_2$. Therefore, it is actually possible to compute the normalized Laplacian matrix using $\mathbf{K}_2$ giving us $\Gamma_N(\mathbf{S}) = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2}$. In fact, the equivalent relationship between $\mathbf{K}_1$, $\mathbf{K}_2$, and the stochastic matrix $\mathbf{P} = \mathbf{D}^{-1}\mathbf{S}$ can be proven, and therefore in the remaining part of this chapter, we will use $\mathbf{K}_2$ in place of $\mathbf{K}_1$ and $\mathbf{P}$ if any.

Lemma 4.1 (Equivalence of $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{P}$) *If λ and $\mathbf{x}$ are correspondingly the eigenvalue and eigenvector of matrix $\mathbf{K}_1$, then $(1 - \lambda)$ are the eigenvalues of the matrices $\mathbf{K}_2$ and $\mathbf{P}$; and the eigenvectors of $\mathbf{K}_2$ and $\mathbf{P}$ are $\mathbf{x}$ and $\mathbf{D}^{-1/2}\mathbf{x}$ respectively.*

Proof: By definition of $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{P}$, we have:

$$\mathbf{K}_1 = \mathbf{I} - \mathbf{K}_2 \quad (\text{Eq. 4.1})$$

$$\mathbf{K}_2 = \mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2} \quad (\text{Eq. 4.2})$$

Suppose λ and $\mathbf{x}$ are eigenvalue and eigenvector of $\mathbf{K}_1$, i.e., $\mathbf{K}_1\mathbf{x} = \lambda\mathbf{x}$. By Eq. 4.1, substituting $\mathbf{K}_1$ with $\mathbf{K}_2$ gives us $(\mathbf{I} - \mathbf{K}_2)\mathbf{x} = \lambda\mathbf{x}$. After transformation, we have

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

$\mathbf{K}_2\mathbf{x} = \mathbf{I}\mathbf{x} - \lambda\mathbf{x} = (1 - \lambda)\mathbf{x}$. This proves the relationship between $\mathbf{K}_1$ and $\mathbf{K}_2$. We next complete the proof by showing the equivalence of $\mathbf{K}_2$ and $\mathbf{P}$. Suppose now λ and $\mathbf{x}$ are eigenvalue and eigenvector of $\mathbf{K}_2$, i.e., $\mathbf{K}_2\mathbf{x} = \lambda\mathbf{x}$. By Eq. 4.2, substituting $\mathbf{K}_2$ with $\mathbf{P}$ gives us $\mathbf{D}^{1/2}\mathbf{P}\mathbf{D}^{-1/2}\mathbf{x} = \lambda\mathbf{x}$. By left-multiplication of the matrix $\mathbf{D}^{-1/2}$ on this equation, we get $\mathbf{P}\mathbf{D}^{-1/2}\mathbf{x} = \lambda\mathbf{D}^{-1/2}\mathbf{x}$. Here, λ is the eigenvalue of $\mathbf{P}$, and $\mathbf{D}^{-1/2}\mathbf{x}$ is the eigenvector of $\mathbf{P}$.

Within this framework, we can compute the spectral embedding for any specific instance of the spectral kernel. The steps to do so are given in Figure 4.1. After the spectral components of $\Gamma(\mathbf{S})$ is computed, the k interested extreme eigenvalues and eigenvectors are selected to construct the reduced data space. Let the first k interested eigenvalues of $\Gamma(\mathbf{S})$ be $\lambda_1 \triangle \lambda_2 \dots \triangle \lambda_k$, where “ $\triangle$ ” is “ $\leq$ ” or “ $\geq$ ” according to the different matrix transformation Γ , and $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k$ as their corresponding eigenvectors each of dimension n . The k dimensional data space is constructed by the two steps shown in Figure 4.1. The first step has two implementations that can be selected based on the desired application behavior.

Step 1(a) has been proved to be effective and useful on $\Gamma_N(\mathbf{S})$ in revealing the clustering structure of $\mathbf{S}$ in [MS01]. When considered with Step 2, its effectiveness was proven, both theoretically and empirically, in [NJW01]. When Step 1(a) is used with $\Gamma_I(\mathbf{S})$, it has proven applications in latent semantic analysis and indexing [DDL⁺90, PTRV98]. Step 1(b) on the other hand is well-suited in the context of k -rank approximation when used with $\Gamma(\mathbf{S})$ as supported by Lemma 4.2 below. Furthermore, latent semantic analysis has shown that the k -rank approximation of a similarity matrix (that is also $\Gamma_I(\mathbf{S})$) incorporates semantic information in measure of similarity between two data points (the same conclusion was also given in latent semantic kernels). We will elaborate this point in Section 4.2.3.

Lemma 4.2 (Approximation of $\Gamma(\mathbf{S})$ by embedding of Step 1(b)) *The matrix $\mathbf{S}'$, computed by the embedding of Step 1(b) using inner product (i.e., $\mathbf{S}'_{ij} = \mathbf{y}_i^T \mathbf{y}_j$), is the best k -rank matrix approximation of $\Gamma(\mathbf{S})$.*

Proof: Lemma 4.2 is a variant of the Eckart-Young theorem [GR71]. Given $\mathbf{S}_{n \times n} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}$ (singular value decomposition), $\mathbf{A} = \mathbf{S}_k = \mathbf{U}_k\mathbf{\Lambda}_k\mathbf{V}_k$ is the best rank- k approximation to $\mathbf{S}$ that minimizes $\|\mathbf{A} - \mathbf{S}\|_F^2$ among all matrices $\mathbf{A}$ with rank k (F denotes Frobenius norm of a matrix). And because $\mathbf{S}$ is symmetric, singular values and vectors of $\mathbf{S}$ are the same as eigenvalues and eigenvectors of $\mathbf{S}$.

Step 1(a).

Directly get $\mathbf{y}_i = (\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_k^i)^T$,
where $i=1, 2, \dots, n$.

Step 1(b).

Compute $\mathbf{y}_i = (\sqrt{\lambda_1} \mathbf{v}_1^i, \sqrt{\lambda_2} \mathbf{v}_2^i, \dots, \sqrt{\lambda_k} \mathbf{v}_k^i)^T$,
where $i=1, 2, \dots, n$ or $\mathbf{\Lambda}_k^{1/2}(\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$.

Step 2 (Optional).

Renormalize each $\mathbf{y}_i$ to have the unit length (i.e. $\mathbf{y}_i = \frac{1}{\|\mathbf{y}_i\|} \mathbf{y}_i$).

Figure 4.1: Spectral embedding in the spectral kernel. Let the projected k -dimensional data space be $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n \in \mathbb{R}^{k \times 1}$, and the first k interested eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ are positive. *Note:* $\mathbf{v}_j^i$ denotes the i -th coordinate of the eigenvector $\mathbf{v}_j$. $\mathbf{\Lambda}_k$ is the truncated diagonal of $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, its last $(n - k)$ diagonal entries are set 0.

4.2 Spectral Kernels

In the previous section, the spectral graph analysis of the kernel matrix shows that the spectral embedding (obtained by either Step 1(a) or 1(b)) reveals more latent semantics by encoding the graph cut information to the spectral embedding. By projecting the original feature vectors onto the spectral embedding subspace, we can define a kernel, originating from this subspace, through a particular choice of similarity measure. This effectively registers the clustering information inherent in the subspace into the spectral kernel (SK).

Computing the spectral kernel for classification can be done in three phases: (i) transformation, (ii) spectral embedding, and (iii) kernel computation. Thus, we define the spectral kernel with three components, i.e., $\text{SK}\langle \mathbf{T}, \mathbf{E}, \mathbf{S} \rangle$, where $\mathbf{T}$ is the transformation in Table 4.1; $\mathbf{E}$ is the embedding step in Figure 4.1; and $\mathbf{S}$ is one of the similarity measure in Table 4.2 selected to compute the final spectral kernel value in the spectral embedding subspace.

In classification, the kernel can contain values from either the training set, or from the training set and its new input (from the testing set). Since during transformation and spectral embedding, the input kernel matrix $\mathbf{S}$ only holds kernel values from the training set, the spectral embedding can be computed by following the steps given in Section 4.1.2. In the case where we need to compute the kernel values from both the training and testing set, a different way to compute the spectral embedding of the new input within the same subspace of the training set is needed.

4.2.1 Transforming and Spectral Embedding of New Input

When a new input arrives, its spectral embedding is computed in a similar fashion as described in Section 4.1.2. The difference is that the transformation and computation of spectral embedding is applied to the vector $\mathbf{S}_x$ rather than the symmetric matrix. This gives rise to a different transformation and computation of the spectral embedding. After getting the spectral embedding of the new input, the kernel values can be updated with the same similarity measure used during training.

The new input can be given in the form of a vector containing kernel values, i.e., $\mathbf{S}_x = (\mathbf{S}_{1x}, \mathbf{S}_{2x}, \dots, \mathbf{S}_{nx})^T$, where $\mathbf{S}_{ix}$ represents the kernel value between the i -th training example and itself. The rationale to why we used $\mathbf{S}_x$ instead of the vector $\mathbf{x}$ in the original space is that spectral kernels are based on the other input kernels. In order to compute the spectral embedding of the new input, there is a need to recompute the transformation and the embedding space. Therefore, the vector transformation corresponding to its matrix transformation $\Gamma(\mathbf{S})$ is defined as $\tau(\mathbf{S}_x)$ and is given in Table 4.1 for different Γ . After obtaining $\tau(\mathbf{S}_x)$, the following lemma defines how the spectral embedding of the new input is computed.

Lemma 4.3 (Spectral embedding of new input) *Given a kernel matrix $\mathbf{S}$ for training data, its transformation $\Gamma(\mathbf{S})$, the k interested eigenvalues/vectors of $\Gamma(\mathbf{S})$ ($\lambda_i \geq 0$ and $\mathbf{v}_i, i = 1, 2, \dots, k$), and a new input $\mathbf{S}_x$ in form of kernel values, the spectral embedding of $\mathbf{S}_x$ for Step 1(a) is $\mathbf{y} = \mathbf{\Lambda}_k^{-1} \mathbf{V}^T \tau(\mathbf{S}_x)$, and Step 1(b) is $\mathbf{y} = \mathbf{\Lambda}_k^{-1/2} \mathbf{V}^T \tau(\mathbf{S}_x)$; where the diagonal matrix $\mathbf{\Lambda}_k = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_k, 0, \dots, 0)$, and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k, \mathbf{v}_{k+1}, \dots, \mathbf{v}_n)^T$.*

Proof: By eigendecomposition, $\Gamma(\mathbf{S}) = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^T$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$. Therefore, $\Gamma(\mathbf{S})$ can be approximated by $\Gamma(\mathbf{S}) \approx \mathbf{V} \mathbf{\Lambda}_k \mathbf{V}^T = (\mathbf{\Lambda}_k^{1/2} \mathbf{V}^T)^T (\mathbf{\Lambda}_k^{1/2} \mathbf{V}^T)$, since the interested k eigenvalues are the largest positive eigenvalues of $\Gamma(\mathbf{S})$. Therefore, each training example i can be represented by the spectral embedding $\mathbf{y}_i = \mathbf{\Lambda}_k^{1/2} (\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$ in terms of matrix approximation, where $\mathbf{y}_i$ is the spectral embedding of the i -th training example by Step 1(b) from Table 4.1. If we assume that $\mathbf{y}$ is also the spectral embedding of the new input in terms of matrix approximation, then $\mathbf{y}$ should be the spectral embedding obtained by Step 1(b). By matrix approximation, we can therefore approximate the transformed kernel vector $\tau(\mathbf{S}_x)$ of the new input $\mathbf{S}_x$

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

Table 4.2: The similarity measures $s(\mathbf{x}, \mathbf{y})$ used in the computation of spectral kernels.

	Inner product	Extension of Euclidean distance	Pearson correlation coefficient
Symbol	S_I	S_E	S_P
Formula	$\mathbf{x}^T \mathbf{y}$	$\exp(-\frac{\ \mathbf{x}-\mathbf{y}\ ^2}{\sigma})$	$\frac{(\mathbf{x}-\bar{\mathbf{x}})'(\mathbf{y}-\bar{\mathbf{y}})}{\ \mathbf{x}-\bar{\mathbf{x}}\ \ \mathbf{y}-\bar{\mathbf{y}}\ }$

in the same way. Thus, we have $\tau(\mathbf{S}_x) = (\mathbf{\Lambda}_k^{1/2} \mathbf{V}^T)^T \mathbf{y}$. Since $\mathbf{\Lambda}_k^{1/2} \mathbf{V}^T$ is an orthogonal matrix, it can be solved by taking the matrix inverse to obtain $\mathbf{y} = \mathbf{\Lambda}_k^{-1/2} \mathbf{V}^T \tau(\mathbf{S}_x)$. This gives the result in Step 1(b). Further, the spectral embedding by Step 1(a) can be computed by multiplying the diagonal matrix $\mathbf{\Lambda}_k^{-1/2}$, which gives the spectral embedding of the new input $\mathbf{y} = \mathbf{\Lambda}_k^{-1/2} \left(\mathbf{\Lambda}_k^{-1/2} \mathbf{V}^T \tau(\mathbf{S}_x) \right) = \mathbf{\Lambda}_k^{-1} \mathbf{V}^T \tau(\mathbf{S}_x)$ by Step 1(a).

Essentially, Lemma 4.3 specifies how to project the new input onto the spectral embedding space given by the training examples. And Step 2 of Figure 4.1 served as the optional step that can be applied to the spectral embedding of the new input (computed either by Step 1(a) or 1(b)), and its use is dependent on whether Step 2 was used during training so that the new input can be compared with the training examples within the same embedding space.

4.2.2 Computing the Spectral Kernel

When the spectral embedding of the training and testing set is ready, the final step is to compute the spectral kernel values from the spectral embedding using a selected similarity measure. This step is flexible and many typical similarity measures can be used. Table 4.2 lists some possible options for the spectral kernel. Notice that the magnitude of the similarity measure need not be constrained between 0 and 1 since there is no strict requirement on the range of possible kernel values in classification.

4.2.3 Relationship to Latent Semantic Kernel

We conclude this section with the proof that the latent semantic kernel is only a specific instance of the spectral kernel. The objective is two-fold: (i) we want to clarify the difference between spectral kernels and latent semantic kernels as they appear similar at first glance; and (ii) by this proof, we seek to establish spectral kernels as a framework under which spectral clustering information can be further researched to improve the task of classification.

Theorem 4.1 *Given the term-document matrix $\mathbf{D}$, the latent semantic kernel is a specific instance of the spectral kernel, i.e., $SK\langle\Gamma_I(\mathbf{D}^T\mathbf{D}), 1(b), S_I\rangle$ reduces to the latent semantic kernel.*

Proof: From [CSTL01], we have the following facts: latent semantic kernels (LSK) are computed from the term-document matrix $\mathbf{D}$ or $\mathbf{S} = \mathbf{D}^T\mathbf{D}$; the LSK matrix of training set is $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T$; the LSK values between the training set $\mathbf{d}_i$ and the new input $\mathbf{d}$ is $\kappa(\mathbf{d}_i, \mathbf{d}) = (\mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T\mathbf{t})_i$, where $\mathbf{t} = \mathbf{D}^T\mathbf{d}$ and the matrix $\mathbf{V}$ is obtained from eigen decomposition $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$. Using the above, we prove that spectral kernels with a configuration of $SK\langle\Gamma_I(\mathbf{S}), 1(b), S_I\rangle$ gives the same $\mathbf{K}$ and $\kappa(\mathbf{d}_i, \mathbf{d})$ as LSK. We denote $\hat{\mathbf{K}}$ as the SK matrix of the training set and $\hat{\kappa}(\mathbf{d}_i, \mathbf{d})$ as the SK values between the training set $\mathbf{d}_i$ and the new input $\mathbf{d}$. Since the input kernel matrix of SK is $\mathbf{S} = \mathbf{D}^T\mathbf{D}$ and is computed by the inner product of each document $\mathbf{d}_i$, we can easily get the input kernel values of the new input $\mathbf{d}$ as $\mathbf{S}_x = \mathbf{D}^T\mathbf{d}$. As the transformation is Γ_I , we have $\Gamma_I(\mathbf{S}) = \mathbf{S}$ and $\tau_I(\mathbf{S}_x) = \mathbf{S}_x = \mathbf{D}^T\mathbf{d} = \mathbf{t}$. Then, we compute the spectral embedding (using Step 1(b)) of $\mathbf{d}_i$ as $\mathbf{y}_i = \mathbf{\Lambda}_k^{1/2}(\mathbf{v}_1^i, \mathbf{v}_2^i, \dots, \mathbf{v}_n^i)^T$, where $\mathbf{v}_i$ is the i -th eigenvector of $\mathbf{S}$ or the i -th column of $\mathbf{V}$. Further, the spectral embedding of the new input $\mathbf{d}$ (by Step 1(b)) is $\mathbf{y} = \mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\tau(\mathbf{S}_x) = \mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\mathbf{t}$. Since the final component is the inner product S_I , we immediately get $\hat{\mathbf{K}} = (\mathbf{y}_i^T\mathbf{y}_j)_{n \times n} = (\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T)^T(\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T) = \mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T = \mathbf{K}$ and $\hat{\kappa}(\mathbf{y}_i, \mathbf{y}) = \mathbf{y}_i^T\mathbf{y} = \left((\mathbf{\Lambda}_k^{1/2}\mathbf{V}^T)^T(\mathbf{\Lambda}_k^{-1/2}\mathbf{V}^T\mathbf{t})\right)_i = (\mathbf{V}\mathbf{\Lambda}_k\mathbf{V}^T\mathbf{t})_i = \kappa(\mathbf{y}_i, \mathbf{y})$.

4.3 Experimental Results

We evaluated our spectral kernels on two text data sets, namely Medline1033 and Reuters-21578, to demonstrate its applicability and effectiveness. We chose these two data sets for easy comparison with the experiments reported in [CSTL01]. Compared with the baseline method, i.e., SVM classifier with linear kernel without feature selection, our results were either much better or positively on-par, than that of baseline method according to F_1 measure.

Medline1033 This data set contains 1,033 documents and 30 queries obtained from the National Library of Medicine. Following the experimental setting reported in [CSTL01], we focused on query23 and query20. Each query contains 39 relevant documents. We selected 90% of them as the training set and the remaining

Figure 4.2: Generalization performance of the SVM classifier with the proposed spectral kernel $SK\langle\Gamma_N, 1(a), S_I\rangle$ and the linear kernel for the Medline1033 data set.

10% as the testing set, while always having 24 relevant documents in the training set and 15 relevant documents in the test set. We performed 50 random splits of this data set in our experiments and reported the average performance.

Reuters21578 This data set contains 21,578 news articles organized in 135 categories and has been widely used in text classification [Lew]. Each document is labelled with zero, one, or more categories. We used the **ModApte** split to obtain the training and testing set, and conducted our experiments on the top ten largest categories. They are **earn**, **acq**, **money-fx**, **grain**, **crude**, **trade**, **interest**, **ship**, **wheat**, and **corn**. The number of training documents and testing documents are 6,494 and 2,548 respectively.

In our experiments, each document was represented as a feature vector with term frequency (TF) weighting and all document vectors were normalized to unit length. Terms were stemmed after stopword removal. We used SVM^{light} with default parameter setting for spectral embedding and baseline in all experiments. As a binary classifier, one SVM classifier was trained for each category. The SVM classifier with linear kernel was used as the baseline method where no feature selection was performed. The classification performance was evaluated with F_1 for each category and *Micro*- and *Macro*-averaged F_1 on all the categories [Yan97].

4.3.1 Results on Medline1033

We configured a spectral kernel of the form $SK\langle\Gamma_N, 1(a), S_I\rangle$ and compared its performance against the baseline method. The results are reported in Figure 4.2. We started

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

Figure 4.3: Generalization performance of the SVM classifier with the proposed spectral kernel $SK(\Gamma_N, 1(b)2, S_I)$ for the classes of Reuters21578 data set: **corn**, **acq**, **money-fx**, **grain**, **crude**, **trade**, **interest**, **ship**, **wheat**.

with a small k feature space for the classifier with spectral kernel and increased the dimensionality until the classification performance deteriorated, i.e., when $k > 250$. The results of both **query23** and **query20** proved to be very encouraging.

With a small k (less than 200), the spectral kernel $SK(\Gamma_N, 1(a), S_I)$ increased quickly to a result that was much better than baseline method according to F_1 measure. In particular, for **query23**, the best performance delivered by the spectral kernel was 84.62%, almost twice that of the baseline method which was 42.11%.

4.3.2 Results on Reuters21578

On the Reuters data set, we configured the spectral kernel to $SK(\Gamma_N, 1(b)2, S_I)$ giving the results shown in Figure 4.3. While performing comparably on categories **earn** and **interest**, the spectral kernel outperformed the linear kernel (i.e., baseline method) on the remaining eight categories with small k values (generally less than 300). In particular, on the **ship** category, the best F_1 achieved by spectral kernel was 89.16% which is much higher than 82.05% delivered by linear kernel. More importantly, the F_1 performance under most values of k were much higher than the baseline as shown in Figure 4.3(h). This is an encouraging result showing the effectiveness of spectral kernels in text classification tasks.

Furthermore, eight of the performance plots on Reuters data set, and two of the performance plots on Medline data set showed that a small value of k (usually $100 \leq k \leq 300$) is often sufficient to achieve good F_1 performance. This observation is different from the selection of k in latent semantic kernels, where a larger k implied better performance (i.e., in the range of 500 to 1000 as observed in [CSTL01]). The small values of k are encouraging because they lead to shorter runtime to achieve the same classification accuracy as LSK.

Nevertheless, the experimental results on Reuters data set did reveal a shortcoming of the spectral kernel: there is not a fixed value of k that ensures consistent performance for different categories. This will be a practical limit that requires automated mechanisms to determine k . While we are working on this as part of our future work, using a single value of k enables us to compare the spectral kernel against the linear kernel objectively. As Table 4.3 shows, we achieved better performance than the baseline method when $k = 270$, and comparable performance when $k = 120$.

4.3.3 Computational Complexity

We end this section by discussing the computational complexity of the spectral kernel. If the symmetric matrix $\mathbf{S}$ has n rows and columns, then the complexity of transformation from $\mathbf{S}$ to $\Gamma(\mathbf{S})$ would be $O(n^2 + n)$. The study of how to effectively compute the largest eigenvalues has been well-developed within the domain of symmetric generalized eigen problems that arised from structural analysis in physics and

Table 4.3: F_1 using spectral kernel $SK\langle\Gamma_N, 1(b)2, S_I\rangle$ and linear kernel on the ten Reuters categories with the best F_1 results in bold.

category	k			baseline
	120	270	420	
earn	0.987	0.986	0.988	0.990
acq	0.966	0.977	0.973	0.971
money-fx	0.861	0.847	0.834	0.857
grain	0.908	0.914	0.918	0.914
crude	0.900	0.934	0.922	0.917
trade	0.875	0.880	0.873	0.878
interest	0.811	0.805	0.798	0.827
ship	0.892	0.843	0.813	0.821
wheat	0.846	0.855	0.825	0.831
corn	0.701	0.852	0.849	0.828
micro-avg	0.939	0.945	0.942	0.944
macro-avg	0.875	0.889	0.879	0.883

computational chemistry. Thus, a series of mature mathematical tools are available for our purpose.

We can use the **Lanczos** method to compute the eigenvalues in $k \ll n$ iterations allowing the process to converge quickly [GL96]. Since the complexity of each iteration is $O(n^2 + n)$, the final complexity of this method is therefore $O(k(n^2 + n))$. If we ignore the very small k in real computations, the complexity of our method becomes $O(n^2 + n)$. In our experiments, the ARPACK package is used. It is a collection of **Fortran77** subroutines that effectively computes the k eigenvalues and eigenvectors using only $2nk + O(k^2)$ storage space (with no auxiliary storage [LSY98] required). Since our matrices are symmetric, the method used in ARPACK actually reduces to a variant of the **Lanczos** process known as Implicitly Restarted **Lanczos** Method (IRLM).

4.4 Summary

In this chapter, we proposed a new kernel that uses the semantics extracted from spectral clustering. Unlike the latent semantic kernels, spectral kernels are unique not only by the virtue of using spectral clustering information, but also its ability to support incremental updates to the kernel matrix that keeps the cost of training minimal. Furthermore, we have shown that we can obtain the spectral embedding of both training and testing sets by matrix approximation. Our experiments on text

CHAPTER 4. SPECTRAL KERNELS FOR CLASSIFICATION

data proves the feasibility of spectral kernels in terms of accuracy and efficiency. The results from our experiments are promising. In most cases, the spectral kernel achieved substantial improvement in classification performance without any loss of accuracy.

The closest piece of related work, to our knowledge, is the classification of projected k -dimensional space obtained by spectral clustering algorithms [KKM03]. However, the proposal by Kamvar *et al.* suffers from three drawbacks. First, if a new input arrives, there is a need to eigen-decompose the new $\mathbf{S}$ (which includes the new input) to obtain the new spectral space. This is time-consuming for classification during operation where large number of data may arrive. Second, since the spectral space is dependent on the testing set rather than the training set, the spectral space is unstable if the testing set is highly random. Third, the similarity relationship between training data is not fully exploited to significantly improve the accuracy of classification. Our proposal overcomes these drawbacks and provides a kernel framework of applying spectral clustering to classification.

Chapter 5

Visual Terrain Analysis of High-dimensional Data by Resonance Model

The subfield of data analysis is essentially a collection of algorithms that focus on analyzing large data sets of high-dimensionality. More often than not, the contribution of these algorithms is to address the dimensionality curse when trying to provide effective and efficient results for a given user query. However, as an effectively exploratory data analysis technique, visualizing the high-dimensional data is seldom investigated in the literature. In many cases, users may not clearly tell what they need to acquire from data analysis, and thus visualization of data is an effective means to help users explore the data in a flexible manner. Several established types of visualization methods have been widely used for exploring data [Kei02], e.g. geometric, icon-based, hierarchical and pixel-oriented techniques¹. However, all these standard visualization techniques fail to be applied to explore high-dimensional data as indicated in [Kei02]. To visualize the high-dimensional data, a certain degree of dimensionality reduction is needed before using the visualization techniques, i.e., projecting the high-dimensional data set to 2 or 3-dimensions is typically done. As shown in [Kei02], typical dimensionality reduction methods, e.g., Principal Component analysis (PCA) and multidimensional scaling [CC01] were used to reduce the dimensions and visualization techniques were then applied to visualize the data in the reduced dimension space.

However, the reduced dimension space always loses the detailed information contained in the original high-dimensional space, e.g., how the data is distributed. Let us

¹Details of these techniques have been briefly reviewed in Section 2.2.4 of Chapter 2.

take an example to show it. Our example is based on the survey of popular movies. Figure 5.1(a) shows the rating matrix W of 4 reviewers (R_i) on 5 movies (M_j), where each movie is rated on a scale of 1 to 20. When we use PCA to reduce the dimensionality to 2, its output is shown in Figure 5.1(c) and (d). The relationship distances among the reviewers and movies can be clearly revealed from these two figures. However, Figure 5.1(c) and (d) can only answer what objects or features are related, but fail to give the reason why objects and features are related in such a way. In many cases, the second question is often what analysts want to ask and explore. Most visualization techniques based on dimensionality reduction often fail to answer the second question. To carefully analyze the second question “*why objects and features are correlated (or similar) in such a way?*”, it is a problem of studying the distribution of data values in high-dimensional space. Therefore, in this chapter, we introduce the concept of *data terrain* to visually reveal the distribution of the values in high-dimensional data set. The basic idea of *data terrain* is to view the high-dimensional data (i.e., a matrix) as a terrain where each cell in the matrix is a site (or place) and cell values are their longitudes. The key step in visual terrain analysis is to reorder the matrix by collecting high/low values together to a certain place (i.e. left-top corner of the matrix) so that ‘mountains’ and ‘valleys’ are visually and intuitively shown to users. In the above example, we visually view the rating matrix W in 3D space, as shown in Figure 5.1(b) (i.e., *reviewer*, *movie*, and *rating* are three dimensions of this space. The *rating* becomes the longitude in this terrain.). Interestingly, a direct 3D ‘plot’ of W does not seem to reveal any interesting insights of data. But if we were to reorder W into W' as shown in Figure 5.1(e), we have a 3D terrain of W' as depicted in Figure 5.1(f). Notably, this terrain provides the insights and details why reviewers and movies are correlated in such a way.

Since some patterns pursued by data mining algorithms, i.e., frequent itemset, biclusters and selection of differentially expressed genes, can be viewed as the direct or indirect collection of high or low values in the high-dimensional data (or matrix), visualization of ‘mountains’ and ‘valleys’ in *data terrain* may contribute to the discovery of these patterns. The study in this chapter has shown the relationships between *data terrain* and several patterns theoretically and empirically. More important, a terrain visualization algorithm particularly designed for gene selection has shown that

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

Figure 5.1: The rating matrix for 4 reviewers and 5 movies

data terrain can not only visualize high-dimensional data, but also help analyze data effectively.

We next formally give the definition of data terrain and pose a basic terrain discovery problem of collecting high values (‘mountains’) in Section 5.1. To efficiently approach this problem, we introduced the resonance model in Section 5.2. For illustrating how the concept “data terrain” helps analyze high-dimensional data, we also proposed a more complex terrain discovery problem and applied it to the problem of biomarker discovery and visualization (differentially expressed genes) in gene expression data in Section 5.3.4. Finally, we conclude our work in Section 5.4.

5.1 Problem Formulation of Generic and Basic Terrains

5.1.1 Generic Terrain of Data

Let $\mathcal{O}$ be a set of objects, where $o \in \mathcal{O}$ is defined by a set of attributes $\mathcal{A}$. Further, let w_{ij} be the magnitude of o_i over $a_j \in \mathcal{A}$. Then we can represent the relationship of all objects and their attributes in a matrix $W = (w_{ij})_{|\mathcal{O}| \times |\mathcal{A}|}$ for the weighted bipartite graph $G = (\mathcal{O}, \mathcal{A}, E, W)$, where E is the set of edges.

Considering the example shown in Figure 5.1(b) and 5.1(f), to discover a data terrain is a procedure of moving from W to W' for forming ‘mountains’ and ‘valleys’. Conceptually, moving from W to W' is simply the reordering of the matrix to form the ‘mountains’ and ‘valleys’. Therefore, a generic data terrain can be defined as follows,

Definition 5.1 (Generic Terrain of Data: Matrix Reordering) *Given a non-negative matrix $W_{m \times n}$ and its corresponding bipartite graph $G = (\mathcal{O}, \mathcal{A}, E, W)$, a generic terrain of W is two ordered sequences of $\mathcal{O}$ and $\mathcal{A}$ such that, after reordering W to W' according to the ordered sequences of $\mathcal{O}$ and $\mathcal{A}$, high or low values in W' are collected together.*

Definition 5.1 is a general description of data terrain by emphasizing ‘mountains’ (i.e., collection of high values) and ‘valleys’ (collection of low values). Different types of ‘mountains’ or ‘valleys’ locations will lead to different types of terrains discovered for visualization. In this section, we first formulated a basic terrain discovery problem of collecting the high values of a matrix to the left-top corner of the reordered matrix. The left-top corner of the reordered matrix represents the ‘mountain’ available in the terrain of the target data. In Section 5.3, then we will discuss a more complex terrain existing in the gene expression data sets. Please note the data terrain defined in Definition 5.1 is generic and therefore is applicable to both the basic terrain introduced in this section and the complex terrain defined in Section 5.3.

5.1.2 Basic Terrain of Data: Distinct ‘Mountain’

The basic terrain of high-dimensional data is the most distinct ‘mountain’ where most values are much higher than the rest of values. In subsequent sections, we will

show that this basic terrain is closely related to several typical patterns in knowledge discovery and how it is able to help efficiently and effectively visualize and discover them. In this section, we give the formal problem formulation of the basic terrain.

By representing W as a bipartite graph, discovering the most distinct ‘mountain’ in W is equivalent to the problem of discovering a distinctly heavy subgraph (*in a bipartite graph*) whose most edge weights are very high. Before introducing our problem formulation of heavy bipartite subgraph discovery, we firstly introduce a remarkable connection established by Motzkin and Straus in 1965 [MS65]. This connection is between the clique problem in unweighted graph and the quadratic programming problem.

Result of Motzkin and Straus on unweighted graph

Let the $n \times n$ symmetric matrix W be the adjacency matrix of the unweighted graph $G = (V, E)$ and $\mathbf{x}^S = (x_1^S, x_2^S, \dots, x_n^S)^T \in \{0, 1\}^n$ be the characterization vector of a subgraph $S = (V', E')$ (i.e., if a vertex $v_i \in V'$, then $x_i^S = 1$; otherwise, $x_i^S = 0$). Motzkin and Straus proposed a constraint on $\mathbf{x}^S$: $\sum_{i=1}^n x_i^S = 1$, where $\mathbf{x}^S$ satisfying the constraint is called the normalized characterization vector of a subgraph. Then they built the following theorem revealing the connection between the “Lagrangian” function (which is defined in an unweighted graph, i.e., $L_W(\mathbf{x}^S) = (\mathbf{x}^S)^T W (\mathbf{x}^S) = \sum_{i=1}^n \sum_{j=1}^n w_{ij} x_i^S x_j^S$), and the heaviest graph, i.e., “clique”.

Theorem 5.1 (Motzkin-Straus) A clique S of G is maximum, if and only if its normalized characterization vector $\mathbf{x}^S$ is a global maximizer of L_W . If $\tilde{\mathbf{x}}$ is a global maximizer of L_W on the constraint $\sum_{i=1}^n x_i^S = 1$, the clique number of G is related to $L_W(\tilde{\mathbf{x}})$ by the formula $\omega(G) = \frac{1}{1 - L_W(\tilde{\mathbf{x}})}$.

To illustrate how the normalized characterization vector works as a maximizer of L_W , we used an example taken from proteins’ interaction network and showed the result in Figure 5.2. After we get the maximizer $\mathbf{x}^S$ of L_W , according to the definition of characterization vector of a subgraph and Theorem 5.1, it is clear that larger components x_i^S indicate the corresponding vertices have larger possibility to belong to the heavy subgraph. As a result, we sorted components of $\mathbf{x}^S$ in decreasing order² and reordered the adjacency matrix of the example graph according to this order. Comparing Figure 5.2(a) and (b), the reordered adjacency matrix is able to collect non-zeros (i.e., 1s) to the left-top corner for forming a heavy subgraph which is the most distinct and largest one. This is what we expect in a bipartite graph, although Motzkin and Straus formulated it in an unweighted graph (a symmetric matrix). Since the result of Motzkin-Straus has explored the heavy subgraph formulation in

²Because each component of $\mathbf{x}^S$ corresponds to a vertex, the order of components is also the order of vertices in the graph.

Figure 5.2: A small graph of 40 proteins' interaction network is selected from the core *Saccharomyces cerevisiae* in Database of Interacting Proteins (DIP, available at <http://dip.doe-mbi.ucla.edu>). The black points in figures represent non-zeros (or one) in the adjacency matrix, as DIP data is the unweighted graph.

unweighted graph, considering that the a general matrix $W_{m \times n}$ (representing a bipartite graph) is a generalization of symmetric matrix (standing for a graph), similarly we generalize their formulation from unweighted graph to weighted bipartite graph.

Like characterization vector of subgraph in graph, we give the definition of characterization vector of bipartite graph as follows,

Definition 5.2 (Characterization Vector of Bipartite Subgraph) *Given a bipartite graph $G(\mathcal{O}, \mathcal{A}, W)$ with $m = |\mathcal{O}|$, $n = |\mathcal{A}|$ and $W = (w_{ij})_{m \times n}$, where $\mathcal{O}$ and $\mathcal{A}$ are two disjoint sets of vertices and w_{ij} means the edge weight of the i -th vertex in $\mathcal{O}$ and the j -th vertex in $\mathcal{A}$, its subgraph $S(\mathcal{O}', \mathcal{A}', W')$ (i.e., $\mathcal{O}' \subseteq \mathcal{O}$, $\mathcal{A}' \subseteq \mathcal{A}$ and W' is a corresponding submatrix of W), the characterization vector of the bipartite subgraph S , i.e. $\mathbf{x}^S = (x_1^S, x_2^S, \dots, x_m^S)^T$ for characterizing $\mathcal{O}'$ and $\mathbf{y}^S = (y_1^S, y_2^S, \dots, y_n^S)^T$ for characterizing $\mathcal{A}'$, is defined as follows,*

$$x_i^S = \begin{cases} 1, & \text{if } v_i \in \mathcal{O}' \\ 0, & \text{if } v_i \in (\mathcal{O} - \mathcal{O}') \end{cases} \quad y_i^S = \begin{cases} 1, & \text{if } v_i \in \mathcal{A}' \\ 0, & \text{if } v_i \in (\mathcal{A} - \mathcal{A}') \end{cases} \quad (\text{Eq. 5.1})$$

The component x_i^S or y_i^S of characterization vector of a bipartite graph gives the associated value to the i -th vertex $o_i \in \mathcal{O}$ or the i -th vertex $a_i \in \mathcal{A}$ respectively. According to Eq. 5.1, these values indicate whether vertices belong to a heavy subgraph

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

or not. Based on the definition of characterization vectors of bipartite graph, therefore, how heavy a bipartite subgraph is can be measured by the quadratic function in the bipartite subgraph $S(\mathcal{O}', \mathcal{A}', W')$ as follows,

$$L_W(\mathbf{x}^S, \mathbf{y}^S) = (\mathbf{x}^S)^T W (\mathbf{y}^S) = \sum_{i=1}^m \sum_{j=1}^n w_{ij} x_i^S y_j^S \quad (\text{Eq. 5.2})$$

This is an extended version of *Lagrangian* function L_W from graph to bipartite graph. The straightforward meaning of $L_W(\mathbf{x}, \mathbf{y})$ is the sum of edge weights within a bipartite subgraph characterized by $\mathbf{x}$ and $\mathbf{y}$. We called $L_W(\mathbf{x}, \mathbf{y})$ the *generalized Lagrangian* of bipartite graph G . However, we can clearly see that if there are no constraints on $\mathbf{x}^S$ and $\mathbf{y}^S$, $L_W(\mathbf{x}^S, \mathbf{y}^S)$ will be maximum when $\mathbf{x}^S = \mathbf{y}^S = \mathbf{1}$ ($\mathbf{1}$ is a vector whose components are all 1). Correspondingly, like constraint used by Motzkin-Straus, we used $\sum_{i=1}^m (x_i^S)^2 = 1$ and $\sum_{i=1}^n (y_i^S)^2 = 1$ as the constraints³ on $\mathbf{x}^S$ and $\mathbf{y}^S$. This means that $\mathbf{x}^S$ and $\mathbf{y}^S$ are subject to $\mathbf{x}^S \in \Delta_m$ and $\mathbf{y}^S \in \Delta_n$ respectively, where Δ_n is a quadratic surface in n -dimensional Euclidean space,

$$\Delta_n = \left\{ \mathbf{x} \in \mathbb{R}^n \mid \sum_{i=1}^n x_i^2 = 1, \text{ and } x_i \geq 0 \ (i = 1, 2, \dots, n) \right\} \quad (\text{Eq. 5.3})$$

To find the heavy bipartite subgraph, like the problem formulation given by Motzkin and Straus, we gave the following problem in the bipartite graph,

$$(\text{Basic Terrain}) \quad \arg \max_{\mathbf{x} \in \mathbb{R}_+^m, \mathbf{y} \in \mathbb{R}_+^n} L_W(\mathbf{x}, \mathbf{y}) \quad \text{s.t.} \quad \sum_{i=1}^m x_i^2 = 1 \text{ and } \sum_{i=1}^n y_i^2 = 1 \quad (\text{Eq. 5.4})$$

where $\mathbb{R}_+^n$ denotes the n -dimensional non-negative vector space. After obtaining the solution $\mathbf{x}^*$ and $\mathbf{y}^*$ of (Basic Terrain), we sort components of $\mathbf{x}^*$ and $\mathbf{y}^*$ respectively in decreasing order. From the definition of characterization vectors $\mathbf{x}^*$ and $\mathbf{y}^*$ of bipartite subgraph, we can see that vertices with higher associated values in $\mathbf{x}^*$ or $\mathbf{y}^*$ means they have larger possibility of belonging to the heavy subgraph and contribute more to heavy subgraph (or ‘mountain’). As a result, these vertex-associated values can be viewed as the ranking values of vertices in terms of belonging to ‘mountain’. In this way of solving the problem formulated in Eq. 5.4, we can approach the basic terrain

³In Section 5.2.2, we shall show how these constraints (sum of squared components of a vector) help maximize the generalized Lagrangian.

(distinct ‘mountain’) in high-dimensional data by reordering the matrix and collecting high values to the left-top corner of the reordered matrix. In subsequent sections, we shall give a model to approach Eq. 5.4, then theoretically and empirically show the rationale of this problem formulation.

5.2 Discovering Basic Terrain of Data

To efficiently approach the problem formulated in Eq. 5.4, we introduced a novel model inspired by the physics of resonance, called *resonance model*. The underlying rationale of the resonance model is a mechanism with two essential processes: (1) the resonance process between an outer adjustable object and a group of real objects; (2) reinforcement learning procedure. By simulating the principal of a resonance phenomenon, the resonance process tests whether two objects resonate or not and therefore efficiently reveals if there exists good-quality ‘mountain’ within two objects when aligning their attributes (in resonance phenomenon, attributes of an object are called frequencies). We can simulate a resonance experiment by injecting a response function to the resonance test process. For obtaining a good-quality ‘mountain’ among a set of real objects, we simulate resonance phenomenon again, by introducing a forcing object outside of this set of real objects to the model. It is used for iteratively doing resonance test process with each of real objects for “feeling” how high frequencies are distributed among them. Since this forcing object is virtual, its frequencies can be adjusted according to the “feeling” of a round of resonance test processes with each real object. To further precisely “feel” where ‘mountain’ is by using the forcing object updated, next round of resonance test processes and subsequent adjustment of the forcing object are made. Therefore, with a couple of iterations, the resonance model is actually a reinforcement learning procedure. At last, when the forcing object does not change any more, our theoretical result shows that high frequencies in the forcing object will also exist in those real objects who highly resonates with the forcing object. In this way, the resonance model elicits real objects of interest to the analyst automatically, and certainly the *generalized Lagrangian* of G in Eq. 5.2 arrives at its maximum.

In sections that follow, we present the model and discuss its properties and support practicality of the model by discussing how its relationship with some real-world

applications. As this is the basic model to reveal the basic terrain in high-dimensional data, we will further extend this model to approach a more complex terrain and show how it is applied to visualization of high-dimensional data and helps improve analysis in an application: differentially expressed gene selection.

5.2.1 Resonance Model

To simulate a resonance phenomenon, we require a forcing object $\tilde{o}$, such that when an appropriate response function $\mathbf{r}$ is applied, $\tilde{o}$ will resonate to elicit those real objects $\{o_i, \dots\} \subset \mathcal{O}$ in G , whose ‘natural frequency’ is similar to $\tilde{o}$. This ‘natural frequency’ represents the characteristics of both $\tilde{o}$ and the objects $\{o_i, \dots\}$ who resonated with $\tilde{o}$ when $\mathbf{r}$ was applied. For the weighted bipartite graph $G = (\mathcal{O}, \mathcal{A}, E, W)$ and $W = (w_{ij})_{|\mathcal{O}| \times |\mathcal{A}|}$, this ‘natural frequency’ of $o_i \in \mathcal{O}$ is $\mathbf{o}_i = (w_{i1}, w_{i2}, \dots, w_{i|\mathcal{A}|})$. Likewise, the ‘virtual frequency’ of the forcing object $\tilde{o}$ is defined as $\tilde{\mathbf{o}} = (\tilde{w}_1, \tilde{w}_2, \dots, \tilde{w}_{|\mathcal{A}|})$.

Simply put, if two objects of the same ‘natural frequency’ will resonate, they should have a similar frequency (i.e., high values are located in the same set of attributes). The evaluation of resonance strength between objects o_i and o_j is given by the response function $\mathbf{r}(\mathbf{o}_i, \mathbf{o}_j) : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$. We defined this function abstractly to support different measures of resonance strength. For example, one existing measure to compare two terrains is the well-known *rearrangement inequality theorem*, where $\mathbf{I}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n x_i y_i$ is maximized when the two positive sequences $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$ are ordered in the same way (i.e. $x_1 \geq x_2 \geq \dots \geq x_n$ and $y_1 \geq y_2 \geq \dots \geq y_n$) and is minimized when they are ordered in the opposite way (i.e. $x_1 \geq x_2 \geq \dots \geq x_n$ and $y_1 \leq y_2 \leq \dots \leq y_n$). Therefore, the response function $\mathbf{I}$ is a suitable candidate to characterize the similarity of two objects’ frequencies. Likewise, $\mathbf{E}(\mathbf{x}, \mathbf{y}) = \exp(\sum_{i=1}^n x_i y_i)$ is also an effective response function.

To find the ‘mountains’, the forcing object $\tilde{o}$ evaluates the resonance strength of every objects o_i against itself to locate a ‘best fit’ based on the distribution of its frequency. By running this iteratively, those objects that resonated with $\tilde{o}$ are discovered and placed together to form the ‘mountains’ within the 2-dimensional matrix W . This iterative learning process between $\tilde{o}$ and G is outlined below.

Initialization Set up $\tilde{o}$ with a uniform distribution: $\tilde{\mathbf{o}} = (1, 1, \dots, 1)$; normalize it as $\tilde{\mathbf{o}} = \text{norm}(\tilde{\mathbf{o}})^4$; then let $k = 0$; and record this as $\tilde{\mathbf{o}}^{(0)} = \tilde{\mathbf{o}}$.

⁴ $\text{norm}(\mathbf{x}) = \mathbf{x} / \|\mathbf{x}\|_2$, where $\|\mathbf{x}\|_2 = (\sum_{i=1}^n x_i^2)^{1/2}$ is 2-norm of vector $\mathbf{x} = (x_1, \dots, x_n)$.

Apply Response Function For each object $o_i \in \mathcal{O}$, compute the resonance strength $\mathbf{r}(\tilde{\mathbf{o}}, \mathbf{o}_i)$; store the results in a vector $\mathbf{r} = (\mathbf{r}(\tilde{\mathbf{o}}, \mathbf{o}_1), \mathbf{r}(\tilde{\mathbf{o}}, \mathbf{o}_2), \dots, \mathbf{r}(\tilde{\mathbf{o}}, \mathbf{o}_{|\mathcal{O}|}))$; and then normalize it, i.e., $\mathbf{r} = \text{norm}(\mathbf{r})$.

Adjust Forcing Object Using $\mathbf{r}$ from the previous step, adjust the frequency of $\tilde{o}$ for all $o_i \in \mathcal{O}$. To do this, we define the adjustment function $\mathbf{c}(\mathbf{r}, \mathbf{a}_j) : \mathbb{R}^{|\mathcal{O}|} \times \mathbb{R}^{|\mathcal{O}|} \rightarrow \mathbb{R}$, where the weights of the j -th attribute are given in $\mathbf{a}_j = (w_{1j}, w_{2j}, \dots, w_{|\mathcal{O}|j})$. For each attribute a_j , $\tilde{w}_j = \mathbf{c}(\mathbf{r}, \mathbf{a}_j)$ integrates the weights from $\mathbf{a}_j$ into $\tilde{o}$ by evaluating the resonance strength recorded in $\mathbf{r}$. Again, $\mathbf{c}$ is abstract, and can be materialized using the inner product $\mathbf{c}(\mathbf{r}, \mathbf{a}_j) = \mathbf{r} \bullet \mathbf{a}_j = \sum_i w_{ij} \cdot \mathbf{r}(\tilde{\mathbf{o}}, \mathbf{o}_i)$. Finally, we compute $\tilde{\mathbf{o}} = \text{norm}(\tilde{\mathbf{o}})$ and record it as $\tilde{\mathbf{o}}^{(k+1)} = \tilde{\mathbf{o}}$.

Test Convergence Compare $\tilde{\mathbf{o}}^{(k+1)}$ against $\tilde{\mathbf{o}}^{(k)}$. If the result converges, go to the next step; else apply $\mathbf{r}$ on $\mathcal{O}$ again (i.e., forcing resonance), and then adjust $\tilde{o}$.

Matrix Reordering: *Basic Terrain* Sort the objects $o_i \in \mathcal{O}$ by the components of $\mathbf{r}$ in descending order; and sort the attributes $a_i \in \mathcal{A}$ by the components of $\tilde{\mathbf{o}}$ in descending order.

In this process, both vectors $\mathbf{r}$ and $\tilde{\mathbf{o}}$ satisfy the constraint in Eq. 5.3 after the function norm is applied. We denote the resonance model as $R(\mathcal{O}, \mathcal{A}, W, \mathbf{r}, \mathbf{c})$, where the instances of functions $\mathbf{r}$ and $\mathbf{c}$ can be either $\mathbf{I}$ or $\mathbf{E}$. Interestingly, the HITS algorithm [Kle99] is an instance $R(\mathcal{O}, \mathcal{A}, W, \mathbf{r}, \mathbf{c})$ of the resonance model when W is the adjacency matrix of a directed graph and $\mathbf{r} = \mathbf{c} = \mathbf{I}$. Therefore, the resonance model is a generalization of the framework in the HITS algorithm. Please note that it is different from HITS in three ways: (i) the objective of our model is to obtain a basic terrain, i.e., collecting high values to the left-top corner of the sorted matrix, while HITS is designed for Web IR and looks at only the top k authoritative Web pages; (ii) the implementation is different by the virtue that our model is able to make use of non-linear functions, i.e., $\mathbf{r} = \mathbf{c} = \mathbf{E}$, for discovering data terrains while HITS is strictly linear. Moreover, the resonance model can be extended (as presented in Section 5.3) to fit more complex terrain in high-dimensional data, that can not be obtained by HITS; and (iii) we study a set of properties and functions that are quite different from HITS, e.g., *heavy bipartite subgraph*, and the data terrain.

Figure 5.3: Architecture of Resonance model

5.2.2 Properties of the Model

Important properties of our model are discussed in this section. In particular, we show that the model theoretically approximate the basic terrain very well, and that its iterative process converges quickly.

Optimization Theorem of Resonance Model on Basic Terrain

The abstract architecture of the basic resonance model is given in Figure 5.3. Depending on the response and adjustment function, the abstract model instantiates into different implementations. In practice, we have the linear model $R(\mathcal{O}, \mathcal{A}, W, \mathbf{I}, \mathbf{I})$, and the non-linear model $R(\mathcal{O}, \mathcal{A}, W, \mathbf{E}, \mathbf{E})$. The following theorem shows how the resonance model approaches the basic terrain formulated in Eq. 5.4.

Theorem 5.2 *Given a matrix $W_{m \times n}$, the linear resonance model $R(\mathcal{O}, \mathcal{A}, W, \mathbf{I}, \mathbf{I})$ converges to $\mathbf{r}^*$ and $\tilde{\mathbf{o}}^*$, which is the maximizer of $L_W(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T W \mathbf{y}$ with constraints $\sum_{i=1}^m x_i^2 = 1$ and $\sum_{i=1}^n y_i^2 = 1$ when letting $\mathbf{x} = \mathbf{r}^*$ and $\mathbf{y} = \tilde{\mathbf{o}}^*$.*

Proof: For clearly stating the iterative process of the basic resonance model, we express it in the following formulas,

$$\mathbf{r}^{(k+1)} = \text{norm}(\mathbf{r}(W\tilde{\mathbf{o}}^{(k)})) \quad (\text{Eq. 5.5})$$

$$\tilde{\mathbf{o}}^{(k+1)} = \text{norm}(\mathbf{c}(W^T \mathbf{r}^{(k+1)})) \quad (\text{Eq. 5.6})$$

where Eq. 5.5 corresponds to the step “**Apply Response Function**” and Eq. 5.6 is the step “**Adjust Forcing Object**”. Let $\mathbf{r} = \mathbf{c} = \mathbf{I}$ and thus we have the following equations,

$$\mathbf{r}^{(k+1)} = \text{norm}(W\tilde{\mathbf{o}}^{(k)}) \quad (\text{Eq. 5.7})$$

$$\tilde{\mathbf{o}}^{(k+1)} = \text{norm}(W^T \mathbf{r}^{(k+1)}) \quad (\text{Eq. 5.8})$$

According to the standard result of linear algebra [GL96], $\mathbf{r}^{(k)}$ and $\tilde{\mathbf{o}}^{(k)}$ converges to the principal left and right singular vectors of W , respectively (the principal left and right singular vectors are the corresponding vectors of the largest singular value). From the lemma ⁵ given in [GL96], we arrive at the conclusion that the converged $\mathbf{r}^*$ and $\tilde{\mathbf{o}}^*$ is the maximizer of the generalized Lagrangian function.

Convergence and Complexity of Resonance Model

In practice, it is essential that the model can converge quickly to be efficient. Since the resonance model is iterative, we need a test to judge when to stop the iteration for convergence. The definition of generic terrain has indicated that we only care of the ranking positions of objects and attributes, not their associated values. In the resonance model, we can judge the convergence by investigating if the ranking positions of objects and attributes change or not. As a type of *discrete dynamical system* [San90], we have tested the performance of the resonance model and the results showed that it can quickly converge in very large matrix. When using the exponential function $\mathbf{r} = \mathbf{c} = \mathbf{E}$, the convergence has been empirically proved in many large data sets and is much faster.

Therefore, each iteration of learning is bounded by $\mathbf{O}(|\mathcal{O}| \times t_r + |\mathcal{A}| \times t_c)$, where t_r and t_c is the runtime of the response function $\mathbf{r}$, and the adjustment function $\mathbf{c}$ respectively. With k iterations, the final complexity is $\mathbf{O}(k \times (|\mathcal{O}| \times t_r + |\mathcal{A}| \times t_c))$. Since the complexity of $\mathbf{r}$ is $\mathbf{O}(|\mathcal{O}|)$ and $\mathbf{c}$ is $\mathbf{O}(|\mathcal{A}|)$, we have $\mathbf{O}(k \times |\mathcal{O}| \times |\mathcal{A}|)$. In our experiments, our model converges within 50 iterations even on the non-linear configurations giving a time complexity of $\mathbf{O}(|\mathcal{O}| \times |\mathcal{A}|)$. In all cases, the complexity is sufficiently low to efficiency handle large data sets.

5

Lemma 5.1 *Let $W \in \mathbb{R}^{m \times n}$ be a real matrix and σ_{max} be the largest singular value of W , then we have,*

$$\sigma_{max}(W) = \max_{\mathbf{x} \in \mathbb{R}^m, \mathbf{y} \in \mathbb{R}^n} \frac{\mathbf{x}^T W \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} = \max_{\|\mathbf{x}\|_2=1, \|\mathbf{y}\|_2=1} \mathbf{x}^T W \mathbf{y} \quad (\text{Eq. 5.9})$$

and the left and right singular vectors $\mathbf{x}^$ and $\mathbf{y}^*$ corresponding to σ_{max} is the solution on which the maximum is attained.*

5.2.3 Relationships between Basic Terrain and Typical Patterns

In this section, we will show the relationships between basic terrain and two typical patterns of data mining: frequent itemsets and biclusters. These relationships revealed by our study show that basic terrain is closely related to them and can visually analyzing the data for helping discover frequent itemsets and biclusters. In next section, two real-world examples will be shown how the basic data terrain help the user visually analyze high-dimensional data.

Frequent Itemsets

A transaction data set can be represented as a matrix, where each transaction is an object, and each item is an attribute whose value w_{ij} in $W_{|\mathcal{O}| \times |\mathcal{A}|}$ is 1 if the j -th item occurs in the i -th record, and 0 otherwise. Therefore, a transaction data set can be viewed as unweighted bipartite graph. Correspondingly, we have the following that relates frequent itemsets and complete bipartite subgraph.

Theorem 5.3 (Frequent Itemsets and Complete Bipartite Subgraph) *A frequent itemset is the attribute vertex set of a complete bipartite subgraph, and its support is the number of object vertices in the bipartite subgraph.*

Proof: It can be proved that when a frequent itemset C occurs in n transactions, a submatrix W' with all values 1 can be constructed whose attribute set is C , and object set is the set T of n transactions. Hence, W' corresponds to a complete bipartite subgraph whose two disjoint vertex sets are C and T .

To empirically show the relationship between basic terrain and frequent itemsets. We conducted the experiment on a real-world data set with three steps: (1) firstly obtaining the basic terrain of this data; (2) then selecting the itemsets and transactions occurring in left-top ‘*mountain*’ of the terrain to form a small data set; (3) applying frequent itemset algorithm on both complete data set and small data set and comparing the frequent itemsets discovered in two data sets to see if the results are similar. This experiment indirectly proves that ‘*mountain*’ in the reordered matrix is a collection of most frequent itemsets.

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

Table 5.1: Comparison of frequent itemsets found in D and D' with the same minimum support for the **bms-pos** data set. l is the length of an itemset; $\#FI_D^l$ is the number of frequent itemsets with length l in D ; $\#FI_D$ is the total number of frequent itemsets, i.e., including all lengths in D .

	Min. Sup.=516			Min. Sup.=1032			Min. Sup.=5156		
l	$\#(FI_{D'}^l \cap FI_D^l)$	$\#FI_D^l$	$\frac{\#(FI_{D'}^l \cap FI_D^l)}{\#FI_D^l}$	$\#(FI_{D'}^l \cap FI_D^l)$	$\#FI_D^l$	$\frac{\#(FI_{D'}^l \cap FI_D^l)}{\#FI_D^l}$	$\#(FI_{D'}^l \cap FI_D^l)$	$\#FI_D^l$	$\frac{\#(FI_{D'}^l \cap FI_D^l)}{\#FI_D^l}$
1	166	439	37.8%	164	333	49.3%	41	141	29.1%
2	3784	6877	55.0%	1843	3292	56.0%	174	389	44.7%
3	16885	23993	70.4%	5535	8083	68.5%	229	410	55.9%
4	31438	37649	83.5%	7414	9093	81.5%	102	151	67.6%
5	30191	32566	92.7%	4825	5268	91.6%	8	8	100%
6	15829	16197	97.7%	1391	1434	97.0%			
7	4242	4262	99.5%	137	137	100%			
8	455	455	100%	1	1	100%			
9	11	11	100%						
Conclusion	$\#(FI_{D'} \cap FI_D)=103001$ $\#FI_{D'}=103001$ $\#FI_D=122449$			$\#(FI_{D'} \cap FI_D)=21310$ $\#FI_{D'}=21310$ $\#FI_D=27641$			$\#(FI_{D'} \cap FI_D)=554$ $\#FI_{D'}=554$ $\#FI_D=1099$		

We used the **bms-pos** data set (<http://fimi.cs.helsinki.fi/fimi03>) in the experiment. The data set can be modelled as $\mathcal{D} = (\mathcal{O}, \mathcal{A})$, where $\mathcal{O}$ are transactions and $\mathcal{A}$ are the items; we have $|\mathcal{O}|=515,597$ and $|\mathcal{A}|=1,657$. Using $R(\mathcal{O}, \mathcal{A}, W, I, I)$ on $\mathcal{D}$, we obtain its basic terrain and two ordered sequences of items and transactions. According to basic terrain, ‘*mountain*’ is located in the first ranked items and transactions. Therefore, we conclude that we can find a similar set of frequent itemsets from the first ranked fewer transactions and items. To verify it, we selected a subset $\mathcal{D}' = (\mathcal{O}', \mathcal{A}')$, where $\mathcal{O}'$ contains the first $0.1|\mathcal{O}| = 51,560$ transactions and $\mathcal{A}'$ contains the first $0.1|\mathcal{A}| = 166$ items from the terrain of $\mathcal{D}$. We then compare the results using the Apriori algorithm on both $\mathcal{D}$ and $\mathcal{D}'$ using different absolute support thresholds of 516, 1032 and 5156. The results are given in Table 5.1.

From the results, the ‘hit rate’ of finding the frequent itemset in $\mathcal{D}'$ improves quickly as the length of the itemset increases. Nearly all long frequent itemsets in $\mathcal{D}$ can be found in $\mathcal{D}'$, e.g., at minimum support of 516 and $\ell = 5, 6, 7$, we discovered over 90% of the frequent patterns; and when $\ell = 8, 9$, we find all the patterns in $\mathcal{D}'$. This result is encouraging since long frequent itemsets are often more important and interesting in real-life applications.

Biclusters

Bicluster is one of popular patterns in Bioinformatics and data mining introduced in recent years [MO04]. The objective of biclustering algorithms is to find a subset of objects and attributes whose corresponding submatrix is coherent (i.e., values from

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

any two objects in object subset are up and down in the attribute subset). The first and most popular measure, “mean squared residue score” – $H(W)$, of biclusters was proposed by Cheng and Church [CC00]. It is defined as

$$H(W) = \frac{1}{mn} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} (w_{ij} - w_{iJ} - w_{Ij} + |W|)^2 \quad (\text{Eq. 5.10})$$

where $w_{iJ} = \frac{1}{n} \sum_{j=1}^n w_{ij}$, $w_{Ij} = \frac{1}{m} \sum_{i=1}^m w_{ij}$, and $|W| = \frac{1}{mn} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}$. From Eq. 5.10, the lower $H(W)$, the better W as a bicluster. Therefore, Given a matrix W , it is called a δ -bicluster if $H(W) \leq \delta$.

The following theorem reveals the relationship between bicluster and average resonance strength in resonance model for basic terrain.

Theorem 5.4 (Bicluster and Average Resonance Strength of Basic Terrain)

Given a matrix $W = (w_{ij})_{m \times n}$, where $\mathcal{O}$ are the rows and $\mathcal{A}$ are columns, we have the inverse relation of the average inter-resonance strength and $H(W)$ as follows

$$H(W) = \|W\|^2 + |W|^2 - \frac{1}{n} \bar{r}(W) - \frac{1}{m} \bar{r}(W^T) \quad (\text{Eq. 5.11})$$

where $\|W\| = \sqrt{\frac{1}{mn} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2}$, $\bar{r}(W) = \frac{1}{\binom{m}{2}} \sum_{\substack{1 \leq i, j \leq m \\ i \neq j}} I(\mathbf{w}_{i-}, \mathbf{w}_{j-})$ is the average inter-resonance strength among $\mathbf{w}_{i-}$, and $\bar{r}(W^T) = \frac{1}{\binom{n}{2}} \sum_{\substack{1 \leq i, j \leq n \\ i \neq j}} I(\mathbf{w}_{-i}, \mathbf{w}_{-j})$ is the average inter-resonance strength among $\mathbf{w}_{-j}$, and $\mathbf{w}_{i-}$ is the i -row vector of W with $\mathbf{w}_{-j}$ the j -column vector of W .

Proof: For simplicity, suppose

$$w_{iJ} = \frac{1}{n} \sum_{j=1}^n w_{ij}, \quad w_{Ij} = \frac{1}{m} \sum_{i=1}^m w_{ij}$$

and by the definition of “mean squared residue score” in Eq. 5.10, we have

$$\begin{aligned} H(W) &= \frac{1}{mn} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} (w_{ij} - w_{iJ} - w_{Ij} + |W|)^2 \\ &= \frac{1}{mn} \left(\sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2 + mn|W|^2 - n \sum_{i=1}^m w_{iJ}^2 - m \sum_{j=1}^n w_{Ij}^2 \right) \end{aligned} \quad (\text{Eq. 5.12})$$

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

because

$$\sum_{i=1}^m w_{iJ}^2 = \frac{1}{n^2} \left(\sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2 + 2 \binom{n}{2} \bar{\mathbf{r}}(W^T) \right) \quad (\text{Eq. 5.13})$$

and

$$\sum_{j=1}^n w_{Ij}^2 = \frac{1}{m^2} \left(\sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2 + 2 \binom{m}{2} \bar{\mathbf{r}}(W) \right) \quad (\text{Eq. 5.14})$$

and by Eq. 5.12, Eq. 5.13 and Eq. 5.14, we get

$$\begin{aligned} \mathbf{H}(W) &= \frac{(m-1)(n-1)-1}{m^2 n^2} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2 + |W|^2 \\ &\quad - \frac{2 \binom{m}{2}}{n m^2} \bar{\mathbf{r}}(W) - \frac{2 \binom{n}{2}}{m n^2} \bar{\mathbf{r}}(W^T) \end{aligned} \quad (\text{Eq. 5.15})$$

and since we assume m and n are sufficiently large,

$$\frac{1}{mn} \approx \frac{(m-1)(n-1)-1}{m^2 n^2}, \quad \binom{m}{2} \approx \frac{m^2}{2}, \quad \binom{n}{2} \approx \frac{n^2}{2}$$

we can conclude

$$\begin{aligned} \mathbf{H}(W) &= \frac{1}{mn} \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} w_{ij}^2 + |W|^2 - \frac{1}{n} \bar{\mathbf{r}}(W) - \frac{1}{m} \bar{\mathbf{r}}(W^T) \\ &= \|W\|^2 + |W|^2 - \frac{1}{n} \bar{\mathbf{r}}(W) - \frac{1}{m} \bar{\mathbf{r}}(W^T) \end{aligned}$$

Theorem 5.4 can be interpreted as follows. Since $\|W\|$ and $|W|$ are sum of W in different forms, we can considered them as fixed constant. If the average inter-resonance strength of W and W^T , i.e., $\bar{\mathbf{r}}(W)$ and $\bar{\mathbf{r}}(W^T)$, is higher, then $\mathbf{H}(W)$ is lower and thus, W behaves like a better bicluster. For $\mathbf{R}(\mathcal{O}, \mathcal{A}, W, \mathbf{I}, \mathbf{I})$ and $\mathbf{R}(\mathcal{A}, \mathcal{O}, W^T, \mathbf{I}, \mathbf{I})$, we conclude that if we select the first k rows and columns of W with large resonance strength $\mathbf{r}(\mathbf{o}_i, \tilde{\mathbf{o}})$ to form W' , it is straightforward that we will have a smaller $\mathbf{H}(W')$ and thus, W a bicluster.

Because the objects and attributes ranked higher are those with larger resonance strengths r_i^* , W' formed must have large average inter-resonance strength $\bar{\mathbf{r}}(W')$ and $\bar{\mathbf{r}}(W'^T)$. Hence, $\mathbf{H}(W')$ is sufficiently low to be a bicluster. Empirically, we verified this conclusion using the yeast microarray data (<http://arep.med.harvard.edu/>

(a) overlapping genes in B_k and G_{800}

	G_{1000}	G_{800}	G_{600}
F_1	0.66	0.70	0.74
P	0.87	0.80	0.73
R	0.53	0.62	0.75

$$P = \frac{|B_1 \cup G_k|}{|B_1|}, \quad R = \frac{|B_1 \cup G_k|}{|G_k|}, \quad F_1 = \frac{2PR}{P+R}$$

(b) overlapping genes in B_1 and G_k

Figure 5.4: Comparison of genes in G_k and biclusters B_k ($1 \leq k \leq 100$) for gene expression data

`network.discovery`) containing 2,884 genes and 17 conditions. From [CC00], 100 biclusters B_k ($1 \leq k \leq 100$) are found. As the number of conditions is small, we only investigate the effects of $R(\mathcal{O}, \mathcal{A}, W, \mathbf{I}, \mathbf{I})$ on genes, where $\mathcal{O}$ are genes, $\mathcal{A}$ are conditions, and W is a row-normalized matrix. After the resonance simulation on W , we consider the subset G_k ($k \leq 2884$) which contains the first k genes with the largest resonance strength. We then compare how the genes in the 100 biclusters B_k ($1 \leq k \leq 100$) overlap with those in G_k . Figure 5.4 verified our conclusions about biclusters.

5.2.4 Two Real-World Examples for Visually Analyzing Data With Basic Terrain

We shall analyze two data sets to illustrate how we apply visual basic terrain analysis to high-dimensional data and demonstrate the insights uncovered through the visualization of the basic terrain. The first is the Wisconsin breast cancer data set (from the UCI Repository of Machine Learning Databases) with 699 instances, 9 attributes (each rated from 1 to 10), and two classes: benign (458 instances) and malignant (241 instances) [MW90]. This data set is often used in testing outlier detection algorithms. Following the setup by He *et al.* [HXD03], we removed some of the malignant records to form a test data set with small number of outliers. The test data set has 39 (8%) malignant records (as outliers) and 444 (92%) benign records.

In our scenario, the users may use our model to obtain the terrain so as to decide where the outliers are. Here, we used the non-linear configuration $R(\mathcal{O}, \mathcal{A}, W, \mathbf{E}, \mathbf{E})$ to

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

Figure 5.5: Visualization of Wisconsin breast cancer data and comparison with outlier methods

visualize the data terrain as shown in Figure 5.5(a). By inspection, we see that only a few records (within the first 100) have attributes with large values, whereas the other records don't. Immediately, the user would be able to easily recognize the outliers' distribution and property. That is, the user is able to conclude that the *outliers are records whose attributes have high values*. Therefore, the user may choose to remove them by selecting the top k records in this terrain. Figure 5.5(b) shows the results of our approach against well-known outlier detection algorithms adapted from [HXD03]. While the results speak for itself, it is worth to note that our approach allows the users to detect the outliers without much domain knowledge of the data.

The MovieLens data set is our second data set that is collected by the GroupsLen research project at the University of Minnesota (<http://www.grouplens.org>). It contains 943 users, 1,682 movies and 100,000 ratings (on the scale of 1 to 5), and each user is required to rate at least 20 movies. If the user is not interested in some movies, they will not rate them and this is recorded as a 0.

Again, we obtain the basic terrain using $R(\mathcal{O}, \mathcal{A}, W, I, I)$ in Figure 5.6(a). Clearly, this data set is very sparse and most of the ratings are concentrated on a small number of movies and users (i.e., they are collected to the left-top corner of Figure 5.6(a)). If we divide the users and movies on the terrain into 25 equal intervals, as shown in Figure 5.6(a), we have 25 submatrices $(M_1, \dots, M_{25})$ as shown in Figure 5.7(a). We then counted the number of ratings in each unit to get Figure 5.7(b). This 2-dimensional distribution now shows a clearer statistics of rating distribution in the basic terrain. Upon further analysis of Figure 5.7(b), we obtain an exponential curve describing the percentage of size and rating in each submatrix M_i (with size $m_i \times n_i$,

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

(a) gray scale image of overall *basic terrain* (b) gray scale image of local *basic terrain*

Figure 5.6: Visualization of MovieLens data and a local *basic terrain* in detail

(a) partitions of overall *basic terrain* (b) rating distribution of overall *basic terrain* (c) size/rating percents

Figure 5.7: Statistics of overall *basic terrain* of MovieLens data

and number of ratings r_i), whose percentage of size is $\zeta_i = \frac{m_i \times n_i}{943 \times 1682}$, and percentage of ratings is $\eta_i = \frac{r_i}{100000}$. For example, the submatrix M_{15} in Figure 5.7(c) has a ratings of 82.5% while its size is surprisingly only 35.2%.

It is possible that the user is not satisfied with an overview of the data set. Instead, the user might be interested in further analysis. From Figure 5.7, the user may further explore the ‘crowded’ but a region of the terrain, i.e., M_{10} of size 377×672 . To do so, the user may simply apply $R(\mathcal{O}, \mathcal{A}, W, E, E)$ on M_{10} . This allows us to ‘zoom in’ on a specific region on the terrain giving us some further insights to the data as shown in Figure 5.6(b). In this case, the user finds that the first several movies are classics (e.g., “Star Wars”, “Raiders of the Lost Ark”, “Godfather”, etc.) with most of them being old movies, and the first several reviewers have occupations as ‘students’ or

‘educators’. And within this group of reviewers, more than half disliked “Titanic”.

5.3 ‘Mountain-Valley’ Terrain and Application to Gene Selection

Although the *basic terrain* has been shown to have close relationships with typical patterns in Bioinformatics and data mining, and demonstrated how to visually helping analyze high-dimensional data, we have not really applied the concept *terrain* to solving a real-world problem. In this section, we shall focus on a challenging and important real-world problem, i.e. the selection of differentially expressed genes from microarray data (“*gene selection*”), and design a more complex terrain to approach it.

With the rapid advances of microarray technologies, massive amounts of gene expression data are generated in experiments. One of the most important features in microarray data is the very high dimensionality with a small number of samples. There are over thousands of genes and at most several hundred of samples in the data set. Such characteristic that has never existed in any other type of data, has made the traditional data mining and analysis methods not effective, and therefore attracted the focus of recent research. Among these methods, a crucial approach is to select a small portion of informative genes for further analysis, e.g., disease classification. In detail, gene selection is to detect the most significantly differentially expressed genes under different classes of samples. For instance, there are differential expressions across the normal and diseased classes of samples. From perspective of generic terrain, there exist some attributes (or genes) which have very low values in a type of objects (or normal samples), but very high values in another type of objects (or diseased samples). This observance of gene expression data leads to a new but more complex terrain which allows the coexisting of ‘mountain’ and ‘valley’. We called it ‘*mountain-valley*’ terrain.

5.3.1 Background and Related Work

Recent gene selection methods fall into two categories: *filter methods* and *wrapper methods* [TA]. The wrapper methods [CGFW05] are closely “embedded” in the classifier and thus are often time-consuming. On the other hand, the filter methods

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

analyze the data by investigating their domain-specific targets: differentially expression across sample classes. Most existing filter methods follow the methodologies of statistics [JSR03] and information theory [YL04, TA] to rank the genes, such as t-like-statistics, mutual information or information gain based methods. These methods are computationally efficient. However, they select the genes by only considering the binary class labels, e.g. healthy/diseased, while the sample classes in the observed experiments are often ordinal with the gradually changing tendency [CGFW05]. For example, in the Lupus experiment (see Subsection 5.3.4), four classes of persons are considered. They are normal ones, relatives of patient, patients who show the early symptoms, and patients whose symptoms are complete. Also some gene expression experiments⁶ consider classes of samples that are the composite of normal and disease ingredients with different scale, e.g. 1:4 or 3:4. In these gene expression data, although there are two types of classes, i.e. positive and negative, the labels of multiple classes show the ordinal scales according to the degree of their membership to the positive or negative type, e.g. ‘normal’, ‘low-grade’ tumor, ‘intermediate-grade’ tumor and ‘high-grade’ tumor [Se02]. However, when dealing with the data sets with such multiple classes and two types, most existing filtering methods e.g. information gain and t-statistics, combine all classes in positive type into a positive class and similarly combine all classes in negative type into a negative class, and then do the filtering process on the two combined classes. Such analysis may ignore the characteristics of the expression data within each single class, and therefore may lose the accuracy of selecting informative genes. On the other hand, most general feature selection methods, e.g. ReliefF [KR92], consider the multiple classes, but ignore the special characteristic of gene selection, up and down regulations. Therefore, they are not specific to the task of gene selection as well. There have been few works in the wrapper methods on investigating the biomarker on these data sets, such as Gaussian process model based method [CGFW05]. However, it is a wrapper method by using the leave-one-out error and forward selection and therefore is not efficient. Moreover, the wrapper method is like a black-box screening the user out of the analysis process.

Therefore, in this section, we proposed ‘*mountain-valley*’ terrain, a filter method, that takes into account the global between-class data distribution (differentially expression) and local within-class data distribution (collection of low or high values). In this

⁶In MAQC project, the description of the data sets are available at <http://www.fda.gov/nctr/science/centers/toxicoinformatics/maqc>.

way, it simultaneously ranks the genes and samples ⁷ to identify the differentially expressed genes in the data sets with multiple classes. By reordering the gene expression matrix with ‘*mountain-valley*’ terrain, we can visually observe the overall distribution (see Fig.5.10) of the values, where top genes are differentially expressed across classes and top samples are important to the class. Moreover, the sorted matrix according to the ranking of both genes and samples can be visually shown to the user for further analysis. Hence, the ‘*mountain-valley*’ terrain provides an intuitive and visual means to select genes from multiple-class microarray data by overcoming the drawbacks of both existing filter and wrapper methods.

5.3.2 ‘Mountain-Valley’ Terrain of Data

Consider the general case of the gene expression data $W_{m \times n}$ ⁸, suppose the data set consists of m objects (or genes) and n attributes (or samples) with k classes, whose number of samples are $n_1, \dots, n_k$ respectively and $n_1 + \dots + n_k = n$. Without losing the generality, we suppose the first k_- classes are negative, the following k_+ classes are positive, and $k_- + k_+ = k$. Therefore, a general gene-sample matrix $W_{m \times n} = [\underbrace{W_i^-}_{1 \leq i \leq k_-}, \underbrace{W_i^+}_{1 \leq i \leq k_+}]$ is shown with submatrix blocks in Fig.5.8(a).

Because the up and down regulated genes among negative and positive sample classes are the target that biologists are interested in, the ‘*mountain-valley*’ Terrain are defined in two types: (1) ‘*valley-mountain*’ terrain (up regulation) by collecting low values (i.e., ‘*valleys*’) of W_i^- in negative classes to their left-top corner, and simultaneously by collecting high values (i.e. ‘*mountains*’) of W_i^+ in positive classes to their left-top corner. (2) ‘*mountain-valley*’ terrain (down regulation) by collecting high values (i.e., ‘*mountains*’) of W_i^- in negative classes to their left-top corner, and simultaneously by collecting low values (i.e. ‘*valleys*’) of W_i^+ in positive classes to their left-top corner.

⁷The samples are ranked within each class.

⁸In the gene expression data, to make sure W is a non-negative matrix, we scale the values of W to the range of $[0, 1]$ by $\frac{w_{ij} - \min}{\max - \min}$, where min and max can be the minimum and maximum of each rows or of the whole matrix. In the rest of the section, W is supposed to be a matrix whose range is in $[0, 1]$ if we do not mention.

CHAPTER 5. VISUAL TERRAIN ANALYSIS OF HIGH-DIMENSIONAL DATA BY RESONANCE MODEL

Figure 5.8: Transformation of the matrix W : the transformed matrix W' has the same structure of submatrix blocks as shown in (a), but with different submatrix $W_i'^-$ and $W_i'^+$ as listed in (b).

5.3.3 Extended Resonance Model

As the resonance model is a basic means of discovering distinct ‘mountain’ by collecting high values to the left-top corner of reordered matrix, in this section, we shall keep its core iteratively reinforcement learning process but add more information of attribute classes and transformation of matrix for extending it to discover ‘mountain-valley’ terrain. Because the target of analyzing differentially expressed genes is to find up-regulated or down-regulated genes between negative and positive sample classes, the basic resonance model should be changed, from collecting high values to the left-top corner (in the case of ‘valley-mountain’ terrain, i.e., up regulation), to:

- (i) A series of low values collections in each W_i^- into the left-top corner, and simultaneously a series of high values collections in each W_i^+ into the left-top corner.
- (ii) Controlling the differences of left-top corners between the negative classes W_i^- and W_i^+ .

An example figure of such matrix approximation is illustrated in Fig.5.10. Therefore, to meet these two goals, we extended the basic resonance model, called as **ER**, as follows.

- (i) Transformation of W : before doing the ER, we need to transform the original gene-sample matrix W to W' . The structure of W is made of the submatrix blocks W_i^- and W_i^+ of negative classes and positive classes as shown in Fig.5.8(a). In the case of finding up-regulated differentially expressed genes, since we need to collect the low values of W_i^- into the left-top corner, we need to reverse the values of W_i^- so that low values become high and vice versa. In other words, we do the transformation by $W_i'^- = 1 - W_i^-$. In this way, the result of collecting high values of $W_i'^-$ and $W_i'^+$ into their own left-top corners naturally lead to the result of collecting the low values of W_i^- into the left-top corners and the high values of W_i^+ into the left-top corners. This is an essential step to meet the first goal above. We can also use other reverse functions instead of the simple $1 - x$ function used in Fig.5.8(b). Similarly, we can transform W by $W_i'^+ = 1 - W_i^+$ in the case of finding down-regulated differentially expressed genes.
- (ii) The k partitions of the forcing object $\tilde{\mathbf{o}}$: an implicit requirement in the first goal is that the relative order of each class (submatrix $W_i'^-$ or $W_i'^+$) should be kept the same after doing ER and sorting W' . For example, after running our algorithm, it is required that all columns of the submatrix $W_2'^-$ must appear after all columns of $W_1'^-$, although we can change the order of columns or samples within $W_1'^-$ or $W_2'^-$. To satisfy this requirement, we partition the original forcing object's frequency vector $\tilde{\mathbf{o}}$ into k parts corresponding to k classes or submatrices. Specifically, $\tilde{\mathbf{o}} = (\tilde{\mathbf{o}}_1; \dots; \tilde{\mathbf{o}}_k)$ ⁹, where each $\tilde{\mathbf{o}}_i$ corresponds to a sample class. In the process of ER, we separately normalize each $\tilde{\mathbf{o}}_i$ and then sum their resonance strength vectors together with the factor α to control the differentiation between the negative and positive classes.
- (iii) The factor α for controlling the differentiation between the negative and positive classes: the frequency vector of $\tilde{\mathbf{o}}$ is divided into $k = k_- + k_+$ parts, each of which is normalized independently. Therefore, we can control the differentiation between the negative and positive classes, by magnifying the resonance strengths $\mathbf{r}_i^+ = \text{norm}(W_i'^+ \tilde{\mathbf{o}}_i)$ of k_+ positive classes, or minifying the frequency subvectors $\mathbf{r}_i^- = \text{norm}(W_i'^- \tilde{\mathbf{o}}_i)$ of k_- negative classes. In formal,

⁹The concatenation of $k = k_- + k_+$ vectors is expressed in MATLAB format.

Figure 5.9: Architecture of extended resonance model

$$\mathbf{r} = \text{norm} \left(\underbrace{\mathbf{r}_1^- + \dots + \mathbf{r}_{k_-}^-}_{k_- \text{ negative classes}} + \underbrace{\alpha \mathbf{r}_1^+ + \dots + \alpha \mathbf{r}_{k_+}^+}_{k_+ \text{ positive classes}} \right) \quad (\text{Eq. 5.16})$$

where $\alpha \geq 1$ and α as a scaling factor is multiplied with the normalized positive classes' resonance strength vectors. With the increasing of α , the proportions of positive classes in the resonance strength vector $\mathbf{r}$ will increase and thus result in the increasingly large differences in the top-left corners between positive and negative classes. In this way, the user can tune α to get a suitable differential contrast of two types of classes.

To summarize the above changes of the resonance model, we draw the architecture of the ER in Fig.5.9 and express its process in the following formulas:

$$\begin{aligned} \mathbf{r}_i^{-(k+1)} &= \text{norm}(W_i'^- \tilde{\mathbf{o}}_i^{-(k)}), & i = 1, \dots, k^- \\ \mathbf{r}_i^{+(k+1)} &= \text{norm}(W_i'^+ \tilde{\mathbf{o}}_i^{+(k)}), & i = 1, \dots, k^+ \\ \mathbf{r}^{(k+1)} &= \text{norm} \left(\sum_{i=1}^{k^-} \mathbf{r}_i^{-(k+1)} + \alpha \sum_{i=1}^{k^+} \mathbf{r}_i^{+(k+1)} \right) \\ \tilde{\mathbf{o}}_i^{-(k+1)} &= \text{norm}((W_i'^-)^T \mathbf{r}^{(k+1)}), & i = 1, \dots, k^- \\ \tilde{\mathbf{o}}_i^{+(k+1)} &= \text{norm}((W_i'^+)^T \mathbf{r}^{(k+1)}), & i = 1, \dots, k^+ \end{aligned} \quad (\text{Eq. 5.17})$$

where $\mathbf{r}_i, \mathbf{r}_i^+, \mathbf{r}_i^- \in \mathbb{R}^{m \times 1}$ and $\tilde{\mathbf{o}}_i^- \in \mathbb{R}^{n_i^- \times 1}$, $\tilde{\mathbf{o}}_i^+ \in \mathbb{R}^{n_i^+ \times 1}$. Comparing Eq. 5.5, Eq. 5.6 and Eq. 5.17, besides using the linear functions $\mathbf{r} = \mathbf{c} = \mathbf{I}$, we partitioned the matrix W' to k submatrix blocks and divided the frequency vector $\tilde{\mathbf{o}}$ to k subvectors. Therefore, two equations in the basic resonance model is expanded to the $(2k+1)$ equations

Algorithm 2 (ERM): Gene Selection

Input:

- (1) $W_{m \times n}$, expression matrix from m genes set G and n samples set S ;
- (2) $(n_1, \dots, n_k)^T$, sizes of the k sample classes with the submatrix structure as in Fig.5.8(a).
- (3) $(k_-, k_+)^T$, numbers of neg. and pos. classes.
- (4) *regulation option*, down or up;
- (5) α , differentiation factor.

Output:

- (1) $(g_1, \dots, g_m)$, ranking sequence of m genes;
- (2) $(s_1, \dots, s_n)$, ranking sequence of n samples.

- 1: preprocess W so that the values of W in $[0,1]$ as following the steps in subsection 5.3.2.
- 2: transform W to W' according to formulas in Fig. 5.8(b) with the knowledge of the matrix structure given by $(n_1, \dots, n_k)^T$, and $(k_-, k_+)^T$ and *regulation option*.
- 3: iteratively run equations in Eq. 5.17 to obtain the converged $\mathbf{r}^*$ and $\tilde{\mathbf{o}}_i^*$ ($i=1, 2, \dots, k$).
- 4: sort $\mathbf{r}^*$ in decreasing order to get the ranking gene sequence $(g_1, \dots, g_m)$, and sort each of $\tilde{\mathbf{o}}_1^*, \dots, \tilde{\mathbf{o}}_k^*$ in decreasing order to get the sorted sample sequence *{comment: Because the positions of all sample classes in W' keep not changing as shown in Fig.5.8(a), each sorting of $\tilde{\mathbf{o}}_i^*$ can only change the order of samples within the i -th sample class W'_i .}*

in ER. We also formally summarize it as Algorithm 2 ERM for gene selection. A real-life example of the overall process in Algorithm ERM is visually shown in Fig.5.10.

In practice, ERM can quickly converge. Considering ERM is a generalized resonance model by partitioning the matrix into k submatrices, its computational complexity is the same as the resonance model on the whole matrix, i.e. $\mathbf{O}(mn)$.

5.3.4 Empirical Study

In this section, we conducted the experiments on two data sets and compared our method with three most popular filter method, T-statistics (T), Information Gain (IG) and ReliefF [KR92]. We firstly used the ERM¹⁰, T and IG to rank the genes and compared them over different feature sizes, $k=2,4,10,20,50,100,200,500,1000$. Each resulting feature subset was used to train an SVM classifier¹¹ with the linear kernel function. Because of the small number of samples, the Leave-One-Out Cross Validation (LOOCV), a popular performance validation procedure adopted by many researchers, was performed to assess the classification performance.

¹⁰Because ERM can rank genes in terms of up and down regulation respectively, in this experiment of comparing k top-ranking genes, we selected $0.5k$ top-ranking genes in up regulation and $0.5k$ top-ranking genes in down regulation to form k top-ranking genes given by ERM.

¹¹The SVMlight was used.

Table 5.2: LOOCV accuracy rate (%) of the experiment: ALL versus MLL&AML.

$k=$	2	4	10	20	50	100	200	500	1000
T	79.2	86.1	91.7	93.1	98.6	98.6	98.6	100	100
IG	76.4	80.6	95.8	98.6	98.6	98.6	98.6	98.6	98.6
RliefF	63.9	86.1	95.8	95.8	98.6	98.6	100	98.6	98.6
ERM	100	100	98.6	100	100	100	100	100	100

Table 5.3: LOOCV accuracy rate (%) of the experiment: MLL versus ALL&AML.

$k=$	2	4	10	20	50	100	200	500	1000
T	69.4	65.2	81.9	80.6	84.7	86.1	93.1	90.3	87.5
IG	72.2	88.9	88.9	88.9	98.6	98.6	97.2	98.6	97.2
RliefF	72.2	88.9	95.8	94.4	94.4	94.4	97.2	98.6	98.6
ERM	86.0	88.9	97.2	98.6	100	97.2	98.6	98.6	98.6

Classification

We used the Leukemia gene expression data in [Ae02], where besides the classes “ALL” and “AML”, a new class “MLL” of samples is identified. It contains 12,582 genes and 72 samples with these 3 sample classes. Therefore, we performed three experiments to test our method by using one class versus the rest classes as positive versus negative: (1) ALL versus MLL&AML, (2) MLL versus ALL&AML and (3) AML versus ALL&MLL. In each experiment, the gene expression matrix partition for our method is $W = [W^+, W_1^-, W_2^-]$ with one positive and two negative classes. In all three experiments, α was set to 2 for ERM. The results are shown in Table 5.2, 5.3 and 5.4. As shown in the three tables, our method ERM outperforms the other methods in,

- High Accuracy: in all three experiments, ERM maintains very high accuracies in different k . In the experiment “MLL versus ALL&AML”, where the class MLL is hard to distinguish, ERM can still obtain high accuracy even when k is very small.
- Compact genes selected: observing the accuracies of three methods from the small k to the large, ERM is able to quickly obtain the high accuracy in the very small k , while the methods T and IG require larger k to arrive at the same accuracy (the numbers in bold in three tables show the minimum k each

Table 5.4: LOOCV accuracy rate (%) of the experiment: AML versus ALL&MLL.

$k=$	2	4	10	20	50	100	200	500	1000
T	66.7	77.8	97.2	98.6	100	98.6	97.2	97.2	97.2
IG	79.2	76.4	87.5	93.1	97.2	97.2	97.2	97.2	97.2
RliefF	86.1	84.7	95.8	94.4	97.2	97.2	97.2	98.6	97.2
ERM	90.3	91.7	97.2	97.2	97.2	97.2	97.2	97.2	97.2

method requires to get the highest accuracy). This means that **ERM** outperforms the other methods in terms of discovering the compact or minimal genes. For example, in Table 5.2, the top 2 ranking genes are found by **ERM** and their accuracy is 100%, while the accuracy of the other two methods' top 2 ranking genes are less than 80%. Similar cases also appear in Table 5.3 and 5.4.

- Stability: not only do the small amount of selected genes have the higher accuracies than the other methods, but also the large subset of selected genes maintain the high accuracy. This is a stable property with k increasing, and may be interesting to the biologists when they try to analyze more relevant genes contributing to the diseases. In contrast, the method T is not stable, especially in Table 5.3 when the samples are hard to distinguish.

An important factor that enables **ERM** to perform well, is that the extended resonance model equips it a global searching ability to take into account the value distribution of the whole matrix and multiple classes in macroview way. This is different from the way of individually considering genes or samples, or gene-to-gene.

To test if the genes found by our method are effective or not, for instance, we checked two genes found by **ERM** in Table 5.2 with Entrez Gene in NCBI Website (<http://www.ncbi.nlm.nih.gov/entrez>). Two genes are MME which is under-expressed and LGALS1 which is overexpressed ¹². By investigating the result of Armstrong *et al.* [Ae02], these two genes were also ranked as the first genes in the underexpressed and overexpressed genes respectively. MME is a common acute lymphocytic leukemia antigen that is an important cell surface marker in the diagnosis of human acute lymphocytic leukemia (ALL); while LGALS1 was also reported to be highly correlated with ALL [Re03].

¹²The GeneBank No. of MME is J03779 and the GeneBank No. of LGALS1 is AI535946.

Figure 5.10: Visualization of the sorted matrix W , sorted transformation matrix $W' = [1 - W_{\text{NC}}, 1 - W_{\text{FDR}}, W_{\text{ILE}}, W_{\text{SLE}}]$, and sorted approximation matrix $\mathbf{r}^* \tilde{\mathbf{o}}^{*T} \approx W'$, where $\tilde{\mathbf{o}}^*$ is the concatenation of k vectors: $\tilde{\mathbf{o}}^* = (\tilde{\mathbf{o}}_1^*; \dots; \tilde{\mathbf{o}}_k^*)$.

Visualization

In this experiment, we demonstrate the visualization ability of our method for facilitating the user to analyze both the genes and samples simultaneously, although the results (not shown) of the classification accuracy also outperform other methods. We used the unpublished data set taken from the Lupus gene expression experiments of Microarray core facility in UT SouthWestern Medical Center. This data set contains 1,022 genes and 84 samples with 4 sample classes: “NC” (Normal Control), “FDR” (First-Degree Relative), “ILE” (Incomplete Lupus Erythematosus) and “SLE” (Systemic Lupus Erythematosus). Among these classes, “NC” and “FDR” are from the normal persons while “ILE” and “SLE” are from patients.

We performed ERM with $\alpha = 5$ on the data. The sorted matrix W with up regulation setting (see Fig.5.8(b)) is visualized by the color image in Fig.5.10(a). From this redistribution of the whole matrix, the dominant tendency within each class can be clearly observed. While the most differentially expressed genes (or rows) are placed in the top of W , the low values of the first two classes “NC” and “FDR” are collected to the left-top corner of each submatrix W_{NC} and W_{FDR} , and the high values of the first two classes “ILE” and “SLE” are collected to the left-top corner of each submatrix W_{ILE} and W_{SLE} . In this way, the data within-class distributions and the between-class distribution are fully considered. To illustrate the process of ERM, we

also drew the color image of the transformed matrix W' for up regulation and the final approximation matrix $\mathbf{r}^* \tilde{\mathbf{o}}^{*T}$ given by the converged resonance strength vector $\mathbf{r}^*$ and the frequency distribution vector $\tilde{\mathbf{o}}$.

By observing the color image of approximation matrix $\mathbf{r}^* \tilde{\mathbf{o}}^{*T}$ in Fig.5.10(c), we found that the outlier samples of each class are put in the rightmost place of the corresponding class submatrix. For example, the colors of the rightmost sample (the 13-th column) in the class “NC” are significantly different from the colors of all other left samples, which indicates that this sample may be an outlier of the class “NC”. This can also be observed in Fig.5.10(a) of the original sorted gene expression matrix. After analyzing this visualization, besides obtaining the top-ranking relevant genes, the user can also draw the conclusion that some normal persons may be early-stage, undetected patients. Similar cases occur in the other classes as well.

5.4 Summary

Finally, we can compare our work to the study of statistics of data in 1-dimensional spaces. Well-known distributions such as the power-law (or Zipf’s law) has been found to be ubiquitous in most real-world data set [LY02, FFF99, BAJB00, BA99], and ever since has been the hallmark of complex systems [BA99, CD99]. As a matter of fact, the power-law distribution indicates data skew that can be exploited for data analysis. Interestingly, power-law has not been exploited by large in the domain of data engineering [FMS96, BFK01, PF00]. In this chapter, the power-law distribution actually relates to the reordering of data in one dimension¹³. In this case, our work can be seen as an extension of the power-law to reorder data in 2-dimensions.

In this chapter, we proposed the data terrain as a means to visualize and analyze high-dimensional data sets. With this terrain, patterns in high-dimensional space can be visualized and analyzed intuitively and visually. From the perspective of information visualization and data mining, this is a novel contribution. In addition to introducing the concept of data terrain, we gave a case study by applying the visual terrain analysis to the problem of biomarker discovery and therefore provided an empirical evidences of the feasibility of our proposal by means of this real-world application.

¹³Joint distributions (e.g., normal distribution) are not considered as they seldom characterize the real-world data set, and are not suitable for high-dimensional data sets.

Chapter 6

Conclusions

This century is surely the century of data, particularly the data with high dimensionality and complex relationships. One of the most challenging problems in the analysis of such data sets is the combinatorial explosion of patterns found in high-dimensional space or complex graphs, and the inefficiency of traditional data analysis techniques. The case is much worse when the size of a data set is very large. In order to attack difficulties of mining data with such complex properties, we argue that the deep understanding of data characteristics is important. Without previewing and feeling how an data set is, it is often blind for users to run algorithms on the data. One of the immediate consequences is how to tune parameters of the algorithm to obtain the satisfactory results. The ad-hoc algorithms have been proposed and touted as particular technologies for some types of data. In the context of high-dimensional data and graphs, we argue that quickly providing the insights of such data sets to users is important to effectively mining them, and thus state-of-the-art techniques need to be further improved by considering these insights. Therefore, in practice, such snapshot techniques of the data are quite useful to enhance those ad-hoc algorithms.

To support our arguments, we discussed three classic issues in data analysis that are problematic in the context of high-dimensional data – clustering, classification, and visualization. Considering the equivalence of matrix and graph, we discussed them together. We propose to make use of spectral techniques and their variants in the form of discrete dynamic systems for meeting our targets. As investigated in Section 2.4 of Chapter 2, possessing extensive and foundational relationships with many areas, spectral techniques and their variants are able to provide quick insights of high-dimensional data and graphs. We are seeing a series of trends towards this awareness and several

notable successes in other domains [Kle99, PBMW98, SM00], although researchers in the data mining community are making initial efforts [AFK⁺01]. As a matter of fact, if data mining is to fulfill its prophecy as the solution to high-dimensional data and graphs, then it has to be better integrated with spectral analysis that has been theoretically and empirically proven successful in other information technologies (e.g. information retrieval, image processing, and etc.). This thesis is an effort towards fulfilling this important goal. In doing so, we select the preprocessing step as a starting point where spectral analysis and data mining techniques are integrated. We hope that there will be more research that are able to make the best use of spectral information in the process of data mining techniques.

6.1 Future Works

As works in the thesis have a broad scope over classic issues of data mining, future works in each issue comprises either extending previously taken approaches and resolving remaining problems, or broadening the scope further in new and unexplored directions. We briefly discuss them below.

In Chapter 3, we present spectral analysis of a similarity matrix by p -normalization for estimating an important parameter – the number of clusters. The theoretical and empirical results have shown its effectiveness in quickly “feeling” the clustering structure of this similarity matrix. This means not only eigenvalues, but also eigenvectors of p -normalization matrix can do so. However, our discussion is limited to eigenvalues. Therefore, a natural extension to eigenvectors may reveal more interesting results. Moreover, if regarding a graph as the input of our method, such extension to eigenvectors may lead to a novel method of partitioning graph.

As an attempt to develop a novel kernel for classification in Chapter 4, we foresee more future works for research. In particular, our immediate interest is to be able to find a suitable value of k for each category in classification by means of automated mechanisms. This is important for practical reasons as maintaining the right value of k over the lifetime of classification can significantly improve classification accuracy. In addition, we also presented two transformation methods used in spectral clustering. Our future work along this line is to make other transformation methods¹ used in spectral clustering effective and practical in the framework of spectral

¹For example, Ham *et al.* introduces some of them in the framework of kernel view [HLMS04].

kernels. Since there are only two text collections used in experiments, it would be interesting to do experiments on more data sets with different characteristics, such as gene expression data. However, because characteristics of gene expression data and the task of biomarker discovery are different from text data and text analysis tasks, the application of spectral kernels to gene expression data may need extensive study.

Finally in Chapter 5, there are a number of interesting future works on resonance models. First, a theoretical analysis of the non-linear model will provide insights to the understanding of more properties and characteristics of the model. Second, we would like to explore other types of data terrains that could potentially provide deeper insights about the data so that they can assist users in achieving better analysis. Third, text is an interesting data with wide applications in the real world. It would be a pioneering and useful exploration by visualizing text data with our method. However, due to the sparseness of text data and complexity of word distributions in different topics, new algorithms based on resonance models need to be carefully designed. This situation is similar to the application of resonance models to the biomarker discovery task in gene expression data. Fourth, we will be implementing a system that provides such a visualization of the data terrain. Such a system will be interactive and display the terrain in 3D view providing mechanisms to zoom and pan on specific areas of the data for further analysis, or exportation to analytical algorithms.

6.2 List of Publications

The work in this thesis contributed a total of 17 international publications as listed below. The publications that result indirectly from the work done in this thesis are marked with a ‘★’.

Book Chapters:

- Wenyan Li, Wee-Keong Ng and Kok-Leong Ong. Graph-based Data Mining. In David Tanar, editor, Research and Trends in Data Mining Technologies and Applications: Advances in Data Warehousing and Mining, volume 1, in press, Idea Group Inc, 2007.

CHAPTER 6. CONCLUSIONS

- Wenyuan Li, Kok-Leong Ong and Wee-Keong Ng. Information Navigation and Knowledge Discovery in Virtual Communities. In Subhasish Dasgupta, editor, Encyclopedia of Virtual Communities and Technologies, in press. Idea Group, 2005.
- Wenyuan Li and Wee-Keong Ng. Spectral Methods for Data Clustering. In John Wang, editor, Encyclopedia of Data Warehousing and Mining, in press. Idea Group, 2005.

Journal Papers:

- Wenyuan Li, Kok-Leong Ong and Wee-Keong Ng. Enhancing the Effectiveness of Clustering with Spectra Analysis. IEEE Transaction on Knowledge and Data Engineering. To appear.
- Wenyuan Li, Yanxiong Peng, Hung-Chung Huang and Ying Liu. Biomarker Discovery And Visualization In Gene Expression Data With Efficient Generalized Matrix Approximations. Journal of Bioinformatics and Computational Biology (JBCB). To appear.
- Wenyuan Li, Ying Liu, Hung-Chung Huang, Yanxiong Peng, Yongjing Lin, Wee-Keong Ng and Kok-Leong Ong. Dynamical Systems for Discovering Protein Complexes and Functional Modules from Biological Networks. IEEE/ACM Transaction on Computational Biology and Bioinformatics (TCBB). To appear.
- Xiuwen Zheng, Hung-Chung Huang, Wenyuan Li, Peng Liu, Quan-Zhen Li, and Ying Liu. Modeling Nonlinearity in Dilution Design Microarray Data. Bioinformatics. To appear.*
- Wenyuan Li, Yongjing Lin, and Ying Liu. The Structure of Weighted Small-World Network. Physica A: Statistical Mechanics and its Applications, 376: 708-718, 2007.
- Yanxiong Peng, Wenyuan Li and Ying Liu. A Hybrid Approach for Biomarker Discovery from Microarray Gene Expression Data. Cancer Informatics. 2: 301-311, 2006.

Conference Papers:

- Wenyan Li, Yanxiong Peng, Hung-Chung Huang and Ying Liu. Efficient Generalized Matrix Approximations for Biomarker Discovery and Visualization in Gene Expression Data. Life Science Society Computational Systems Bioinformatics Conference (CSB), Stanford University, USA, August 2006.
- Wenyan Li, Kok-Leong Ong and Wee-Keong Ng. Text Mining - A Discrete Dynamical System Approach Using the Resonance Model. In Proceeding of the 4th Australasian Data Mining Conference, Sydney, Australia, December 2005.
- Wenyan Li, Kok-Leong Ong and Wee-Keong Ng. Visual Terrain Analysis of High-dimensional Datasets. In Proceeding of the 9th European Conference on Principles of Data Mining and Knowledge Discovery, Porto, Portugal, October 2005.
- Wenyan Li, Kok-Leong Ong, Wee-Keong Ng and Aixin Sun. Spectral Kernels for Classification. In Proceeding of the 7th International Conference on Data Warehousing and Knowledge Discovery, Copenhagen, Denmark, August 2005.
- Hai Wang, Wenyan Li, Zengzhi Li and Lin Fan. Finding Closed Itemsets in Data Streams. Special Session on Knowledge Discovery in Data Streams in Conjunction with the 9th International Conference on Knowledge-based Intelligent Information & Engineering Systems, Melbourne, Australia, September 2005.*
- Wenyan Li, Wee-Keong Ng and Ee-Peng Lim. Spectral Analysis of Text Collection for Similarity-based Clustering. In Proceeding of the 20th International Conference on Data Engineering, Boston, USA, March 2004.
- Wenyan Li, Wee-Keong Ng, Kok-Leong Ong and Ee-Peng Lim. A Spectroscopy of Texts for Effective Clustering. In Proceeding of the 8th European Conference on Principles of Data Mining and Knowledge Discovery, Pisa, Italy, September 2004.
- Kok-Leong Ong, Wenyan Li, Wee-Keong Ng and Ee-Peng Lim. SCLOPE: An Algorithm for Clustering Data Streams of Categorical Attributes. In Proceeding of the 6th International Conference on Data Warehousing and Knowledge Discovery, Zaragoza, Spain, September 2004.*

References

- [AC03] M. H. Arshad and P. K. Chan. Identifying outliers via clustering for anomaly detection. Technical Report TR CS-2003-19, Florida Institute of Technology, 2003.
- [ACL00] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proc. of STOC*, pages 171–180, Portland, OR, May 2000.
- [Ada99] L. A. Adamic. The small world Web. In *Proc. of ECDL*, pages 443–452, Paris, France, September 1999.
- [Ae02] SA Armstrong and *et. al.* MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics*, 30(1):41–47, 2002.
- [AFK⁺01] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proc. of STOC*, pages 619–626, Crete, Greece, July 2001.
- [AFKM01] D. Achlioptas, A. Fiat, A. Karlin, and F. McSherry. Web search through hub synthesis. In *Proc. of FOCS*, pages 611–618, Las Vegas, 2001.
- [Agg01] C. C. Aggarwal. On the effects of dimensionality reduction on high dimensional similarity search. In *Proc. of PODS*, pages 256–266, Santa Barbara, CA, May 2001.
- [Agg02] C. C. Aggarwal. Hierarchical subspace sampling: A unified framework for high dimensional data reduction, selectivity estimation and nearest neighbor search. In *Proc. of SIGMOD*, pages 452–463, Madison, Wisconsin, June 2002.

REFERENCES

- [AGGR98] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of SIGMOD*, 1998.
- [AKK96] M. Ankerst, D. A. Keim, and H.-P. Kriegel. Circle segments: A technique for visually exploring large multidimensional data sets. In *Proc. of Visualization '96, Hot Topic Session*, San Francisco, CA, 1996.
- [Alb01] R. Albert. *Statistical Mechanics of Complex Networks*. PhD thesis, University of Notre Dame, Notre Dame, Indiana, April 2001.
- [Alo86] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.
- [AM01] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. In *Proc. of STOC*, pages 611–618, Hersonissos, Greece, 2001.
- [APW⁺99] C. C. Aggarwal, C. Procopiuc, J. L. Wolf, P. Yu, and J. S. Park. Fast algorithm for projected clustering. In *Proc. of SIGMOD*, pages 61–72, 1999.
- [ASP⁺93] E. A., M. C. Stone, K. Pier, W. Buxton, and T. DeRose. Toolglass and magic lenses: The see-through interface. In *Proc. of SIGGRAPH*, pages 73–80, Anaheim, CA, 1993.
- [AW95] C. Ahlberg and E. Wistrand. IVEE: An information visualization and exploration environment. In *Proc. of Int. Symp. on Information Visualization*, pages 66–73, Atlanta, GA, 1995.
- [AY00] C. C. Aggarwal and P. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proc. of SIGMOD*, pages 70–81, 2000.
- [BA99] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, October 1999.
- [BAJB00] A.-L. Barabási, R. Albert, H. Jeong, and G. Bianconi. Power-law distribution of the world wide web. *Science*, 287(5461):2115, March 2000.

REFERENCES

- [BDR⁺04] A.-L. Barabási, Z. Deszo, E. Ravasz, S. H. Yook, and Z. N. Oltvai. Scale-free and hierarchical structures in complex networks. In *Sitges Proc. on Complex Networks*, 2004.
- [Ber02] P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, 2002.
- [BFK01] Z. Bi, C. Faloutsos, and F. Korn. The DGX distribution for mining massive, skewed data. In *Proc. of SIGKDD*, pages 17–26, San Francisco, CA, August 2001.
- [BGRS99] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is “nearest neighbor” meaningful? In *Proc. of ICDT*, pages 217–235, Jerusalem, Israel, January 1999.
- [BJ03] F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2003.
- [BLS99] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: A Survey*. SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1999.
- [Bol98] D. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, December 1998.
- [Bol01] B. Bollobás. *Random Graphs*. Cambridge University Press, 2nd edition, 2001.
- [CC00] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. of 8th Int. Conf. on Intelligent System for Molecular Biology*, 2000.
- [CC01] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2nd edition, 2001.
- [CD99] J. M. Carlson and J. Doyle. HOT: A mechanism for power laws in designed systems. *Physical Review E*, 60:1412–1427, August 1999.

REFERENCES

- [CD04] H. Chai and C. Domeniconi. An evaluation of gene selection methods for multi-class microarray data classification. In *Proc. of the 2nd European Workshop on Data Mining and Text Mining in Bioinformatics*, pages 7–14, 2004.
- [CFT03] Z. Chen, A. Fu, and J. Tang. On complementarity of cluster and outlier detection schemes. In *Proc. of DaWaK*, Prague, Czech Republic, September 2003.
- [CFZ99] C.-H. Cheng, A. W. Fu, and Y. Zhang. Entropy-based subspace clustering for mining numerical data. In *Proc. of SIGKDD*, pages 84–93, 1999.
- [CGFW05] W. Chu, Z. Ghahramani, F. Falciani, and D. Wild. Biomarker discovery in microarray gene expression data with gaussian processes. *Bioinformatics*, 21(16):3385–3393, 2005.
- [CH00] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, March/April 2000.
- [Chu97] F. R. K. Chung. *Spectral Graph Theory*. Number 92 in CBMS Regional Conference Series in Mathematics. American Mathematical Society, 1997.
- [CKS98] G. Chartrand, G. Kubicki, and M. Schultz. Graph similarity and distance in graphs. *Aequationes Mathematicae*, 55(1-2):129–145, 1998.
- [Cle93] W. S. Cleveland. *Visualizing Data*. H. Press, Summit NJ, 1993.
- [CLV03] F. Chung, L. Lu, and V. Vu. The spectra of random graphs with given expected degrees. *Proc. of National Academy of Sciences*, 100(11):6313–6318, May 2003.
- [CM00] K. Chakrabarti and S. Mehrotra. Local dimensionality reduction: A new approach to indexing high dimensional spaces. In *Proc. of VLDB*, pages 89–100, Cairo, Egypt, September 2000.

REFERENCES

- [CST00] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.
- [CSTL01] N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. In *Proc. of ICML*, pages 66–73, Williams College, US, July 2001.
- [CSZ94] P. K. Chan, M. Schlag, and J. Y. Zien. Spectral k-way ratio-cut partitioning and clustering. *IEEE Trans. on CAD-Integrated Circuits and Systems*, 13:1088–1096, 1994.
- [CTL03] V. Castelli, A. Thomasian, and C.-S. Li. Csvd: Clustering and singular value decomposition for approximate similarity search in high-dimensional spaces. *IEEE Trans. on Knowledge and Data Engineering*, 15(3):671–685, May/June 2003.
- [CTX⁺04] G. Cong, A. K. H. Tung, X. Xu, F. Pan, and J. Yang. FARMER: Finding interesting rule groups in microarray datasets. In *Proc. of SIGMOD*, pages 143–154, Paris, France, 2004.
- [dABY05] M. A. M. de Aguiar and Y. Bar-Yam. Spectral analysis and the dynamic response of complex networks. *Physical Review E*, 71(016106), 2005.
- [DB02] Z. Dezso and A.-L. Barabási. Halting viruses in scale-free networks. *Physical Review E*, 65:055103, May 2002.
- [DB04] J. G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5:845–889, December 2004.
- [DDL⁺90] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.
- [dFCRTB05] L. d. F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas. Characterization of complex networks: A survey of measurements, 2005.
- [DFK⁺04] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine Learning*, 56(1-3):9–33, July 2004.

REFERENCES

- [DGY04] C. Demir, S. H. Gultekin, and B. Yener. Spectral analysis of cell-graphs of cancer. Technical Report TR-04-17, RPI, 2004.
- [DH73] W. E. Donath and A. J. Hoffman. Lower bounds for partitioning of graphs. *IBM J. Res. Develop*, 17:420–425, 1973.
- [Dhi01] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of SIGKDD*, San Francisco, CA, August 2001.
- [DHZ01a] C. Ding, X. He, and H. Zha. A spectral method to separate disconnected and nearly-disconnected web graph components. In *Proc. of SIGKDD*, pages 275–280, San Francisco, CA, August 2001.
- [DHZ⁺01b] C. Ding, X. He, H. Zha, M. Gu, and H. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *Proc. of ICDM*, 2001.
- [DHZS02] C. Ding, X. He, H. Zha, and H. Simon. Unsupervised learning: Self-aggregation in scaled principal component space. In *Proc. of PKDD*, pages 112–124, 2002.
- [Din04] C. Ding. Tutorial: Spectral clustering. In *Proc. of ICML*, Alberta, Canada, July 2004.
- [DL99] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *Proc. of SIGKDD*, pages 43–52, San Diego, CA, 1999.
- [DL00] M. Dash and H. Liu. Feature selection for clustering. In *Proc. of PAKDD*, pages 110–121, 2000.
- [DLR77] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. of Royal Statistical Society*, 39:1–38, 1977.
- [DMM03] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proc. of SIGKDD*, pages 89–98, Washington, D.C., 2003.

REFERENCES

- [DMW03] P. Sheridan Dodds, R. Muhamad, and D. J. Watts. An experimental study of search in global social networks. *Science*, 301:827–829, August 2003.
- [Don00] D. L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. *Math Challenges of the 21st Century*, August 2000.
- [DR01] P. Domingos and M. Richardson. Mining the network value of customers. In *Proc. of SIGKDD*, pages 57–66, San Francisco, California, August 2001.
- [DUdA06] R. Diaz-Uriarte and S. Alvarex de Andrez. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7(3), 2006.
- [ea99] T. R. Golub et. al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:(5439):531–537, October 1999.
- [ea00] D. T. Ross et. al. Systematic variation in gene expression patterns in human cancer cell lines. *Nature*, 24(3):227–235, March 2000.
- [EAd03] F. Evans, M. Alder, and C. deSilva. Determining the number of clusters in a mixture by iterative model space refinement with application to free-swimming fish detection. In *Proc. Digital Imaging Computing: Techniques and Applications*, Sydney, Australia, December 2003.
- [ER59] P. Erdős and A. Rényi. On random graphs. *Publicationes Mathematicae*, 6:290–297, 1959.
- [ER60] P. Erdős and A. Rényi. On the evolution of random graphs. *Publication of Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–61, 1960.
- [ESBB98] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 95(25):14863–14868, December 1998.

REFERENCES

- [FFF99] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. In *Proc. of SIGCOMM*, pages 251–262, Cambridge MA, September 1999.
- [Fie73] M. Fiedler. Algebraic connectivity of graphs. *Czech. Math. J.*, 23:298–305, 1973.
- [FJS01] R. Ferrer, C. Janssen, and R. V. Solé. Topology of technology graphs: Small world patterns in electronic circuits. *Physical Review E*, 64(046119), September 2001.
- [FK81] Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.
- [FKV04] A. Frieze, R. Kannan, and S. Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, November 2004.
- [FMS96] C. Faloutsos, Y. Matias, and A. Silberschatz. Modeling skewed distribution using multifractals and the ‘80-20’ law. In *Proc. of VLDB*, pages 307–317, Mumbai (Bombay), September 1996.
- [FR98] C. Fraley and A. E. Raftery. How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998.
- [FRB03] B. Fischer, V. Roth, and J. M. Buhmann. Clustering with the connectivity kernel. In *Proc. of NIPS*, Cambridge, MA, 2003.
- [FS01] R. Ferrer and R. V. Solé. The small world of human language. *Proc. of the Royal Society of London Series B - Biological Sciences*, 268(1482):2261–2265, November 2001.
- [FT74] J. Friedman and J. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transaction on Computer*, 23(9):881–890, 1974.

REFERENCES

- [Gär03] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explorations Newsletter*, 5(1):49–58, July 2003.
- [GC04] P. H. Gosselin and M. Cord. Semantic kernel updating for content-based image retrieval. In *Proc. of IEEE 6th International Symposium on Multimedia Software Engineering*, pages 537–544, Miami, Florida, December 2004.
- [Ge99] T.R.S. Golub and *et. al.* Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
- [GKR00] D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical system. *VLDB Journal*, 8(3/4):222–236, 2000.
- [GL96] G. Golub and C. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, November 1996.
- [GMS90] R. Grone, R. Merris, and V. S. Sunder. The Laplacian spectrum of a graph. *SIAM Journal on Matrix Analysis and Applications*, 11(2):218–238, 1990.
- [GMS03] C. Gkantsidis, M. Mihail, and A. Saberi. Conductance and congestion in power law graphs. In *Proc. of SIGMETRICS*, pages 148–159, San Diego, CA, June 2003.
- [Gor99] A. Gordon. *Classification*. Chapman and Hall/CRC, 2nd edition, 1999.
- [GR71] G. Golub and C. Reinsch. *Handbook for Matrix Computation II, Linear Algebra*. Springer-Verlag, New York, 1971.
- [Gra03] M. Granovetter. Ignorance, knowledge, and outcomes in a small world. *Science*, 301:773–774, August 2003.
- [GW69] R. Gnanadesikan and M. Wilk. *Multivariate Analysis II*, chapter Data analytic methods, pages 593–638. Academic Press, 1969.

REFERENCES

- [GZRL05] D. Gleich, L. Zhukov, M. Rasmussen, and K. Lang. The world of music: SDP layout of high dimensional data. In *Proc. of 11th IEEE Conf. on Information Visualization*, 2005.
- [Hac03] R. Hackathorn. The link is the thing. *DB Review*, August 2003.
- [Har72] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.
- [HBV01] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3):107–145, December 2001.
- [HH03] M. A. Hall and G. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Engineering*, 15(6):1437–1447, 2003.
- [HK92] L. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE. Trans. on Computed Aided Desgin*, 11:1074–1085, 1992.
- [HK01] J. Han and M. Kambr. *Data Mining Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.
- [HLMS04] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proc. of ICML*, Alberta, Canada, July 2004.
- [Hof99] T. Hofmann. Probabilistic latent semantic indexing. In *Proc. of SIGIR*, pages 50–57, Berkeley, CA, August 1999.
- [Hop82] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA*, 79(8):2554–2558, April 1982.
- [HS89] T. Hastie and W. Stuetzle. Principal curve. *Journal of the American Statistical Association*, 84:502–516, 1989.

REFERENCES

- [HSL⁺00] E. Hartuv, A. O. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrach, and R. Shamir. An algorithm for clustering cDNA fingerprints. *Genomics*, 66(3):249–256, June 2000.
- [Hub85] P. J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–474, 1985.
- [HXD03] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24:1641–1650, 2003.
- [HY01] M. Hansen and B. Yu. Model selection and the principle of minimum description length. *J. of the American Statistical Association*, 96(454):746–774, 2001.
- [IBL02] B. I.S. Inza, R. Blanco, and P. Larranaga. Gene selection by sequential search wrapper approaches in microarray cancer class prediction. *Journal of Intelligent and Fuzzy Systems*, 12:25–33, 2002.
- [ID90] A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proc. of Visualization '90*, pages 361–370, San Francisco, CA, 1990.
- [IWM03] A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50(3):321–354, March 2003.
- [JK01] S. Lawrence J. Kleinberg. The structure of the web. *Science*, 294:1849–1850, November 2001.
- [JMF99] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(13):264–323, 1999.
- [Joa02] T. Joachims. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms*. Kluwer, Norwell, MA, May 2002.
- [Joa03] T. Joachims. Transductive learning via spectral graph partitioning. In *Proc. of ICML*, pages 290–297, Washington, DC, August 2003.

REFERENCES

- [JSR03] J. Jaeger, R. Sengupta, and W. L. Ruzzo. Improved gene selection for classification of microarrays. In *Proc. of PSB*, 2003.
- [Kar02] G. Karypis. CLUTO: A clustering toolkit. Technical Report #02-017, University of Minnesota, 2002.
- [KDK03] K. Kummamuru, A. Dhawale, and R. Krishnapuram. Fuzzy co-clustering of documents and keywords. In *Proc. of Fuzzy Systems*, pages 772–777, 2003.
- [Kei02] D. A. Keim. Information visualization and visual data mining. *IEEE Trans. on Visualization and Computer Graphics*, 7(1):100–107, January-March 2002.
- [KG03] M. Koyutürk and A. Grama. Proximus: A framework for analyzing very high dimensional discrete-attributed datasets. In *Proc. of SIGKDD*, pages 147–156, Washington, D.C., 2003.
- [KGR] M. Koyutürk, A. Grama, and N. Ramakrishnan. Non-orthogonal decomposition of binary matrices for bounded-error data compression and analysis. *ACM Trans. on Mathematical Software*.
- [KGR05] M. Koyutürk, A. Grama, and N. Ramakrishnan. Compression, clustering and pattern discovery in very high dimensional discrete-attribute datasets. *IEEE Trans. on Knowledge and Data Engineering*, 17(4):447–461, 2005.
- [KHK99] G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: Hierarchical clustering using dynamic modeling. *IEEE Computer*, 32(8):68–75, 1999.
- [KK94] D. A. Keim and H.-P. Kriegel. Visdb: Database exploration using multi-dimensional visualization. *Computer Graphics and Applications*,, pages 40–49, Sept. 1994.
- [KKA95] D. A. Keim, H.-P. Kriegel, and M. Ankerst. Recursive pattern: A technique for visualizing very large amounts of data. In *Proc. of Visualization '95*, pages 279–286, Atlanta, GA, 1995.

REFERENCES

- [KKM03] S. D. Kamvar, D. Klein, and C. D. Manning. Spectral learning. In *Proc. of 18th International Joint Conference on Artificial Intelligence (IJCAI)*, August 2003.
- [KKT03] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proc. of SIGKDD*, pages 137–146, Washington, DC, August 2003.
- [Kle99] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [Kle00] J. Kleinberg. Navigation in a small world. *Nature*, 406:845, August 2000.
- [KN98] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *Proc. of VLDB*, pages 392–403, New York, USA, August 1998.
- [KR92] K. Kira and L. Rendell. A practical approach to feature selection. In *Proc. of ICML*, 1992.
- [KSS97] H. Kautz, B. Selman, and M. Shah. ReferralWeb: combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, March 1997.
- [KT99] J. Kleinberg and A. Tomkins. Applications of linear algebra in information retrieval and hypertext analysis. In *Proc. of PODS*, pages 185–193, Philadelphia, Pennsylvania, May 1999.
- [KVV00] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. In *Proc. of FOCS*, pages 367–377, Redondo Beach, 2000.
- [LAN] LANSO. Dept. of Computer Science and the Industrial Liason Office, Univ. of Calif., Berkeley.
- [LDR00a] J. Li, G. Dong, and K. Ramamohanarao. Instance-based classification by emerging patterns. In *Proc. of PKDD*, pages 191–200, Lyon, France, 2000.

REFERENCES

- [LDR00b] J. Li, G. Dong, and K. Ramamohanarao. Making use of the most expressive jumping emerging patterns for classification. In *Proc. of PAKDD*, pages 220–232, Kyoto, Japan, 2000.
- [Lew] David D. Lewis. Reuters-21578 text categorization test collection. <http://www.daviddlewis.com/resources/testcollections/reuters21578/>.
- [LFL98] T. Landauer, P. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, 25:259–284, 1998.
- [Li95] S. Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer-Verlag, London, UK, 1995.
- [LLW02] H. Liu, J. Li, and L. Wong. A comparative study of feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics*, 13:51–60, 2002.
- [LNO05] W. Li, W.-K. Ng, and K.-L. Ong. *Encyclopedia of Virtual Communities and Technologies*, chapter Information Navigation and Knowledge Discovery in Virtual Communities. Idea Group Inc., 2005.
- [LSY98] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems by Implicitly Restarted Arnoldi Methods*. SIAM, Philadelphia, PA, 1998.
- [LW03] J. Liu and W. Wang. OP-Cluster: Clustering by tendency in high dimensional space. In *Proc. of ICDM*, page 187, Melbourne, Florida, November 2003.
- [LY02] W. Li and Y. Yang. Zipf's law in importance of genes for cancer classification using microarray data. *Journal of Theoretical Biology*, 219(4):539–551, December 2002.
- [LYW04] J. Liu, J. Yang, and W. Wang. Biclustering of gene expression data by tendency. In *Proc. of IEEE CSB*, pages 182–193, 2004.

REFERENCES

- [LZJ04] H. Li, K. Zhang, and T. Jiang. Minimum entropy clustering and applications to gene expression analysis. In *Proc. 3rd CSB*, pages 142–151, Stanford, CA, 2004.
- [LZO04] T. Li, C. Zhang, and M. Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20:2429–2437, 2004.
- [MB04] A. Moschitti and C. A. Bejan. A semantic kernel for predicate argument classification. In *Proc. of 8th Conference on Natural Language Learning*, pages 17–24, Boston, MA, USA, May 2004.
- [Men02] F. Menczer. Growing and navigating the small world Web by local content. *Proc. of the National Academy of Sciences*, 99(22):14014–14019, October 2002.
- [MGZ03] M. Mihail, C. Gkantsidis, and E. Zegura. Spectral analysis of internet topologies. In *Proc. of INFOCOM*, San Francisco, CA, 2003.
- [Mil67] S. Milgram. The small world problem. *Psychology Today*, pages 60–67, 1967.
- [Mla98] D. Mladenić. Feature subset selection in text-learning. In *Proc. of ECML*, 1998.
- [MO04] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE Trans. on Computational Biology and Bioinformatics*, 1(1):24–45, January/March 2004.
- [MS65] T. S. Motzkin and E. G. Straus. Maxima for graphs and a new proof of a theorem of Turán. *Canad. J. Math.*, 17(4):533–540, 1965.
- [MS01] M. Maila and J. Shi. A random walks view of spectral segmentation. In *Proc. of the 8th International Workshop on Artificial Intelligence and Statistics*, Florida, January 2001.
- [MS02] J. M. Montoya and R. V. Solé. Small world patterns in food webs. *Journal of Theoretical Biology*, 214(3):405–412, February 2002.

REFERENCES

- [MW90] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23:1&18, 1990.
- [Neu66] J. V. Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign-Urbana, Illinois, 1966.
- [New01] M. E. J. Newman. The structure of scientific collaboration networks. *Proc. of the National Academy of Sciences*, 98(2):404–409, January 2001.
- [New03] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, May 2003.
- [NFW05] E. K. K. Ng, A. W.-C. Fu, and R. C.-W. Wong. Projective clustering by histograms. *IEEE Trans. on Knowledge and Data Engineering*, 17(3):369–383, March 2005.
- [NJW01] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. of NIPS*, pages 849–856, Vancouver, December 2001.
- [NK01] S. Nijssen and J. N. Kok. Faster association rules for multiple relations. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence*, pages 891–896, Seattle, Washington, August 2001.
- [NSP03] M. A. Nascimento, J. Sander, and J. Pound. Analysis of sigmod’s co-authorship graph. *SIGMOD Record*, 32(3), September 2003.
- [NZJ01] A. Y. Ng, A. X. Zheng, and M. I. Jordan. Stable algorithms for link analysis. In *Proc. of SIGIR*, pages 258–266, New Orleans, Louisiana, 2001.
- [ONL02] K.-L. Ong, W.-K. Ng, and E.-P. Lim. Mining relationship graphs for effective business objectives. In *Proc. of PAKDD*, pages 561–566, Taipei, Taiwan, May 2002.

REFERENCES

- [PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.
- [PF00] C. R. Palmer and C. Faloutsos. Density biased sampling: An improved method for data mining and clustering. In *Proc. of SIGMOD*, pages 82–92, Dallas, TX, May 2000.
- [PG88] R. M. Pickett and G. G. Grinstein. Iconographic displays for visualizing multidimensional data. In *Proc. of IEEE Conf. on Systems, Man and Cybernetics*, pages 514–519, Piscataway, NJ, 1988.
- [PGF02] C. R. Palmer, P. B. Gibbons, and C. Faloutsos. ANF: a fast and scalable tool for data mining in massive graphs. In *Proc. of SIGKDD*, Edmonton, Alberta, Canada, July 2002.
- [PHL04] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explorations Newsletter*, 6(1):90–105, June 2004.
- [PJAM02] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *Proc. of SIGMOD*, pages 418–427, Madison, Wisconsin, 2002.
- [Pla99] J. Platt. *Advances in Kernel Methods - Support Vector Learning*, chapter Fast Training of Support Vector Machines using Sequential Minimal Optimization, pages 185–208. MIT Press, Cambridge, MA, USA, 1999.
- [PLD05] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.
- [PTRV98] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Proc. of PODS*, pages 159–168, Seattle, Washington, USA, June 1998.

REFERENCES

- [RB03] E. Ravasz and A.-L. Barabási. Hierarchical organization in complex networks. *Physical Review E*, 67, 2003.
- [Re03] T Rozovskaia and *et. al.* Expression profiles of acute lymphoblastic and myeloblastic leukemias with all-1 rearrangements. *Proc. of National Academy of Sciences USA*, 100(13):7853–7858, 2003.
- [RG01] G. Royle and C. Godsil. *Algebraic Graph Theory*. Springer-Verlag, New York, 2001.
- [RK01] L. D. Raedt and S. Kramer. The levelwise version space algorithm and its application to molecular fragment finding. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence*, pages 853–862, Seattle, Washington, August 2001.
- [RLBB02] V. Roth, T. Lange, M. Braun, and J. Buhmann. A resampling approach to cluster validation. In *Proc. of COMPSTAT*, Berlin, Germany, August 2002.
- [RMC91] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3D visualizations of hierarchical information. In *Proc. of Human Factors in Computing Systems CHI '91 Conf.*, pages 189–194, New Orleans, LA, 1991.
- [RS00] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, December 2000.
- [San90] J. T. Sandefur. *Discrete Dynamical Systems*. Oxford: Clarendon Press, 1990.
- [SAS03] R. Sharan, Moron-Katz Adi, and R. Shamir. CLICK and EXPANDER: a system for clustering and visualizing gene expression data. *Bioinformatics*, 19:1787–1799, 2003.
- [SC02] M. P. Sinka and D. W. Corne. *Soft Computing Systems: Design, Management and Applications*, chapter A Large Benchmark Dataset for Web Document Clustering, pages 881–890. IOS Press, 2002.

REFERENCES

- [SC03] S. Salvador and P. Chan. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. Technical report 2003-18, Florida Institute of Technology, 2003.
- [SCMV02] R. V. Solé, R. F. Cancho, J. M. Montoya, and S. Valverde. Selection, tinkering and emergence in complex networks. *Complexity*, 8(1):20–33, September 2002.
- [SdB00] G. Siolas and F. d’Alché Buc. Support vector machines based on a semantic kernel for text categorization. In *Proc. Int. Joint Conf. on Neural Networks (IJCNN)*, pages 5205–5210, July 2000.
- [Se02] D Singh and *et. al.* Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.
- [Seb02] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
- [Sed01] R. Sedgewick. *Algorithms in C, Part 5: Graph Algorithms*. Addison-Wesley, 3rd edition, 2001.
- [SFYD04] Marco Saerens, Francois Fouss, Luh Yen, and Pierre Dupont. The principal components analysis of a graph, and its relationships to spectral clustering. In *Proc. of ECML*, pages 371–383, Pisa, Italy, September 2004.
- [SGM00] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc. AAAI Workshop on AI for Web Search*, pages 58–64, July 2000.
- [Shn92] B. Shneiderman. Tree visualization with treemaps: A 2d space-filling approach. *ACM Trans. on Graphics*, 11(1):92–99, 1992.
- [SKK00] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, Boston, MA, USA, August 2000.

REFERENCES

- [SL00] H. S. Seung and D. D. Lee. The manifold ways of perception. *Science*, 290(5500):2268–2269, December 2000.
- [SL04] D. Surendran and S. Levy. Visualizing high dimensional datasets with Partiview. In *Proc. of 10th IEEE Conf. on Information Visualization*, 2004.
- [SM00] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905, August 2000.
- [Smy96] P. Smyth. Clustering using monte carlo cross-validation. In *Proc. of SIGKDD*, pages 126–133, Oregon, USA, 1996.
- [SR03] A. J. Seary and W. D. Richards. *Dynamic Social Network Modeling and Analysis*, chapter Spectral Methods for Analyzing and Visualizing Networks: An Introduction, pages 209–228. National Academy Press, Washington, DC, 2003.
- [SRLH97] H. S. Seung, T. J. Richardson, J. C. Lagarias, and J. J. Hopfield. Minimax and hamiltonian dynamics of excitatory-inhibitory networks. In *Proc. of NIPS*, Denver, Colorado, 1997.
- [SSM98] B. Scholkopf, A. Smola, and K.-R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.
- [STC04] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
- [TA] I. Tabus and J. Astola. Gene feature selection. Technical report. <http://www.cs.tut.fi/~tabus/course/GSP/Chapter2GSPSR.pdf>.
- [TdSL00] J. B. Tennenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2322, December 2000.

REFERENCES

- [TM69] J. Travers and S. Milgram. An experimental study of the small world problem. *Sociometry*, 32:425–443, 1969.
- [TM87] T. Toffoli and N. Margolus. *Cellular Automata Machines*. MIT Press, Cambridge, Massachusetts, 1987.
- [Tsa04] P. Tsaparas. Using non-linear dynamical systems for web searching and ranking. In *Proc. of PODS*, pages 59–70, Paris, France, 2004.
- [TWBB01] R. Tibshirani, G. Walther, D. Botstein, and P. Brown. Cluster validation by prediction strength. Technical report, Stanford University, 2001.
- [TWH00] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the gap statistic. Technical Report 208, Dept. of Statistics, Stanford University, 2000.
- [Vap95] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, NY, 1995.
- [Vap98] V. Vapnik. *Statistical Learning Theory*. Wiley, NY, 1998.
- [VDG⁺02] M. Vlachos, C. Domeniconi, D. Gunopulos, G. Kollios, and N. Koudas. Non-linear dimensionality reduction techniques for classification and visualization. In *Proc. of SIGKDD*, Alberta, Canada, July 2002.
- [VHE01] D. Vukadinovic, P. Huan, and T. Erlebach. A spectral analysis of the internet topology. Technical Report 118, ETH TIK-NR, 2001.
- [VM03] D. Verma and M. Meilă. A comparison of spectral clustering methods. Technical Report 03-05-01, UW CSE, 2003.
- [VS03] S. Valverde and R. V. Solé. Hierarchical small worlds in software architecture. *IEEE Trans. in Software Engineering*, 2003.
- [WDN02] D. J. Watts, P. S. Dodds, and M. E. J. Newman. Identity and search in social networks. *Science*, 296:1302–1305, May 2002.

REFERENCES

- [WF00] A. Wagner and D. Fell. The small world inside large metabolic networks. Technical Report 00-07-041, Santa Fe Institute, 2000.
- [WHP03] J. Wang, J. Han, and J. Pei. CLOSET+: searching for the best strategies for mining frequent closed itemsets. In *Proc. of SIGKDD*, Washington, D.C., August 2003.
- [WL93] J. J. Van Wijk and R. D. Van Liere. Hyperslice. In *Proc. of Visualization*, pages 119–125, San Jose, CA, 1993.
- [WM03] T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explorations Newsletter*, 5(1):59–68, July 2003.
- [Wol84] S. Wolfram. Cellular automata as models of complexity. *Nature*, 311:419–424, 1984.
- [WR97] P. Waldemar and T. Ramstad. Hybrid klt-svd image compression. In *IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Los Alamitos, CA, 1997.
- [WR02] J. A. Walter and H. Ritter. On interactive visualization of high-dimensional data using the hyperbolic plane. In *Proc. of SIGKDD*, pages 123–132, 2002.
- [Wri95] W. Wright. Information animation applications in the capital markets. In *Proc. of Int. Symp. on Information Visualization*,, pages 19–25, Atlanta, GA, 1995.
- [WS98] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440–442, 1998.
- [WWYY02] H. Wang, W. Wang, J. Yang, and P. Yu. Clustering by pattern similarity in large data sets. In *Proc. of SIGMOD*, pages 394–405, Madison, Wisconsin, June 2002.
- [XJ03] E. P. Xing and M. I. Jordan. On semidefinite relaxation for normalized k-cut and connections to spectral clustering. Technical Report CSD-03-1265, UC Berkeley, 2003.

REFERENCES

- [Yan97] Yiming Yang. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University, April 1997.
- [YCN04] K. Y. Yip, D. W. Cheung, and M. K. Ng. Harp: A practical projected clustering algorithm. *IEEE Trans. on Knowledge and Data Engineering*, 16(11):1387–1397, November 2004.
- [Ye04] Jieping Ye. Generalized low rank approximations of matrices. In *Proc. of ICML*, page 112, Banff, Alberta, Canada, 2004.
- [YGY02] Y. Yang, X. Guan, and J. You. CLOPE: A fast and effective clustering algorithm for transactional data. In *Proc. of SIGKDD*, pages 682–687, Edmonton, Canada, July 2002.
- [YH03] X. Yin and J. Han. CPAR: Classification based on predictive association rules. In *Proc. of SIAM ICDM*, San Francisco, CA, May 2003.
- [YL95] J.-F. Yang and C.-L. Lu. Combined techniques of singular value decomposition and vector quantization for image coding. *IEEE Trans. on Image Processing*, 4(8):1141–1146, August 1995.
- [YL04] L. Yu and H. Liu. Redundancy based feature selection for microarray data. In *Proc. of SIGKDD*, Seattle, Washington, 2004.
- [YP97] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proc. of ICML*, pages 412–420, Nashville, US, 1997.
- [YWWY02] J. Yang, W. Wang, H. Wang, and P. Yu. delta-cluster: Capturing subspace correlation in a large data set. In *Proc. of ICDE*, pages 517–528, San Jose, California, February 2002.
- [YWWY03] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *Proc. of the 3rd IEEE Conf. on Bioinformatics and Bioengineering*, pages 321–327, 2003.

REFERENCES

- [ZDG⁺01] H. Zha, C. Ding, M. Gu, X. He, and H. D. Simon. Spectral relaxation for k-means clustering. In *Proc. of NIPS*, pages 1057–1064, Vancouver, Canada, 2001.
- [ZHD⁺01] H. Zha, X. He, C. Ding, M. Gu, and H. Simon. Bipartite graph partitioning and data clustering. In *Proc. of CIKM*, pages 25–31, Atlanta, 2001.
- [ZK02a] Y. Zhao and G. Karypis. Criterion functions for document clustering experiments and analysis. Technical Report #01-40, University of Minnesota, 2002.
- [ZK02b] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. Technical Report #02-22, University of Minnesota, 2002.
- [ZK04] Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3):311–331, 2004.