Multi-View Signal Processing and Learning on Graphs

THÈSE N^{O} 6213 (2014)

PRÉSENTÉE LE 19 SEPTEMBRE 2014

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE DE TRAITEMENT DES SIGNAUX 4

PROGRAMME DOCTORAL EN GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Xiaowen DONG

acceptée sur proposition du jury:

Prof. M. Grossglauser, président du jury
Prof. P. Frossard, Prof. P. Vandergheynst, directeurs de thèse
Prof. M. Bronstein, rapporteur
Dr D. Gatica-Perez, rapporteur
Prof. A. Ortega, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2014

To my grandma

Acknowledgement

It certainly takes a miracle for someone like me who did not like signal processing at all at college to finish a PhD in this field at one of the top engineering schools in Europe. And I would never have reached this point without all the people behind me. A PhD is a special journey in our life, and life is about everything.

PhD is about passion, creativity and determination. I have been fortunate to have the opportunity to work with my advisors, Prof. Pascal Frossard and Prof. Pierre Vandergheynst, who have taught me just that in the past five years. Especially, I would like to thank Pascal for encouraging me to always face new challenges, guiding me towards a Swiss precision in academic research, and giving me much freedom in exploring directions that I am interested in. I would like to thank Pierre for sharing his passion and energy with me, teaching me the right attitude in doing science, and inspiring me with all those mind-blowing ideas. I thank both of them for giving me the opportunity to study in such a nice environment, never losing faith in me even during the most difficult moments, and, most importantly, tolerating all those "strange" thoughts their first ever Chinese student has inherited from an ancient oriental culture. To obtain a PhD is certainly not easy, but they made me feel less difficult and more enjoyable.

PhD is about endless exploration and rethinking. I thank the members of my thesis jury, Prof. Matthias Grossglauser, Prof. Michael Bronstein, Dr. Daniel Gatica-Perez, and Prof. Antonio Ortega, for spending time reading the draft of my PhD thesis, and giving me valuable comments and suggestions during and after my PhD exam. I also thank Dorina and Tasos for proofreading the draft, and Thomas and Benjamin for their help with the French translation of the abstract. They made me realize that, even though I thought I was clear and confident on what I have been doing, there is always room for a second thought and possible improvement. I will always take their words with me in future research. The work presented in this thesis has been kindly supported by the Nokia Research Center (NRC), Lausanne and the Hasler Foundation, to whom I will always remain grateful.

PhD is about teamwork and collaboration. During my studies at EPFL, I am thankful for having the chance to work closely with a number of people. My sincere gratitude goes to Dr. Nikolai Nefedov, who guided me in the early stages of my PhD in collaboration with Nokia; to Prof. Antonio Ortega, with whom I had inspirational discussions while working together on some very interesting research ideas; to Dorina for her countless help and support in our joint project and the writing of my thesis; to David and Benjamin for the fruitful discussions on graphs; to Hussein for our obsession to the classification problem; and to Xuan, Zhe, Daphne, Pulkit, Sapan, Stéphane, Matthieu, Clément, and Renata, for working together with me on several interesting projects. I have learned a lot from all of you.

PhD is about an open mind. I feel grateful to Dr. Olivier Verscheure, who provided me with the opportunity to do an internship at IBM Research in Dublin, during which I have learned so many things from different perspectives. I would like to thank Dr. Francesco Calabrese and Dr. Michele Berlingerio, who always put trust in me and guided me through my internship; my IBM colleagues Dimitris, Yiannis, Aris, and Giusy, who shared their inspiring thoughts with me and made me feel at home in the new environment. Special thanks to Dimitris for all the discussions we had together, and his continued support in the preparation for our paper even after I returned from my internship.

PhD is also about friendship. During the past five years, I have been extremely lucky to stay in one of the best lab environments one could ever hope for. Many thanks to my LTS4 lab mates Tamara, Eirini, Elif, Dorina, Dorna, Sofia, Ana, Ye, Laura, Yanhui, Pinar, Zafer, Jacob, Nikos, Luigi, Vijay, Hyunggon, Attilio, Eymen, Lorenzo, David, Thomas, Xavi, Davide, Arthur, Cédric, Hussein, and Stefano, and the LTS2 colleagues Alex, Momo, Anna (Llagostera), Emmanuel, Gilles, Simon, Jason, Mahdad, Kirell, Benjamin, Vassilis, Johann, and Nathanaël. You are not just wonderful colleagues but also my dear friends. Special thanks to Hyunggon and Jacob for tips and encouragement when I started my PhD, to Tamara who has always been there for any kind of help, to Luigi for the Italian mafia, to Ana for the fresh air from Latin America, and to Hussein who is always ready to challenge me. I would like to especially thank Nikos for tolerating me for almost two years in the office (which must have been a daunting task), taking care of our plant (which is still growing healthily), and sharing with me his intelligence from the Aegean sea. My deep gratitude also goes to Dorina (Nana) and Sofia, who are always there for interesting discussions, the psychological sessions, and, most importantly, the smoothies. I am so happy to have you as my Greek sisters. Many thanks also to Rosie for all her administrative support and the hiking and music experiences.

And friendship goes beyond the lab environment. I would like to thank all the dear friends that I have met in the LTS corridor in these years, for the time we spent and the interesting experiences we had together. Special thanks to Ashkan, who trusted and helped me from day one; and to Anna (Auría), who introduced me to a Spanish world in Lausanne. Many thanks to my Chinese friends for the soccer we have always enjoyed together, and especially to Jingmin, Jun, Rui and Shenjie for the unforgettable experiences at San Siro; to Ye and Jingge for hosting me for two months when I was homeless, and all the games and laughters. I also feel deeply grateful to all the friends I met in Dublin. I will remember playing soccer with Fabio, chatting with Yuxiao, singing karaoke with Bei and Jiayuan, and going round with Lina and Hu; I will also cherish the hospitality of Xiaoyu and Qingli, and the trip to the beautiful Galway and Aran Islands with Liping and Shen.

PhD is not only about doing research, but also learning to behave. And I had my lessons even before I started the journey. My sincere thanks go to Prof. Nick Kingsbury at University of Cambridge, and to Prof. Steven D. Blostein at Queen's University. Without the kindness they showed towards me before I came to EPFL, this thesis would have never existed. I will always remain grateful to them and follow their examples.

Finally, PhD is about love. Words cannot express my infinite gratitude to Hongyan, and to my family, whose unconditional love and endless support made me brave and dare to dream the impossible. You are the reasons I was in the past, I am at the present, and I will be in the future. "You are all my reasons" (A Beautiful Mind).

Abstract

In modern information processing tasks we often need to deal with complex and "multi-view" data, that come from multiple sources of information with structures behind them. For example, observations about the same set of entities are often made from different angles, or, individual observations collected by different sensors are intrinsically related. This poses new challenges in developing novel signal processing and machine learning techniques to handle such data efficiently.

In this thesis, we formulate the processing and analysis of multi-view data as signal processing and machine learning problems defined on graphs. Graphs are appealing mathematical tools for modeling pairwise relationships between entities in datasets. Moreover, they are flexible and adaptable to incorporate multiple sources of information with structures. For instance, multiple types of relationships between the same entities in a certain set can be modeled as a multi-layer graph, where the layers share the same set of vertices (entities) but have different edges (relationships) between them. Alternatively, information from one source can be modeled as signals defined on the vertex set of a graph that corresponds to another source. While the former setting is an extension of the classical graph-based learning in machine learning, the latter leads to the emerging research field of signal processing on graphs. In this thesis, we bridge the gap between the two fields by studying several problems related to the clustering, classification and representation of data of various forms associated with weighted and undirected graphs.

First, we address the problem of analyzing multi-layer graphs and propose methods for clustering the vertices by efficiently combining the information provided by the multiple layers. To this end, we propose to combine the characteristics of individual graph layers using tools from subspace analysis on a Grassmann manifold. The resulting combination can be viewed as a low dimensional representation of the original data, which preserves the most important information from diverse relationships between entities. We use our algorithm in clustering methods and demonstrate that the proposed method is superior to baseline schemes and competitive to state-of-the-art techniques.

Next, we approach the problem of combining different layers in a multi-layer graph from a different perspective. Specifically, we consider the eigenvectors of the graph Laplacian matrix of one layer as signals defined on the vertex set of another layer. We propose a novel method based on a graph regularization framework, which produces a set of "joint eigenvectors", or a "joint spectrum", shared by the two layers. We use this joint spectrum in clustering problems on multi-layer graphs. Compared to our previous approach and most of the state-of-the-art techniques, a unique characteristic and potential advantage of this method is that, it allows us to combine individual layers based on their respective importance in a convincing way.

Third, we build on the setting in the second approach and study in general the classification problem of signals on graphs. To this end, we adopt efficient signal representations defined in

the graph spectral domain, and analyze theoretically their potential for the classification of graph signals under a generic signal model. We then propose discriminative spectral features and test their performances in synthetic classification tasks compared to features extracted in the spatial domain. We apply the proposed spectral classification framework to applications such as the inference of mobility patterns in traffic and social media data.

Finally, given the crucial role of a meaningful graph, which is not always naturally defined from the application domains, in efficient signal representations, we are interested in learning a graph topology that well explains the observed signals on graphs. In this problem, to address the key challenge of understanding the relationships between the signal and the graph topology, we propose to adopt a factor analysis model for the graph signals and impose a Gaussian probabilistic prior on the latent variables that control these signals. Due to the properties of the graph Laplacian and the Gaussian prior, the proposed approach learns graphs that enforce the smoothness properties of the graph signals.

In summary, this thesis addresses several emerging research problems in modern information processing on graphs. We show that graphs of various forms can be used for the representations of complex and multi-view data, and novel signal processing and machine learning techniques can be combined to solve the associated learning and representation problems in efficient ways. The research efforts in this thesis would benefit applications such as community detection in mobile and social networks, mobility inference from traffic and social media data, and urban dynamics analysis.

Keywords: Multi-layer graphs, signals on graphs, clustering, classification, graph learning.

Résumé

Les récents algorithmes de traitement de l'information doivent souvent manipuler des données multi-vues complexes, qui proviennent de plusieurs sources d'information avec des structures spécifiques. Par exemple, les observations d'un même ensemble d'entités sont souvent effectuées depuis plusieurs angles, dans d'autres cas, des observations collectées individuellement par plusieurs senseurs sont intrinsèquement liées. L'objectif de traiter de telles données efficacement, ouvre de nouvelles problématiques en traitement du signal et en techniques d'apprentissage.

Dans cette thèse, nous formulons le traitement et l'analyse de données multi-vues comme des problèmes de traitement du signal et d'apprentissage définis sur des graphes. Les graphes constituent un outil mathématique attrayant pour modéliser des relations entre paires d'entités dans une collection de données. De plus, cet outil est flexible et adaptable afin d'introduire plusieurs sources d'information avec leurs structures. Par exemple, plusieurs types de relations entre des mêmes entités dans un certain ensemble peuvent être modélisés par un graphe à plusieurs niveaux, où les niveaux partagent le même ensemble de noeuds (entités) mais ont différentes connections (relations) entre eux. D'un autre côté, l'information d'une source peut être modélisée comme des signaux définis sur l'ensemble des noeuds d'un graphe, celui-ci correspondant à une autre source. Alors que la première approche constitue une extension des méthodes usuelles d'apprentissage basées graphes, la dernière s'inscrit dans un thème de recherche émergent : le traitement du signal sur les graphes. Dans cette thèse, nous ferons le lien entre ces deux visions en étudiant plusieurs problèmes liés au clustering, à la classification et à la représentation, pour des données de types variés associées à des graphes pondérés et non orientés.

Dans un premier temps, nous étudions le problème d'analyse de graphes multi-niveaux et nous proposons des méthodes pour le clustering des noeuds en combinant efficacement l'information fournie par les différents niveaux. Pour cela, nous proposons de combiner les caractéristiques de chaque niveau du graphe en utilisant des outils d'analyse de sous-espaces de variétés de Grassmann. La combinaison obtenue peut être vue comme une représentation de faible dimension des données originales, qui préserve une grande partie de l'information importante à partir des diverses relations entre les entités. Nous utilisons notre algorithme dans des méthodes de clustering et nous démontrons que notre méthode est meilleure que les algorithmes de référence et compétitive par rapport aux techniques de l'état de l'art.

Ensuite, nous nous penchons sur le problème qui consiste à combiner les différents niveaux en graphes multi-niveaux depuis différentes perspectives. Plus particulièrement, nous considérons les vecteurs propres de la matrice Laplacienne du graphe d'un niveau comme signaux sur le graphe d'un autre niveau. Nous proposons une approche de régularisation de graphe, qui produit un ensemble de "vecteurs propres joints", ou de "spectre joint", partagés entre deux niveaux. Nous utilisons ce

spectre joint dans des problèmes de clustering sur des graphes multi-niveaux. Par rapport à notre approche précédente et à la plupart des techniques de l'état de l'art, une caractéristique unique et donc un avantage potentiel de cette méthode, est qu'elle permet de combiner des niveaux, d'une manière convaincante, en se basant sur leur importance respective.

Troisièmement, à partir de l'approche précédente, nous étudions le problème de classification de signaux sur les graphes d'un point de vue général. Pour cela, nous adoptons des représentations de signaux efficaces définis dans le domaine spectral du graphe, et nous analysons théoriquement leur potentiel pour la classification des signaux du graphe sous un modèle de signaux génériques. Nous proposons ainsi des caractéristiques spectrales discriminantes et testons leurs performances pour des tâches de classification synthétiques, en les comparant à des méthodes extrayant les caractéristiques dans le domaine spatial. Nous étudions l'approche proposée de classification spectrale dans des applications telles que la déduction de modèles de mobilité dans les données de traffic ou de média sociaux.

Enfin, étant donné le rôle crucial d'un graphe significatif, qui n'est pas toujours naturellement défini suivant les domaines d'applications, dans les représentations de signal, nous nous intéressons à l'apprentissage de la topologie des graphes qui explique bien les signaux observés sur ceux-ci. Dans ce problème, pour résoudre le défi clef de la compréhension des relations entre le signal et la topologie du graphe, nous proposons d'adopter un modèle d'analyse de facteurs pour les signaux de graphes et nous imposons un a priori de gaussianité de la probabilité des variables latentes qui contrôlent ces signaux. Etant donné les propriétés du Laplacien de graphe et l'hypothèse de gaussianité, l'approche proposée apprend les graphes qui forcent les signaux à être lisses.

En résumé, cette thèse traite plusieurs problèmes de recherche émergents et actuels, pour le traitement de l'information sur les graphes. Nous montrons que des graphes de types variés peuvent être utilisés pour la représentation de données multi-vues complexes, et que des techniques nouvelles de traitement du signal et d'apprentissage peuvent être combinées pour résoudre les apprentissages et représentations correspondantes de manière efficace. Les efforts de recherche de cette thèse apportent un bénéfice aux applications comme la détection de communauté dans les réseaux sociaux mobiles, la déduction de mobilité dans les données de traffic ou de média sociaux, et enfin dans l'analyse de dynamiques urbaines.

Mots Clés : Graphes multi-niveaux, signaux définis sur des graphes, clustering, classification, apprentissage de graphes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Modern information processing inevitably involves an extremely large volume of increasingly complex data. The complexity comes, in particular, from the fact that we often need to deal with "multi-view" data, which usually come from multiple sources of information at the same time, with many relationships and structures among them. First, relationships between the same set of entities are often evaluated in different respects. For example, in a document database, similarities between documents can be defined in terms of their *titles*, *abstracts*, or *citations*. In a social network, there usually exist multiple types of relationships among the same group of persons, such as *friendship*, *family relationship*, *having the same professions*, and so on. Second, data observed by different sensors could be intrinsically related by some structures, where the data and the structures correspond to different sources of information. For instance, *temperatures* observed at different regions are related to their *geographical proximities*, *traffic volumes* at different locations in a transportation network depends on the *topology* of the network, and *behaviors* of a group of persons may be influenced by the *friendships* among them. To handle such multi-view data efficiently, we need to understand the interactions between different sources of information as well as the relationships and structures among them.

Graphs are powerful mathematical tools to model relationships and structures in the data. In a graph representation, the vertices represent the entities and the edges represent the pairwise relationships between these entities. More importantly, graph-based data representations are flexible and adaptable to incorporate multiple sources of information with relationships and structures among them, yet remaining sufficiently simple for efficient processing. For example, multiple types of relationships between the same set of entities can be conveniently represented by a set of weighted and undirected graphs that form a global **multi-layer graph**, where the common vertex set represents the entities and the edges on different layers capture the similarities of the entities in term of the different sources of information. Furthermore, data with structures can be modeled as a signal defined on the vertex set of a weighted and undirected graph, often referred to as **signals on graphs** or **graph signals**[1], where the signal value associated with each vertex represents the observation, and the graph captures the relationships between these observations. Both multi-layer

[1]Throughout this thesis, we use "signals on graphs" and "graph signals" interchangeably.

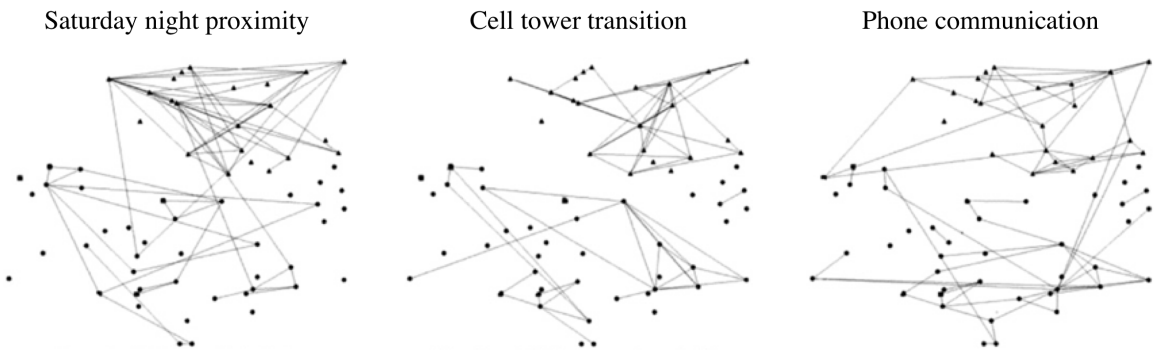


Figure 1.1: A three-layer graph in a mobile social network (figure from [42]), where vertices at the same position in the three figures represent a unique mobile user. On the left, two mobile users are connected with an edge if they were close physically to each other during one Saturday night. In the middle, two are linked together if they made the same series of cell tower transitions at relatively similar timestamps. On the right, an edge is assigned between each pair of users who had ever interacted by phone communication.

graphs and signals on graphs can be considered as representations for multi-view data in a general sense (since they both contain multiple sources of information), hence are the two data representations of interest in this thesis. Moreover, these graph-based representations highlight the close connections between traditionally different fields in the processing and analysis of multi-view data. Indeed, while a multi-layer graph can be considered as an extension to a single graph that has been extensively studied in machine learning [112], graph signals are the central focus of the emerging research field of signal processing on graphs [119, 109]. The main effort of this thesis is thus devoted to developing novel techniques in the intersection of signal processing and machine learning in order to solve several problems related to the representation, clustering, classification, and learning of multi-layer graphs and signals on graphs.

Graph-based learning techniques have been studied extensively in the machine learning community and widely applied in various domains in the last two decades. Compared to the traditional setting of a single graph in these techniques, multi-layer graphs present attractive advantages in representing and analyzing multiple types of relationships in the data. As an example in mobile and social network analysis, consider the cell phone data from the MIT Reality Mining project [40, 41] illustrated in Fig. 1.1, where three different types of relationships between the same group of mobile users can be captured at the same time by a three-layer graph, namely, Saturday night proximity, similarity in cell tower transitions, and phone communication. Intuitively, each layer corresponds to a certain type of human interaction, and a proper combination of these three layers is likely to provide a better understanding of the relationships between the mobile users than any layer taken in isolation. Our objective is thus to analyze such multi-layer graphs to gain insight into the intrinsic relationships between the entities in the dataset by properly combining the information in the individual layers. Compared to the traditional case of graph-based learning, the analysis of multi-layer graphs brings several new challenges. First, we need to define a principled and generic framework to study learning problems related to multi-layer graphs. Second, we need to analyze the multiple layers simultaneously, in order to efficiently merge the information contained in the individual lay-

ers. Ideally, we also would like to take into account the respective importance of these layers in the merging framework. In addition to the research challenges, we also need to demonstrate that the outcome of our learning framework is indeed beneficial in terms of specific learning tasks. In this thesis, we first propose a generic framework for learning with multi-layer graphs [36, 38]. We then propose a second framework that takes into account the respective importance of the individual layers in a unique learning process [35]. We show that the proposed frameworks lead to novel clustering algorithms with applications in mobile and social network analysis such as community detection. We believe that our frameworks are beneficial not only to specific learning problems such as clustering, but also to many other information processing tasks based on multi-layer graphs or data that contain multiple sources of information in general.

Signal processing on graphs as an emerging research field has recently attracted growing interests in the signal processing community. In this setting, the vertices of the graph represent entities and the edge weights reflect the pairwise relationships between them, while a graph signal assigns a scalar value to each vertex based on some observation associated with the entities. On the one hand, graph signals capture the relationships between the observations, thus reflect the structures in the data. On the other hand, the signals and the graphs are usually defined using different sources of information. Numerical examples of graph signals can be found in geographical, transportation, biomedical and social networks, such as temperatures within a geographical area, traffic capacities at hubs in a transportation network, or human behaviors in a social network. In particular, in many applications, various patterns of practical interest are present in the graph signals. For example, consider data collected from the Flickr social network, where the aggregated number of distinct Flickr users that have taken photos at different locations in a city is a signal defined on a city-scale geographical graph. In this case, it is clear that the signals in the local regions of the graph are related to mobility patterns of the Flickr users in those areas, as illustrated in Fig. 1.2. Our objective is thus to infer various patterns of interest present in such observations, by studying the classification problem of graph signals. Compared to traditional signal classification, the main research challenge in this new problem is, to introduce efficient and discriminative representations and appropriate models for graph signals, which are defined in irregular domains and are less studied than traditional signals. Furthermore, we need to demonstrate in real world applications the meaningfulness and efficiency of graph signal classification. In this thesis, we analyze the potential of signal representations defined in the graph spectral domain for classifying graph signals under a generic signal model, and then design discriminative spectral features for graph signal classification. We apply our framework to real world applications such as the inference of mobility patterns in traffic and social media data [37]. The proposed graph signal classification framework not only sheds new lights in the research of learning problems related to graph signals, but will also benefit a wide range of applications that involve structured data.

The efficiency of most of the graph-based learning and signal processing techniques rely on a meaningful graph that captures the relationships between the entities in the dataset. However, such a meaningful graph is not always readily available or easy to define. In practice, the graph is either assumed to be given a priori or defined based on heuristics. For example, a similarity graph is usually defined using a Gaussian radial basis function (RBF) based on Euclidean distances between the feature vectors. In graph signal processing research, graph signals are usually defined on natural graphs from the applications domain, such as a geographical graph or a social friendship graph. Although these choices are meaningful to some extent, they may not be able to explain the intrinsic

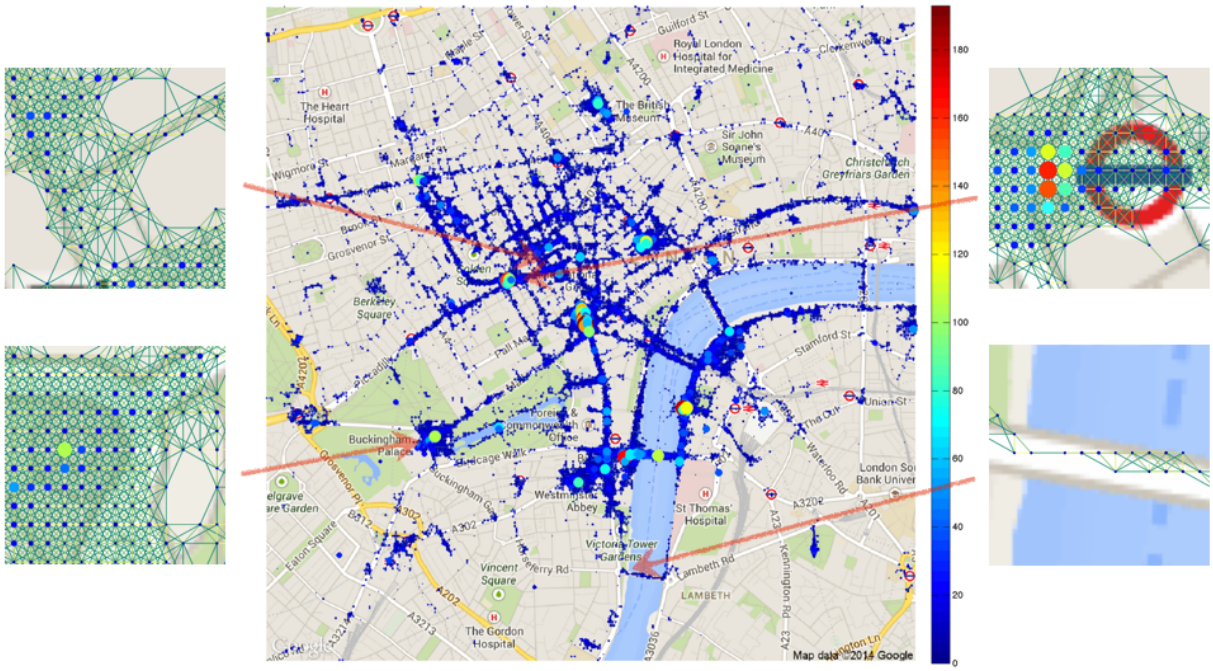


Figure 1.2: An example of the graph signal classification problem: This figure shows the aggregated number of distinct Flickr users that have taken photos at different locations in the central area of London from January 2010 to June 2012, as a signal defined on a geographical graph based on the physical distances between these locations. The size and color of the balls indicate the number of distinct users associated with the corresponding geographical positions. In this example, the signals observed in different local regions of the geographical graph correspond to different mobility patterns of the Flickr users.

relationships between entities in the data. Furthermore, there could be scenarios where there is no information for defining a suitable graph, such as network inference problems. Therefore, there is certainly a need for searching for a most appropriate graph representation given the observations. This is especially true in graph signal processing, given the crucial role of a meaningful graph in most of the state-of-the-art techniques in the field. Based on this understanding, we are interested in learning a meaningful graph topology given the observed graph signals. The key challenge here is twofolds, (i) to propose a learning criteria based on the understanding of the interactions between the signal values and the graph topologies, (ii) to show that the learned graph topology would enforce certain desirable properties of the observed signals. In this thesis, we propose to use a factor analysis model to understand the role of the graph in representing graph signals, and develop a framework to learn graphs that enforce the smoothness property of the signals. The proposed graph learning framework and the learning algorithm would surely lead to new theoretical development in the research field of signal processing on graphs, and are useful for learning similarities or relationships between data samples in general. It will also open possibilities in emerging real world applications, such as network analysis where it is beneficial to infer some hidden relationships between the entities based on observed data.

1.2 Thesis outline

The outline of this thesis is as follows.

In Chapter 2, we provide an extensive review of the related works in the literature that are related to this thesis. First, we summarize the state-of-the-art techniques in graph-based learning problems, with a special emphasis on recent advances in the analysis of multi-view data. Next, we report recent developments in the field of signal processing on graphs, and discuss especially those that concern the learning problems related to graph signals. Finally, we present existing methods for learning graph topologies given observed data.

In Chapter 3, we study the problem of learning with multi-layer graphs, where we are interested in efficiently combining the information provided by multiple graph layers for learning tasks such as clustering. To this end, we propose to model individual graph layers as subspaces on a Grassmann manifold, and combine their characteristics by seeking for a representative subspace on the manifold whose average distance to the individual subspaces is minimized. We consider the resulting subspace as a summarization of the information contained in the multi-layer graph, and use it for designing a novel clustering algorithm that partitions the vertex set of the graph into disjoint clusters. We justify our framework theoretically using results form statistical learning theory, and test the performance of the proposed clustering algorithm on several synthetic and real world datasets. Our algorithm is shown to be superior to baseline schemes and competitive to a state-of-the-art method, with a much simplified implementation and lower computational complexity.

In Chapter 4, we investigate the same learning problem on multi-layer graphs from a signal processing perspective. Our approach is motivated by the observation that, a set of smooth signals on the graphs, such as the eigenvectors of the graph Laplacian matrix, tends to contain the most important information in the graph. Therefore, in order to combine the information of two different graph layers, we propose to search for a set of signals that are jointly smooth on the two layers. Specifically, we consider the eigenvectors of the graph Laplacian from one layer as signals defined on another layer, on which their regularities are enforced by a novel graph regularization framework. The proposed framework leads to a set of "joint eigenvectors" shared by the two layers, which we call a "joint spectrum", that captures the characteristics of both layers. We then generalize this framework to multiple layers via a greedy information-theoretic approach. We use the proposed approach to design a novel clustering algorithm and show that it achieves competitive clustering performance compared to our previous approach in Chapter 3. More importantly, a unique characteristic of our method is that, individual layers play different roles in the merging process. This presents a potential advantage in multi-layer graph learning since it allows us to take into account the respective importance of the individual layers.

In Chapter 5, we follow the direction towards graph signal processing initiated in Chapter 4 and study the problem of classification of graph signals. To this end, we adopt efficient representations of graph signals defined in the graph spectral domain, especially via the recently proposed spectral graph wavelet transform [62], and analyze their potential for graph signal classification under a generic signal model, namely, the random walk model. We then propose discriminative spectral features and test their performances in synthetic classification tasks compared to PCA-based spectral features as well as features extracted in the spatial domain. As a concrete example, we study the problem of inference of mobility patterns, where we design synthetic models tailored for different signal patterns of interest, and apply the proposed spectral classification framework.

Both experiments on synthetically generated data and on real world mobility data demonstrate that localized spectral features have great potential in the classification of graph signals with noticeable local variations, such as those under the random walk model. The proposed classification framework is an important step towards solving learning problems related to graph signals in the field of signal processing on graphs.

Finally, in Chapter 6, we address the problem of learning meaningful graph topologies given observed signals. To understand the interaction between the signal and the graph, and the important role of the graph in representing the graph signals, we adopt a factor analysis model and establish connections between signal representations in the traditional setting and graph setting. By revisiting the representation learning theory [10], we show that a Gaussian probabilistic prior imposed on the latent variables in the factor analysis model leads to smooth signals on the graph. This allows us to design a framework for learning graph topologies given the assumption that the observed signals follow a multivariate Gaussian distribution. We apply the proposed graph learning framework to both synthetic and real world data, and demonstrate that it achieves better performance compared to a learning algorithm for estimating a sparse inverse covariance matrix for Gaussian graphical models. We further establish connections between graph learning and the problems addressed in the previous chapters.

In summary, we address in this thesis several important problems related to the processing and analysis of complex, multi-view, and structured data. Specifically, we first propose new methods for learning the intrinsic relationships between the entities given observations in the form of multi-layer graphs. We then propose a graph signal classification framework to capture various patterns of interest from graph signals. Finally, we introduce a framework for learning meaningful graph topologies given observed signals. On the one hand, the research effort presented in this thesis provides insight into developing novel signal processing and machine learning techniques for processing multi-view data on graphs. On the other hand, we show that it already benefits real world applications in transportation, mobile, and social network analysis.

1.3 Summary of contributions

We summarize the main contributions of this thesis as follows.

- We develop a principled and generic framework for analyzing and efficiently merging information contained in the different layers of a multi-layer graph. The proposed framework leads to a novel clustering algorithm that produces a unified clustering of the vertices in the graph, and achieves competitive performance compared to a state-of-the-art approach with a much simpler implementation and low computational complexity.

- We further propose a second framework for combining multiple graph layers, where individual layers play different roles in the merging process. Different from the approach in Chapter 3 and most of the state-of-the-art techniques, this allows us to take into account the respective importance of the individual layers in a unique and convincing way.

- We study the graph signal classification problem, where we show the potential of representation in the graph spectral domain for the classification of graph signals. We design a novel

algorithm for classifying graph signals that correspond to different patterns of interest and demonstrate its efficiency in real world applications such as the inference of mobility patterns.

- We propose a framework and an algorithm for learning graph topologies given observed signals. We establish theoretical connections between signal representations in the traditional setting and graph setting, by adopting a factor analysis model and a Gaussian probabilistic prior on the latent variables. We demonstrate in real world applications the meaningfulness of the proposed framework as well as the superior performance of our algorithm compared to a state-of-the-art approach.

This thesis provides solutions to several important research problems in processing and analyzing multi-view data on graphs, which would not only help meet the research challenges in modern information processing with complex data, but also benefit numerous emerging applications in transportation, biomedical and social network analysis.

Chapter 2

State of the art

2.1 Overview

In 1736, Swiss mathematician Leonhard Euler published a paper on the Königsberg Bridge problem, which is widely considered as the birth of graph theory. Originally a vibrant subject in discrete mathematics, graphs have been of increasing importance in computer science and modern information processing tasks, since they provide powerful mathematical tools for modeling pairwise relationships between a set of entities. Such relationships usually help in the understanding of networked data hence provide benefits to various analysis tasks. Over the last two decades, graph-based techniques in data mining, pattern recognition and machine learning have become extremely popular. The applications of these graph-based analysis are numerous, noticeably in transportation, biomedical and social network analysis. Recently, traditional signal processing techniques have also been generalized to handle signals defined on the vertex set of graphs. This brings new ingredients to traditional graph-based processing and analysis techniques, making graphs one of the most important analysis tools in the intersection of different research fields in modern data analysis.

At the heart of the techniques presented in this thesis is one particular branch of graph theory, namely, the spectral graph theory [29], which utilizes tools from matrix theory and linear algebra to study graph theory and its applications. Specifically, spectral graph theory focuses on the "spectrum" of the graph, namely, the eigenvalues of the adjacency matrix or the Laplacian matrix of the graph. In this thesis, we are particularly interested in the eigenvalues and eigenvectors of the Laplacian matrices of the graph. As we will see, they are of central importance to many graph-based machine learning and signal processing techniques, in particular, unsupervised learning (clustering and dimensionality reduction), semi-supervised and supervised learning (classification and regression), and spectral filtering in graph signal processing.

In this chapter, we review some of the most important techniques developed in the areas of graph-based signal processing and learning in the literature. First, we start in Section 2.2 with machine learning approaches on graphs, from the traditional setting of a single graph to more recent analysis problems on multi-layer graphs. Next, we move onto the emerging field of signal processing on graphs in Section 2.3, where we survey the latest developments for solving the approximation, reconstruction and learning problems related to the graph signals. Finally, given the central role

of the graph in these techniques, we present related works in the literature that propose to learn graph topologies from the observed data in Section 2.4.

2.2 Learning with graphs and multi-view data

2.2.1 Unsupervised and supervised learning on graphs

Graph-based unsupervised and supervised learning has been extensively studied in the machine learning and data mining communities since the last two decades, due to their numerous applications in various domains where pairwise relationships between entities are of special interest. In this section, we first describe some important works in the literature regarding learning techniques developed on graphs.

First, as a typical instance of unsupervised learning problems, clustering analysis have been studied extensively due to its numerous applications in different domains, and has benefited significantly from tools developed on graphs. The works in [112] and [48] have given comprehensive reviews of the state-of-the-art techniques and recent advances in graph-based clustering, ranging from hierarchical clustering to graph cuts, spectral methods and Markov chain based methods, and their applications to the problem of community detection. Clustering methods derived based on spectral analysis on the graph are of particular interest in this thesis. As a typical example, spectral clustering algorithms [116, 99, 134, 141] have become one of the most popular graph-based clustering techniques. These methods propose to embed the vertices of the graph into subspaces spanned by the first k eigenvectors, which correspond to the k smallest eigenvalues, of various graph Laplacian matrices. Due to the properties of the graph Laplacian matrix, this transformation reveals the intrinsic relationships between the original vertices. Consequently, clusters can be eventually detected in the resulting low dimensional space by many common clustering algorithms, such as the k-means algorithm [86]. Due to promising performance and close connections to other well-studied mathematical problems, a large number of variants of the original algorithm have been proposed, such as the constrained spectral clustering algorithms [138, 85, 81, 80, 135]. In general, these works have suggested different ways to incorporate prior constraints in the clustering tasks. It is worth noting that the graph Laplacian matrices are not the only choices for the embedding matrix. In fact, Newman has also proposed clustering methods based on modularity maximization via spectral methods [97, 98], which, instead of the graph Laplacian matrices, essentially use the first k eigenvectors of the modularity matrix of the graph as the embedding matrix.

Second, graphs have often been linked with another instance of the unsupervised learning problem, which is dimensionality reduction. As we have seen above, the success of spectral clustering algorithms can actually be attributed to a dimensionality reduction procedure, where data in the original space has been projected to a low dimensional space spanned by the first k eigenvectors of the graph Laplacian matrix. Another example is an interpretation of Principal Component Analysis (PCA) on graphs described in [108], which again links the graph structure to a subspace spanned by the top eigenvectors of the graph Laplacians. In [100], the authors have proposed to use dimensionality reduction to help improve the performance of spectral clustering by removing the noisy dimensions in the data. All these approaches show the close connections between tools developed on graphs and techniques proposed for unsupervised learning problems.

Third, graphs have also played an important role in supervised and semi-supervised learning

tasks, such as classification. As a reference, in [148] the author has discussed in detail semi-supervised learning approaches on graphs, from the label propagation algorithm and learning using Gaussian fields and harmonic functions, to active learning and kernel approaches. One of the basic assumptions in this problem is that the labels of data samples are in some way consistent with the way the corresponding vertices are connected in the graph. Algorithms like label propagation can then be applied to classify unlabeled vertices. Many semi-supervised learning techniques on graphs have been closely related to the regularization theory on graphs [123]. In [145, 144, 146], the authors have defined a family of differential operators on graphs, and used them to propose a "smoothness" measure of functions defined on graphs based on the graph Laplacian matrix. They have then proposed a discrete regularization framework for classification based on the proposed smoothness measure. In addition to regularization in the spatial (vertex) domain, the authors of [81] have also proposed a regularization framework in the graph spectral domain.

In summary, graphs have been an integral part of many classical machine learning and data mining algorithms developed in the last decade for solving various unsupervised and supervised learning problems. These algorithms have been widely applied in numerous application domains ranging from transportation and biomedical network analysis to social science. However, most of these techniques focus on a single graph that captures a certain type of relationships between the entities. To handle multiple types of relationships between the entities considered in this thesis, we need to generalize traditional graph models to the new setting.

2.2.2 Learning with multi-view data

Compared to a single graph representation, multi-layer graphs are convenient tools for representing multiple types of relationships between the entities, or more generally for representing multi-view data. These data form multi-layer graph representations (or multi-view representations), and appear sometimes in the context of learning with multiple kernels, which generally refer to data that can be analyzed from different viewpoints. Multi-view data usually provide a more complete picture about the entities in the dataset through multiple angles. Therefore, by combining data from different views we expect to gain an improved understanding of the intrinsic relationships between the entities or the true causes of the observations. In this section, we survey recent works in the literature that aims at solving learning problems related to multi-layer graphs or multi-view data in general. The key challenge in this problem is to combine efficiently the information from multiple graphs (or multiple views) for the learning objectives. The research effort towards this direction can be roughly grouped into the following categories.

First, the most straightforward way to combine multi-view data is to form a linear combination of the information from the individual views or graphs. For example, in [5], the authors have developed a method to learn an optimal convex combination of Laplacian kernels from different graphs. In [143], the authors have proposed a Markov mixture model for supervised and unsupervised learning, which can essentially be interpreted as a convex combination of the normalized adjacency matrices of the individual graphs. In [127], the authors have presented several averaging techniques for combining information from the individual graphs for clustering. Finally, in [90], the authors have proposed to combine multiple kernels by forming summations of weighted or projected kernels. As a straightforward idea, averaging or linear combinations can be efficient in some cases, but may not be able to capture the specificities in the individual views.

Second, built on the intuitive approach of a linear combination of information layers, many existing works aim at finding a unified representation of the multiple graphs (or multiple views), by using more sophisticated models and methods. For instance, several authors have developed joint matrix factorization approaches to combine matrix representations of the individual views of data, such as the adjacency matrix [128] or the Laplacian matrix [46] of the graph, through a unified optimization framework. In [137], the authors have proposed to find a unified spectral embedding of the original data by integrating information from different views. Similarly, clustering algorithms based on Canonical Correlation Analysis (CCA) first project the data from different views into a unified low dimensional subspace, and then apply simple algorithms like single-linkage or k-means to achieve the final clustering [14, 24]. In [136], the authors have proposed to consider the eigenspaces of the graph Laplacian of the individual graph layers as points on a Grassmann manifold and studied the distances between them for the possible combination. In [21], the authors have proposed to find a pair of closest commuting operators (CCO) to a given pair of graph Laplacian matrices, which are a pair of commuting matrices and thus have the same eigenbasis. They have further showed that their approach is equivalent to joint diagonalization but allows explicitly preserving the properties of the Laplacian matrices. The approaches in this category can potentially be more meaningful than a simple averaging scheme, if the learned unified representation better captures the relationships between different views in the multi-view data.

Third, unlike the previous methods that try to find a unified representation before applying learning techniques, another strategy in the literature is to integrate the information from individual graphs (views) directly into the optimization problem for the learning purposes. Examples include the co-EM clustering algorithm proposed in [13], and the clustering approaches proposed in [71, 73] based on the frameworks of co-training [15] and co-regularization [122], respectively. These methods can be useful when a unified representation for the multiple views is not easy to find from the data.

Last but not least, other representative approaches include (i) the works in [11] where the authors have generalized several concepts on a single graph, such as the degrees and neighbors, to a multidimensional (multi-layer) graph, (ii) the works in [61, 33, 92, 133] where the authors have defined additional graph representations to incorporate information from the original individual graphs, (iii) the works in [91, 111] where the authors have worked with coupled graphs (a set of graphs that have the same vertex set with possible links between the same vertex across different graphs) or paths defined in multigraphs (graphs that allow multiple edges between the same vertex pair, which is essentially the same as a multi-layer graph in our terminology), and finally (iv) the works in [126, 22, 56, 26] where the authors have proposed ensemble clustering approaches by integrating clustering results from individual views.

To summarize, the above mentioned works for learning with multi-layer graphs or multi-view data have tackled the problem of combining information contained in multiple graphs or views from different perspectives, usually based on different learning criteria. Such criteria can be driven by specific forms of optimization problems, or based on some prior knowledge about the interactions between different views or graphs. However, a principled and generic framework is still needed for learning multi-view data. The proposed approach in Chapter 3 of this thesis is motivated by this observation, and belongs to the second category mentioned above, where we first find a representative subspace for the information provided by a multi-layer graph, and then we implement clustering or other learning tasks. We believe that this type of approaches is intuitive, yet remaining flexible and generic enough to be applied to different types of data. Furthermore, most of the

methods in the literature treat the individual views of the data in an equal manner. This motivates us to propose a graph regularization framework in Chapter 4 to better take into account the respective importance of the individual layers in a multi-layer graph.

2.3 Signal processing on graphs

2.3.1 Representation and approximation of signals on graphs

Many real world signals are intrinsically structured, such as temperatures in different geographical areas, traffic volumes at different locations in transportation networks, or human behaviors in social networks. Such structures can usually be represented by networks or graphs, which explain the intrinsic relationships between the observations made from individual sensors. It is thus beneficial to take into account such structures in the processing of the signals. Motivated by this observation, the understanding, representation and processing of signals defined on the vertex set of weighted and undirected graphs (namely, graph signals) have recently attracted increasing attention in the signal processing community, leading to the emerging field of signal processing on graphs [119, 109]. The main challenge in this field is to deal with the irregular and discrete support of the signals; tools from spectral graph theory have proved crucial in such treatment.

The fundamental analogy between traditional signal processing and graph signal processing is established through the eigenvalues and eigenvectors of the graph Laplacian matrix, which carry a notion of frequency for graph signals and leads to the generalization of traditional signal processing techniques to the setting of graph signal processing. In [62], the authors have first proposed the concept of a graph Fourier transform, which generalizes the traditional Fourier analysis to the graph setting. Some properties of the graph Fourier transform have been analyzed in [149]. In [120], the authors have generalized the fundamental concepts of traditional signal processing techniques, such as the convolution, translation and modulation operators, to graph signal processing. Based on the generalization of these operators, they have proposed a windowed graph Fourier transform, analogous to the classical short-time Fourier transform. They have then analyzed the localization of the proposed transform in the vertex-frequency domain, similarly to the classical time-frequency analysis. Along the same line of ideas, the authors of [4] have proposed a spectral graph uncertainty principle that provides a tradeoff between the localization of graph signals in the graph (vertex) and spectral (frequency) domain. Their work has been followed by the work in [102] where the authors have studied the same localization property through the uncertainty curves.

The generalization of the wavelet-like and multiscale transforms to the graph setting is of particular interests in the graph signal processing community. Classical wavelet transforms have proved to be appealing in many signal processing tasks due to their properties of being localized in both time and frequency, and of providing a notion of multiscale representation of the signals. Therefore, a large amount of effort has been devoted to the designs of wavelet transforms on graphs, both in the vertex domain and graph spectral domain. Representative examples in the former case include the wavelet transform based on the shortest path distance between vertices proposed in [31], the lifting-based transforms proposed in [93], and the tree-based transforms proposed in [50, 103, 104, 107]. The examples in the latter case are of special interest in this thesis, especially the spectral graph wavelets proposed in [62]. The spectral graph wavelet transform is defined in precise analogy to the classical continuous wavelet transform, based on the dilation of a series of

band-pass filters defined in the graph spectral domain. These filters provide a clear interpretation of wavelets at different scales and are computationally efficient to implement. The idea has then been generalized to adapt to the spectrum of a particular graph in [121] and to form a tight frame for multislice (multi-layer) graphs in [77]. Other noticeable examples include the diffusion wavelets proposed in [87, 30] and the diffusion wavelets packets in [19], which are constructed based on a diffusion operator; the graph-QMF filterbanks proposed in [94] and its biorthorgonal extension in [95], which are constructed based on two-channel filterbanks on graphs; and the multiscale transform proposed in [118] based on operators for signal filtering, graph downsampling and graph reduction. These transforms, like their highly-successful counterparts in the Euclidean domain, are demonstrated to be efficient in analyzing theoretically the behaviors and properties of graph signals with respect to the graph topology.

Most of the transforms mentioned above are defined based on the structure of the graph topology, and do not adapt specifically to the observed signals. In particular, they do not necessarily guarantee a sparse representation of the signals. Dictionaries [131], on the other hand, are representations that are usually learned from and adapted to the observations; they are aimed at providing sparse representations of the signals. Therefore, in order to adapt to the behaviors of a particular set of graph signals and achieve sparse representations, several works have tried to generalize dictionary representations on graphs. In particular, in [140], the authors have proposed a way to learn a set of modified spectrums to form structured graph dictionaries that are adapted to the given graph signals. In [129], the authors have proposed a form of dictionaries that are polynomials of the normalized graph Laplacian matrix, which guarantees the localization of the atoms in the vertex domain.

The design of efficient transforms and dictionaries has been the focus of the research in graph signal processing. In particular, compared to the traditional focus of mathematical transforms, dictionary representation on graphs has attracted an increasing attention in the field, due to its flexibility and adaptivity to the observed signals. On the one hand, these transforms and dictionaries are mainly aimed at solving classical signal processing problems on graphs, such as the representation, approximation and reconstruction of graph signals. Applications of them can already be found in various fields, such as wireless and sensor network analysis [79, 150] and image and video processing, in particular depth map coding [115, 114], predictive transform coding [139] and multi-view image coding [89]. On the other hand, relatively less research effort has been devoted to the learning problems related to graph signals and their potential applications. This is the motivation behind the graph signal classification problem considered in this thesis.

2.3.2 Learning with signals on graphs

As we have mentioned, the methods discussed in the previous section have mainly been developed for solving classical approximation and reconstruction problems related to the graph signals. It is thus particularly interesting to notice that the graph transforms and dictionaries have also been applied to graph-based learning problems. For example, in [132], the authors have exploited the properties of spectral graph wavelets to study the cluster relationships between vertices at different scales. Of particular interest in this thesis are the works that apply these graph signal processing tools in the context of regression and classification. For instance, in [117] the authors have proposed to use spectral graph wavelets for regression via semi-supervised learning, by imposing a sparse

representation of the label signal under the spectral graph wavelet transform through a L^1-norm penalty on the wavelet coefficients. Similarly, in [96], the authors have proposed two methods for interpolation of graph signals with partial observations based on localized graph filtering. In [110] the authors have defined total variation of a signal on the graph and use it as a way of regularization in learning the class labels. Similarly, in [44] the authors have proposed a regularization framework for learning the label function. Specifically, they have imposed a sparsity regularizer constraint on the wavelet decomposition of the label function. Finally, in [25], the authors have proposed to use adaptive graph filters for data classification by learning the filter coefficients through a semi-supervised optimization framework. The common aspect in these approaches is that they consider a representation graph to capture the pairwise relationships between the entities or data samples, and model the regression or label function as a graph signal. Therefore, they essentially solve the typical graph-based classification problems in the same spirit as approaches in [146, 148] where similar samples are assumed to have similar labels. The difference with the approaches in Section 2.2.1 is that the regression or label function is learned by using some graph-based filters and transforms, which might be more adaptable to the specific behaviors of these functions.

From another perspective, very few works in the literature have tackled the classification problem of graph signals, in which the graph signals are considered as data samples to classify. This can be beneficial in real world applications where the observed graph signals carry different patterns of interest, such as the example shown in Fig. 1.2. From this aspect, the classification frameworks proposed in [65] and [32] are the most closely related works in the literature, for that they also aim at distinguishing data samples that are graph signals in a broad sense. Specifically, the former approach uses the graph Fourier transform for matched signal detection through hypothesis tests, which is essentially a two-class classification problem based on the Fourier coefficients of the graph signal. The latter approach uses the spectral graph wavelet transform in shape analysis, where the features for classification are defined by simply taking a low-dimensional projection of all the wavelet coefficients.

Learning problems related to graph signals are important research problems to study, as the developed learning techniques help us leverage graph signal processing tools to handle the learning of structured data, which is sometimes not easy via traditional means. It would also benefit numerous applications in networked data analysis. However, in the field of signal processing on graphs, a large amount of works has been devoted to the representation and approximation of the graph signals, while the learning aspects have been relatively overlooked. Especially, problems related to the classification and clustering of graph signals have not been much studied. Preliminary works in the literature, such as [65] and [32], have not provided a complete analysis and generic framework for classifying graph signals. These are the main motivations behind the classification studies presented in Chapter 5.

2.4 Learning graphs from data

In the graph-based machine learning and signal processing techniques mentioned above, the graph provides a similarity measure between the entities in the dataset, which is crucial for the learning and data processing objectives. This is especially true in the field of signal processing on graphs, where the edges in the graph and their weights usually influence the signal values associated with

the vertices. Ideally, the graph should capture the true causes behind the relationships of the observed signals; therefore, different choices of the graph lead to different interpretations of the signals. For example, a smooth graph signal may become highly non-smooth if the graph topology has been changed. Given the central role of a meaningful graph for efficient signal representations, there is clearly a need for defining or choosing the graph appropriately.

However, most of the research effort in the signal processing community so far has been devoted to the representation and processing of the signals, while the graph is either given a priori, or chosen naturally form the application domain. Given the graph signals, relatively less research effort has been devoted to the analysis and learning of the graph topologies. One of the most closely related works in learning graph topologies has been proposed in [64], where a regression framework is proposed to learn a sparse graph topology based on a fitness metric between the signals and the graph. Another two examples are the works of [76] and [78], where the authors have proposed to use correlations between wavelet coefficients of the time-series of brain signals, and Principal Component Analysis, respectively, to estimate functional connectivities of distinct brain regions, which can be considered as the learning of an activity graph between the brain regions. These examples all focus on medical analysis applications. Finally, in [139] the authors have tried to make the link between the precision matrix in a Gaussian Markov Random Field and the graph Laplacian matrix. They however have not considered explicitly the structure of the graph Laplacian and have not proposed specifically an algorithm to learn a graph topology.

In the meantime, there is a large amount of work from the statistical machine learning community that aims at solving similar problems, but from a slightly different perspective. For example, one topic in the research of learning graphical models is to estimate a full-rank inverse covariance matrix from the observed data for Gaussian graphical models [6, 49], especially in the case when the number of observations is smaller than the sample dimension and the sample covariance becomes singular; or, to infer the graph structure for discrete Markov Random Fields [82]. It is known that in case of a Gaussian graphical model, there is an exact correspondence between the location of the non-zero entries in the precision matrix and the existence of partial correlations between the random variables [106]. In this case, a maximum-likelihood estimator turns out to be the solution to a log-determinant program. The estimated precision matrix is therefore considered to carry information about the partial corrections between the random variables. However, it is worth noting that the learned precision matrix, which is full-rank and usually has both positive and negative correlations between the random variables, is not a valid graph Laplacian matrix. The overall process cannot thus be interpreted precisely as learning a graph topology. Another line of works is metric and kernel learning, whose objective is to learn the similarities or dissimilarities between a set of entities. For example, in [66] the authors have proposed a general optimization framework for learning metrics via linear transformations, and analyzed as a special case the minimization of the log-determinant divergence subject to linear constraints. Given the statistical nature of these problems (for instance, in a graphical model the graph denotes the conditional independence structure between the random variables), most of the approaches tackle the problem either from a statistical point of view, for example based on a maximum-likelihood estimation (MLE), or from an information-theoretic point of view, for example based on the Bayesian information criterion (BIC). An essential difference with graph learning in graph signal processing is, however, that these approaches do not study the learning problem from the viewpoint of the processing of the graph signals. Specifically, they focus on the correlations between random variables, but do not pay much

attention to the analysis of the characteristics and properties of signals defined on the graph, such as the smoothness or sparsity of the data on the estimated graph. The latter is however the central and ultimate objective in the field of graph signal processing.

To summarize, compared to the representation and processing of the graph signals, the problem of learning graph topologies is an important area that has been relatively overlooked in the signal processing community. Although approaches for solving similar problems exist in the statistical machine learning community, these approaches do not learn graph topologies that are linked to the desired properties of the observed signals. The proposed learning framework in Chapter 6 is motivated by these observations, and is designed to enforce such property of the signals in their representations through the learned graph topology.

2.5 Summary

Based on the review above we would like to point out the following limitations of the related works in the literature, which are the main motivations of the approaches presented in the next four chapters:

- A principled and generic framework is needed for formulating and solving learning problems with multi-layer graphs.
- In most of the state-of-the-art algorithms for combining multiple graphs, different graphs are either treated equally or combined simply through a weighted summation. A novel framework is needed for taking into account the different roles and importance of the individual graphs in the combination.
- The influence of the relationships between the individual graph layers and their respective importance have not been much studied.
- Most of the works related to classification in graph signal processing model the label function as a graph signal. A complete analysis and generic framework for classifying data samples that are graph signals is clearly needed.
- There is little work in the literature that discusses the modeling of graph signals, which is crucial in theoretical classification analysis.
- Most of the research effort in graph signal processing have been devoted to the representation and approximation of the signals. Given the central role of a meaningful graph topology, there is clearly a need for more complete studies about learning optimal graph topologies given observed graph signals.

Chapter 3

Learning with multi-layer graphs: Subspace-based methods

3.1 Introduction

In this chapter, we consider the problem of learning with multi-layer graphs, where the individual layers share a common set of vertices but have different edge weights depending on the type of information in each layer. Assuming that all the graph layers are informative, they are likely to provide complementary information and thus to offer richer information than any single layer taken in isolation. We thus expect that a proper combination of the information contained in the different layers leads to an improved understanding of the structure of the data and the relationships between entities in the dataset.

Specifically, we consider a M-layer graph G with individual graph layers $G_i = \{V, E_i, \omega_i\}$, $i = 1, \ldots, M$, where V represents the common vertex set and E_i represents the edge set in the i-th individual graph G_i with associated edge weights ω_i. An example of a three-layer graph is shown in Fig. 3.1(a), where the three graph layers share the same set of 12 vertices but with different edges (we assume unit edge weights for the sake of simplicity). Clearly, different graph layers capture different types of relationships between the vertices, and our objective is to find a method that properly combines the information in these different layers. We first adopt a subspace representation for the information provided by the individual graph layers, which is inspired by the spectral clustering algorithms [116, 99, 134]. We then propose a novel method for combining the multiple subspace representations into one representative subspace. Specifically, we model each graph layer as a subspace on a Grassmann manifold. The problem of combining multiple graph layers is then transformed into the problem of efficiently merging different subspaces on a Grassmann manifold. To this end, we study the distances between the subspaces and develop a new framework to merge the subspaces where the overall distance between the representative subspace and the individual subspaces is minimized. We further show that our framework is well justified by results from statistical learning theory [60, 57]. The proposed method is a dimensionality reduction algorithm for the original data; it leads to a summarization of the information contained in the multiple graph layers, which reveals the intrinsic relationships between the vertices in the multi-layer graph.

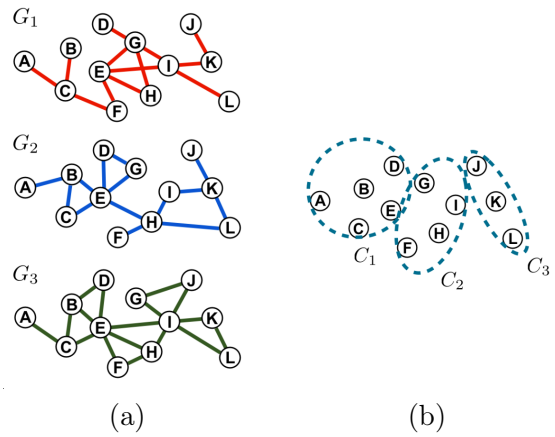


Figure 3.1: (a) An illustration for a three-layer graph G, whose three layers $\{G_i\}_{i=1}^3$ share the same set of vertices but with different edges. (b) A potential unified clustering $\{C_k\}_{k=1}^3$ of the vertices based on the information provided by the three layers.

Various learning problems can then be solved using these relationships, such as classification or clustering. Specifically, we focus in this chapter on the clustering problem: we want to find a unified clustering of the vertices (as illustrated in Fig. 3.1(b)) by utilizing the representative subspace, such that it is better than clustering achieved on any of the graph layers G_i independently. To address this problem, we first apply our generic framework of subspace analysis on the Grassmann manifold to compute a meaningful summarization (as a representative subspace) of information contained in the individual graph layers. We then implement a spectral clustering algorithm based on the representative subspace. Experiments on synthetic and real world datasets demonstrate the advantages of our approach compared to baseline algorithms, like the summation of individual graphs [128], as well as state-of-the-art techniques, such as co-regularization [73]. Finally, we believe that our framework is beneficial not only to clustering, but also to many other data processing tasks based on multi-layer graphs or multi-view data in general.

Subspace-based methods have been widely used in clustering and classification problems since the last two decades, most notably in image processing and computer vision. Due to the growing interests in this field, there is an increasingly large number of works that use tools from the Grassmann manifold theory, which provides a natural tool for subspace-based analysis. In [43], the authors have given a detailed overview of the basics of the Grassmann manifold theory, and developed new optimization techniques on the Grassmann manifold. In [27], the author has presented statistical analysis on the Grassmann manifold. Both works study the distances on the Grassmann manifold. In [59, 60], the authors have proposed learning frameworks based on distance analysis and positive semidefinite kernels defined on the Grassmann manifold. These subspace-based methods on the Grassmann manifold are certainly the inspirations behind the work presented in this chapter.

This chapter is organized as follows. In Section 3.2, we briefly review the spectral clustering algorithms, and describe the subspace representation inspired by spectral clustering, which captures the characteristics of a single graph. In Section 3.3, we introduce the main ingredients of Grassmann

manifold theory, and propose a new framework for combining information from multiple graph layers. We then propose our novel algorithm for clustering on multi-layer graphs in Section 3.4, and compare its performance with other clustering methods on multiple graphs in Section 3.5. Finally, we conclude in Section 3.6.

3.2 Spectral clustering and subspace representation for graphs

In this section, we give a very brief review of the spectral clustering algorithms applied on a single graph, which inspired us to define a subspace representation for graphs. Spectral clustering has become increasingly popular due to its simple implementation and promising performance in many graph-based clustering problems. It can be described as follows. Let us consider a weighted and undirected graph[1] $G(V, E, w)$, where V represents the vertex set of n vertices, E represents the edge set, and $w : E \to R^+$ associates each edge with a positive weight. Without loss of generality, we assume that the graph is connected. The adjacency matrix W of the graph is an $n \times n$ symmetric matrix defined as:

$$W_{ij} = \begin{cases} w_{ij} & \text{if there is an edge between } i \text{ and } j \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

The degree of a vertex i, denoted as $d(i)$, is defined as the sum of weights of all the edges incident to i in the graph. The degree matrix D is then defined as:

$$D_{ij} = \begin{cases} d(i) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

Based on W and D, the Laplacian matrix of the graph L is defined as:

$$L = D - W. \tag{3.3}$$

In the literature, L is usually called the unnormalized or combinatorial graph Laplacian matrix. There are two normalized versions of the graph Laplacian defined as follows:

$$L_{\text{sym}} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}, \tag{3.4}$$

$$L_{\text{rw}} = D^{-1}(D - W). \tag{3.5}$$

The graph Laplacian is of broad interests in the studies of spectral graph theory [29]. Although closely related to each other, different versions of the graph Laplacian have different properties that are desirable in different applications, as we shall see in the different chapters of this thesis[2]. For example, L and L_{sym} are real and symmetric matrices, therefore they both have a complete set of orthonormal eigenvectors with the associated eigenvalues. In the meantime, L_{rw} is not symmetric but has close connection to the random walk process on graphs. Different choices of the graph Laplacian correspond to different versions of the spectral clustering algorithm and detailed

[1] Throughout Chapter 3 and Chapter 4, the notation G (without a superscript or subscript) is used for a single graph unless it is clearly indicated that a multi-layer graph is considered.

[2] Throughout this thesis, we use L, L_{sym}, and L_{rw} as the unnormalized graph Laplacian, the symmetric normalized graph Laplacian, and the graph Laplacian closely related to a random walk, respectively.

discussion on these choices is given in von Luxburg [134]. In this Chapter, we use the normalized graph Laplacian L_{sym} defined above, since it has orthogonal eigenvectors and the eigenvalues are bounded between 0 and 2, a property favorable in comparing different graph layers in the following sections.

We now consider the problem of clustering the vertices i of G into k distinct subsets such that the vertices in the same subset are similar, i.e., they are connected by edges of large weights. This problem can be efficiently solved by the spectral clustering algorithms. Specifically, we focus on the algorithm proposed in [99], which solves the following trace minimization problem:

$$\min_{U \in \mathbb{R}^{n \times k}} tr(U^T L_{\text{sym}} U), \quad \text{s.t.} \quad U^T U = I, \tag{3.6}$$

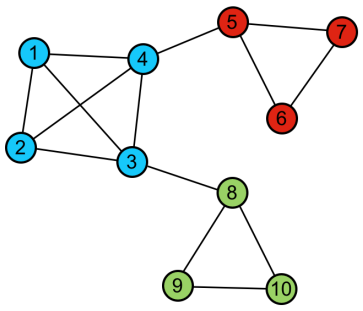
where n is the number of vertices in the graph, k is the target number of clusters, and $(\cdot)^T$ denotes the matrix transpose operator. It can be shown by a version of the Rayleigh-Ritz theorem [134] that the solution U to the problem of Eq. (3.6) contains the first k eigenvectors (which correspond to the k smallest eigenvalues) of L_{sym} as columns. The clustering of the vertices in G is then achieved by applying the k-means algorithm [86] to the normalized row vectors of the matrix[3] U. This algorithm is summarized in Algorithm 1. As shown in [134], the behavior of spectral clustering can be explained theoretically with analogies to several well-known mathematical problems, such as the normalized graph-cut problem [116], the random walk process on graphs [83], and problems in perturbation theory [125, 12].

Algorithm 1 Normalized Spectral Clustering [99]

1: **Input:**
   W: the $n \times n$ weighted adjacency matrix of graph G
   k: target number of clusters
2: Compute the degree matrix D and the normalized graph Laplacian matrix $L_{\text{sym}} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$.
3: Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the first k eigenvectors $u_1, \ldots, u_k$ of L_{sym} as columns (solution of (3.6)). Normalize each row of U to get U_{norm}.
4: Let $y_j \in \mathbb{R}^k$ $(j = 1, \ldots, n)$ be the transpose of the j-th row of U_{norm}.
5: Cluster y_j in $\mathbb{R}^k$ into k clusters $C_1, \ldots, C_k$ using the k-means algorithm.
6: **Output:**
   $C_1, \ldots, C_k$: the cluster assignment

We provide an illustrative example of Algorithm 1. Consider a single graph in Fig. 3.2(a) with ten vertices that belong to three distinct clusters (i.e., n=10 and k=3). For the sake of simplicity, all the edge weights are set to 1. The low dimensional matrix U that solves the problem of Eq. (3.6), which contains k orthonormal eigenvectors of the graph Laplacian L_{sym} as columns, is shown in Fig. 3.2(b). The matrix U is usually called the spectral embedding of the vertices, as each row of U can be viewed as the set of coordinates of the corresponding vertex in the k-dimensional space. More importantly, due to the properties of the graph Laplacian matrix, such an embedding

[3]The necessity for row normalization is discussed in [134] and we omit this discussion here. However, the normalization does not change the nature of spectral embedding, hence, it does not affect our derivation later.



(a)

$$\begin{pmatrix} -0.3273 & 0 & -0.3918 \\ -0.3273 & 0 & -0.3918 \\ -0.3780 & -0.1002 & -0.3048 \\ -0.3780 & 0.1002 & -0.3048 \\ -0.3273 & 0.3999 & 0.2213 \\ -0.2673 & 0.4062 & 0.3199 \\ -0.2673 & 0.4062 & 0.3199 \\ -0.3273 & -0.3999 & 0.2213 \\ -0.2673 & -0.4062 & 0.3199 \\ -0.2673 & -0.4062 & 0.3199 \end{pmatrix}$$

(b)

Figure 3.2: An illustration of spectral clustering. (a) A graph with three clusters (color-coded) of vertices; (b) Spectral embedding of the vertices computed from the graph Laplacian matrix. The vertices in the same cluster are mapped to coordinates that are close to each other in $\mathbb{R}^3$.

preserves the connectivity of the vertices in the original graph. In other words, two vertices that are strongly connected in the graph are mapped to two vectors (i.e., rows of U) that are close too in the k-dimensional space. As a result, a simple k-means algorithm can be applied to the normalized row vectors of U to achieve the final clustering of the vertices.

Inspired by the spectral clustering theory, one can define a meaningful subspace representation of the vertex connectivity in a graph using its k-dimensional spectral embedding, which is driven by the matrix U built on the first k eigenvectors of the graph Laplacian L_{sym}. Each row being the coordinates of the corresponding vertex in the low dimensional subspace, this representation contains the information on the connectivity of the vertices in the original graph. Such information can be used for finding clusters of the vertices, as shown above, but it is also useful for other analysis tasks on graphs. By adopting this subspace representation that "summarizes" the graph information, multiple graph layers can naturally be represented by multiple such subspaces (whose geometrical relationships can be quite flexible). The task of multi-layer graph analysis can then be transformed into the problem of effective combination of the multiple subspaces. This is the focus of the next section.

3.3 Merging subspaces via analysis on the Grassmann manifold

We now discuss the problem of effectively combining multiple graph layers by merging multiple subspaces. The theory of Grassmann manifold provides a natural framework for such a problem. In this section, we first review the main ingredients of the Grassmann manifold theory, and then move onto our generic framework for merging subspaces.

3.3.1 Ingredients of Grassmann manifold theory

By definition, a Grassmann manifold $\mathcal{G}(k, n)$ is the set of k-dimensional linear subspaces in $\mathbb{R}^n$, where each unique subspace is mapped to a unique point on the manifold. The advantage of using tools from Grassmann manifold theory is thus two-fold: (i) it provides a natural and intuitive

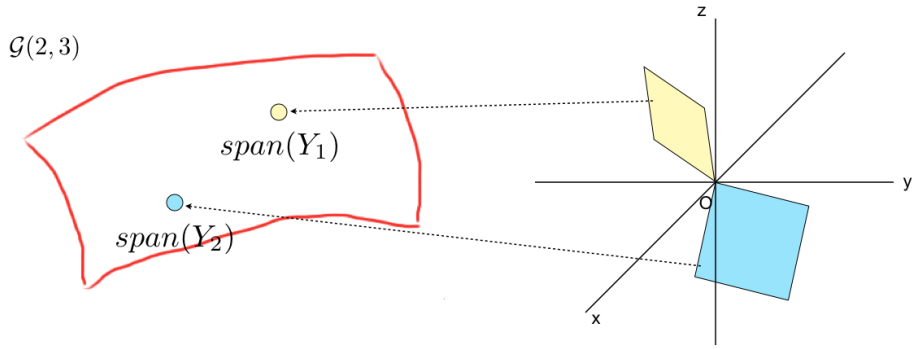


Figure 3.3: An example of two 2-dimensional subspaces $span(Y_1)$ and $span(Y_2)$ in $\mathbb{R}^3$, which are mapped to two points on the Grassmann manifold $\mathcal{G}(2,3)$.

representation for our problem: the subspaces representing the individual graph layers can be considered as individual points on the Grassmann manifold[4]; (ii) the analysis on the Grassmann manifold permits to use efficient tools to study the distances between points on the manifold, namely, distances between different subspaces. Such distances play an important role in the problem of merging the information from multiple graph layers.

Mathematically speaking, each point on $\mathcal{G}(k,n)$ can be represented by an orthonormal matrix $Y \in \mathbb{R}^{n\times k}$ whose columns span the corresponding k-dimensional subspace in $\mathbb{R}^n$; it is thus denoted as $span(Y)$. As an example, Fig. 3.3 shows two subspaces, represented by two orthonormal matrices Y_1 and Y_2, as two points on the Grassmann manifold $\mathcal{G}(2,3)$. The distance between two points on the manifold, or between two subspaces $span(Y_1)$ and $span(Y_2)$, is then defined based on a set of principal angles $\{\theta_i\}_{i=1}^k$ between these subspaces [53]. Specifically, θ_1 is defined as the smallest possible angle between all pairs of unit vectors z_i and z_j that come from $span(Y_1)$ and $span(Y_2)$ respectively. The other principal angles are defined recursively as the smallest possible angle between all pairs of unit vectors that (i) come from $span(Y_1)$ and $span(Y_2)$ and (ii) are orthogonal to all the previously selected z_i and z_j. By definition, the principal angles measure how the subspaces are geometrically close to each other, and are the fundamental measures used to define various distances on the Grassmann manifold, such as the Riemannian (geodesic) distance or the projection distance [43, 59]. In this chapter, we use the projection distance, which is defined as:

$$d_{\text{proj}}(Y_1,Y_2) = (\sum_{i=1}^{k} \sin^2\theta_i)^{\frac{1}{2}}, \tag{3.7}$$

where Y_1 and Y_2 are the orthonormal matrices representing the two subspaces under comparison[5]. The reason for choosing the projection distance is two-fold: (i) the projection distance is defined as the L^2-norm of the vector of sines of the principal angles. Since it uses all the principal angles, it is therefore an unbiased definition. This is favorable as we do not assume prior knowledge on the

[4]Different graph layers naturally lead to different points on the manifold; However, we do not specifically exclude the case where there exist two graph layers that are exactly the same.

[5]In the special case where Y_1 and Y_2 represent the same subspace, we have $d_{\text{proj}}(Y_1,Y_2) = 0$.

importance of specific principal angles, and we consider that all of them carry meaningful information; (ii) the projection distance can be interpreted using a one-to-one mapping that preserves distinctness: $span(Y) \rightarrow YY^T \in \mathbb{R}^{n\times n}$. Note that the squared projection distance can be rewritten as:

$$\begin{aligned} d^2_{\text{proj}}(Y_1, Y_2) &= \sum_{i=1}^{k} \sin^2\theta_i = k - \sum_{i=1}^{k} \cos^2\theta_i \\ &= k - tr(Y_1 {Y_1}^T Y_2 {Y_2}^T) \qquad (3.8) \\ &= \frac{1}{2}[2k - 2tr(Y_1 {Y_1}^T Y_2 {Y_2}^T)] \\ &= \frac{1}{2}[tr(Y_1 {Y_1}^T) + tr(Y_2 {Y_2}^T) - 2tr(Y_1 {Y_1}^T Y_2 {Y_2}^T)] \\ &= \frac{1}{2}||Y_1 {Y_1}^T - Y_2 {Y_2}^T||^2_F, \qquad (3.9) \end{aligned}$$

where the third equality comes from the definition of the principal angles and the fifth equality uses the fact that Y_1 and Y_2 are orthonormal matrices, and the operator $||\cdot||_F$ denote the Frobenius norm. It can be seen from Eq. (3.9) that the projection distance can be related to the Frobenius norm of the difference between the mappings of the two subspaces $span(Y_1)$ and $span(Y_2)$ in $\mathbb{R}^{n\times n}$. Because the mapping preserves distinctness, it is natural to take the projection distance as a proper distance measure between subspaces. Moreover, Eq. (3.8) provides an explicit way of computing the projection distance between two subspaces from their matrix representations Y_1 and Y_2. We are going to use it in developing the generic merging framework in the following section.

3.3.2 Layer merging framework

Equipped with the subspace representation for individual graphs and with a distance measure to compare different subspaces, we are now ready to present our generic framework for merging the information from multiple graph layers. Given a multi-layer graph G with M individual layers $\{G_i\}_{i=1}^M$, we first compute the graph Laplacian matrix L_{sym_i} for each G_i and then represent each G_i by the spectral embedding matrix $U_i \in \mathbb{R}^{n\times k}$ from the first k eigenvectors of L_{sym_i}, where n is the number of vertices and k is the target number of clusters. Recall that each of the matrices $\{U_i\}_{i=1}^M$ defines a k-dimensional subspace in $\mathbb{R}^n$, which can be denoted as $span(U_i)$. The goal is to merge these multiple subspaces in a meaningful and efficient way. To this end, our philosophy is to find a representative subspace $span(U)$ that is close to all the individual subspaces $span(U_i)$, and at the same time the representation U preserves the information about vertex connectivity in each graph layer. For notational convenience, in the rest of the chapter we simply refer to the representations U and U_i as the corresponding subspaces, unless indicated specifically.

The squared projection distance between subspaces defined in Eq. (3.9) can be naturally generalized for analysis of multiple subspaces. More specifically, we can define the squared projection distance between the target representative subspace U and the M individual subspaces $\{U_i\}_{i=1}^M$ as

the sum of squared projection distances between U and each individual subspace given by U_i:

$$\begin{aligned} d^2_{\text{proj}}(U, \{U_i\}_{i=1}^M) &= \sum_{i=1}^M d^2_{\text{proj}}(U, U_i) \\ &= \sum_{i=1}^M [k - tr(UU^T U_i {U_i}^T)] \\ &= kM - \sum_{i=1}^M tr(UU^T U_i {U_i}^T). \end{aligned} \tag{3.10}$$

The minimization of the distance measure in Eq. (3.10) enforces the representative subspace U to be close to all the individual subspaces $\{U_i\}_{i=1}^M$ in terms of the projection distance on the Grassmann manifold. At the same time, we want U to preserve the information about vertex connectivity in each graph layer. This can be achieved by minimizing the Laplacian quadratic form evaluated on the columns of U, as also indicated by the objective function in Eq. (3.6) for spectral clustering. Therefore, we finally propose to merge multiple subspaces by solving the following optimization problem that integrates Eq. (3.6) and Eq. (3.10):

$$\begin{aligned} \min_{U \in \mathbb{R}^{n \times k}} \sum_{i=1}^M tr(U^T L_{\text{sym}_i} U) + \alpha [kM - \sum_{i=1}^M tr(UU^T U_i {U_i}^T)], \\ \text{s.t.} \quad U^T U = I, \end{aligned} \tag{3.11}$$

where L_{sym_i} and U_i are the graph Laplacian and the subspace representation for G_i, respectively. The regularization parameter α balances the trade-off between the two terms in the objective function.

The problem of Eq. (3.11) can be solved in a similar manner as Eq. (3.6). Specifically, by ignoring constant terms and rearranging the trace form in the second term of the objective function, Eq. (3.11) can be rewritten as

$$\min_{U \in \mathbb{R}^{n \times k}} tr[U^T (\sum_{i=1}^M L_{\text{sym}_i} - \alpha \sum_{i=1}^M U_i {U_i}^T) U], \quad \text{s.t.} \quad U^T U = I. \tag{3.12}$$

It is interesting to note that this is the same trace minimization problem as in Eq. (3.6), but with a "modified Laplacian"[6]:

$$L_{\text{mod}} = \sum_{i=1}^M L_{\text{sym}_i} - \alpha \sum_{i=1}^M U_i {U_i}^T. \tag{3.13}$$

[6] A "modified Laplacian" is adopted here as an intuitive terminology. Strictly speaking, L_{mod} does not satisfy the conditions of being a valid Laplacian matrix. In fact, Eq. (3.13) can be rewritten as $L_{\text{mod}} = \sum_{i=1}^M L_{\text{sym}_i} - \alpha \sum_{i=1}^M U_i {U_i}^T = \sum_{i=1}^M U_i (\Lambda_i - \alpha I) {U_i}^T$, where Λ_i is a diagonal matrix that contains the eigenvalues of L_{sym_i} along the diagonal. It is interesting to notice that $\Lambda_i - \alpha I$ can be considered as a filtering function applied on the eigenvalues; therefore, it is possible to choose other filtering functions so that the resulting "modified Laplacian" is a valid Laplacian matrix, or, to learn such functions in certain applications (see for example the work in [3] on learning diffusion kernels in shape analysis). We leave these directions for future work.

Therefore, by the Rayleigh-Ritz theorem, the solution to the problem of Eq. (3.12) is given by the first k eigenvectors of the modified Laplacian L_{mod}, which can be computed using efficient algorithms for eigenvalue problems [124, 75].

In the problem of Eq. (3.11) we try to find a representative subspace U from the multiple subspaces $\{U_i\}_{i=1}^{M}$. Such a representation not only preserves the structural information contained in the individual graph layers, which is encouraged by the first term of the objective function in Eq. (3.11), but also keeps a minimum distance between itself and the multiple subspaces, which is enforced by the second term. Notice that the minimization of only the first term itself corresponds to simple averaging of the information from different graph layers, which usually leads to suboptimal clustering performance as we shall see in the experimental section. Similarly, imposing only a small projection distance to the individual subspaces $\{U_i\}_{i=1}^{M}$ does not necessarily guarantee that U is a good solution for merging the subspaces. In fact, for a given k-dimensional subspace, there are infinitely many choices for the matrix representation, and not all of them are considered as meaningful summarizations of the information provided by the multiple graph layers. However, under the additional constraint of minimizing the trace of the quadratic term $U^T L_{\text{sym}_i} U$ over all the graphs (which is the first term of the objective function in Eq. (3.11)), the vertex connectivity in the individual graphs tends to be preserved in U. In this case, the smaller the projection distance between U and the individual subspaces, the more representative it is for all graph layers.

Finally, we note that the proposed merging framework can be easily extended to take into account the relative importance of each individual graph layer with respect to the specific learning purpose. For instance, when prior knowledge about the importance of the information in the individual graphs is available, we can adapt the value of the regularization parameter α in Eq. (3.11) to the different layers such that the representative subspace is closer to the most informative subspace representations. Also, we do not incorporate specific prior knowledge about vertex pairs that must, or must not be linked in the design of the merging framework. That could be done by introducing additional graph layers that only consist of such connections, which would certainly be emphasized by the spectral (subspace) representations computed from these graph layers. We can then choose rather large regularization parameters for these layers in the optimization problem to enforce such constraints.

3.3.3 Discussion of the distance function

Interestingly, the choice of projection distance as a similarity measure between subspaces in the optimization problem of Eq. (3.11) can be well justified from information-theoretic and statistical learning points of view. The first justification is from the work of Hamm et al. [60], in which the authors have shown that the Kullback-Leibler (K-L) divergence [70], which is a well-known similarity measure between two probability distributions in information theory, is closely related to the squared projection distance. More specifically, the work in [60] suggests that, under certain conditions, we can consider a linear subspace U_i as the "flattened" limit of a Factor Analyzer distribution p_i [51]:

$$p_i : \mathcal{N}(u_i, C_i), \quad C_i = U_i {U_i}^T + \sigma^2 I_n, \tag{3.14}$$

where $\mathcal{N}$ stands for the normal distribution, $u_i \in \mathbb{R}^n$ is the mean, $U_i \in \mathbb{R}^{n \times k}$ is a full-rank matrix with $n > k > 0$ (which represents the subspace), σ is the ambient noise level, and I_n is the identity

matrix of dimension n. For two subspaces U_i and U_j, the symmetrized K-L divergence between the two corresponding distributions p_i and p_j can then be rewritten as:

$$d_{\mathrm{KL}}(p_i, p_j) = \frac{1}{2\sigma^2(\sigma^2+1)}(2k - 2tr(U_i{U_i}^T U_j{U_j}^T)), \tag{3.15}$$

which is of the same form as the squared projection distance when we ignore the constant factor (see Eq. (3.8)). This shows that, if we take a probabilistic view of the subspace representations $\{U_i\}_{i=1}^M$, then the projection distance between subspaces can be considered consistent with the K-L divergence.

The second justification is from the recently proposed Hilbert-Schmidt Independence Criterion (HSIC) [57], which measures the statistical dependence between two random variables. Given $K_{\mathcal{X}_1}, K_{\mathcal{X}_2} \in \mathbb{R}^{n\times n}$ that are the centered Gram matrices of some kernel functions defined over two random variables $\mathcal{X}_1$ and $\mathcal{X}_2$, the empirical estimate of HSIC is given by

$$d_{\mathrm{HSIC}}(\mathcal{X}_1, \mathcal{X}_2) = tr(K_{\mathcal{X}_1} K_{\mathcal{X}_2}). \tag{3.16}$$

That is, the larger the $d_{\mathrm{HSIC}}(\mathcal{X}_1, \mathcal{X}_2)$, the stronger the statistical dependence between $\mathcal{X}_1$ and $\mathcal{X}_2$. In our case, using the idea of spectral embedding, we can consider the rows of the individual subspace representations U_i and U_j as two particular sets of sample points in $\mathbb{R}^k$, which are drawn from two probability distributions governed by the information on vertex connectivity in G_i and G_j, respectively. In other words, the sets of rows of U_i and U_j can be seen as realizations of two random variables $\mathcal{X}_i$ and $\mathcal{X}_j$. Therefore, we can define the Gram matrices of linear kernels on $\mathcal{X}_i$ and $\mathcal{X}_j$ as:

$$K_{\mathcal{X}_i} = ({U_i}^T)^T({U_i}^T) = U_i{U_i}^T, K_{\mathcal{X}_j} = ({U_j}^T)^T({U_j}^T) = U_j{U_j}^T. \tag{3.17}$$

Combining Eq. (3.16) and Eq. (3.17), we can see that:

$$d_{\mathrm{HSIC}}(\mathcal{X}_i, \mathcal{X}_j) = tr(U_i{U_i}^T U_j{U_j}^T) = k - d^2_{\mathrm{proj}}(U_i, U_j). \tag{3.18}$$

This shows that the projection distance between subspaces U_i and U_j can be interpreted as the negative dependence between $\mathcal{X}_i$ and $\mathcal{X}_j$, which reflect the information provided by the two individual graph layers G_i and G_j.

Therefore, from both information-theoretic and statistical learning points of view, the smaller the projection distance between two subspace representations U_i and U_j, the more similar the information in the respective graphs that they represent. As a result, the representative subspace (the solution U to the problem of Eq. (3.11)) can be considered as a subspace representation that "summarizes" the information from the individual graph layers, and at the same time captures the intrinsic relationships between the vertices in the graph. As one can imagine, such relationships are of crucial importance in our multi-layer graph analysis.

3.4 Clustering on multi-layer graphs

In Section 3.3, we introduced a novel framework for merging subspace representations from the individual layers of a multi-layer graph, which leads to a representative subspace that captures the intrinsic relationships between the vertices of the graph. This representative subspace provides a low dimensional form that can be used in several applications involving multi-layer graph analysis. In particular, we study now one such application, namely the problem of clustering vertices in a multi-layer graph[7]. We further analyze the behavior of the proposed clustering algorithm with respect to the properties of the individual graph layers (subspaces).

3.4.1 Clustering algorithm

As we have already seen in Section 3.2, the success of the spectral clustering algorithm relies on the transformation of the information contained in the graph structure into a spectral embedding computed from the graph Laplacian matrix, where each row of the embedding matrix (after normalization) is treated as the coordinates of the corresponding vertex in a low dimensional subspace. In our problem of clustering on a multi-layer graph, the setting is slightly different, since we aim at finding a unified clustering of the vertices that takes into account information contained in all the individual layers of the multi-layer graph. However, the merging framework proposed in the previous section can naturally be applied in this context. In fact, it leads to a natural solution to the clustering problem on multi-layer graphs. In more details, similarly to the spectral embedding matrix in the spectral cluttering algorithm, which is a subspace representation for one individual graph, our merging framework provides a representative subspace that contains the information from the multiple graph layers. Using this representation, we can then follow the same steps of spectral clustering to achieve the final clustering of the vertices with a k-means algorithm. The proposed clustering algorithm is summarized in Algorithm 2.

It is clear that Algorithm 2 is a direct generalization of Algorithm 1 in the case of multi-layer graphs. The main ingredient of our clustering algorithm is the merging framework proposed in Section 3.3, in which information from individual graph layers is summarized, prior to the actual clustering process (i.e., the k-means step) is implemented. This provides an example that illustrates how our generic merging framework can be applied to specific learning tasks on multi-layer graphs.

3.4.2 Analysis of the proposed algorithm

We now analyze the behavior of the proposed clustering algorithm under different conditions. Specifically, we first outline the link between subspace distance and clustering quality, and then compare the clustering performances in two scenarios where the relationships between the individual subspaces $\{U_i\}_{i=1}^M$ are different. Finally, we discuss about the relationships between the choice of the number of clusters and the clustering performance.

As we have seen in Section 3.3, the rows of the subspace representations $\{U_i\}_{i=1}^M$ can be viewed as realizations of random variables $\{\mathcal{X}_i\}_{i=1}^M$ governed by the graph information. At the same time,

[7] In addition to clustering, which is in an unsupervised fashion, the proposed framework can also be applied in a semi-supervised fashion, to problems such as classification or manifold alignment (see for example the work in [61]). It can also be useful in ranking where the intrinsic relationships between entities, which are summarized from the individual graph layers, would certainly help.

Algorithm 2 Spectral Clustering on Multi-Layer Graphs (**SC-ML**)

Input:
$\{W_i\}_{i=1}^M$: $n \times n$ weighted adjacency matrices of individual graph layers $\{G_i\}_{i=1}^M$
k: target number of clusters
α: regularization parameter
Compute the normalized Laplacian matrix L_{sym_i} and the subspace representation U_i for each G_i.
Compute the modified Laplacian matrix $L_{\text{mod}} = \sum_{i=1}^M L_{\text{sym}_i} - \alpha \sum_{i=1}^M U_i U_i^T$.
Compute $U \in \mathbb{R}^{n \times k}$ that is the matrix containing the first k eigenvectors $u_1, \ldots, u_k$ of L_{mod} as columns. Normalize each row of U to get U_{norm}.
Let $y_j \in \mathbb{R}^k$ $(j = 1, \ldots, n)$ be the transpose of the j-th row of U_{norm}.
Cluster y_j in $\mathbb{R}^k$ into $C_1, \ldots, C_k$ using the k-means algorithm.
Output:
$C_1, \ldots, C_k$: The cluster assignment

spectral clustering directly utilizes U_i for the purpose of clustering. Therefore, $\{\mathcal{X}_i\}_{i=1}^M$ can be considered as random variables that control the cluster assignment of the vertices[8]. Since the projection distance can be understood as the negative statistical dependence between such random variables, the minimization of the projection distance in Eq. (3.11) is equivalent to the maximization of the dependence between the random variable from the representative subspace U and the ones from the individual subspaces $\{U_i\}_{i=1}^M$. The optimization in Eq. (3.11) can then be seen as a solution that tends to produce a clustering with the representative subspace that is consistent with those computed from the individual subspace representations.

We now discuss how the relationships between the individual subspaces possibly affect the performance of our clustering algorithm **SC-ML**. Intuitively, since the second term of the objective function in Eq. (3.11) represents the distance between the representative subspace U and all the individual subspaces $\{U_i\}_{i=1}^M$, it tends to drive the solution towards those subspaces that themselves are close to each other on the Grassmann manifold. To show it more clearly, let us consider two toy examples. The first example is illustrated in Fig. 3.4, where we have a 3-layer graph with the individual layers G_1, G_2 and G_3 sharing the same set of vertices. For the sake of simplicity, all the edge weights are set to one. In addition, three groundtruth clusters are indicated by the colors of the vertices. Table 3.1(a) shows the performances of Algorithm 1 with individual layers as well as Algorithm 2 for the multi-layer graph[9], in terms of the *Normalized Mutual Information (NMI)* [88] with respect to the groundtruth clusters. Table 3.1(b) shows the projection distances between various pairs of subspaces. It is clear that the layers G_1 and G_2 produce better clustering

[8]The columns of U are a rotation of the columns of a cluster indicator matrix by $D^{\frac{1}{2}}$. It has been discussed in [134] that if there exist vertices of particularly low degrees, this rotation would make the columns of U differ from the indicator vectors. However, according to [134], one can argue that such low-degree vertices can be considered as outliers anyway, which does not affect much the global clustering quality. Therefore, the columns of U are quite informative about the global clustering structure.

[9]We choose the value of the regularization parameter α that leads to the best possible clustering performance. For the results presented in both Table 3.1 and Table 3.2, the regularization parameter is set to be 0.5. More discussions about the choices of this parameter are presented in Section 3.5.

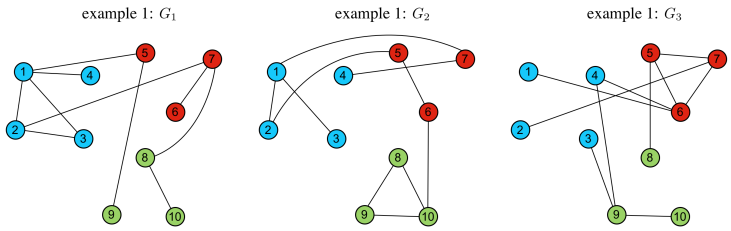


Figure 3.4: A 3-layer graph with unit edge weights for toy example 1. The colors indicate the groundtruth clusters.

Table 3.1: Analysis of toy example 1.

(a) clustering performances for toy example 1

	Layer G_1	Layer G_2	Layer G_3	SC-ML
NMI	0.6279	0.6181	0.2673	1.0000

(b) subspace distances for toy example 1

	Layer G_1	Layer G_2	Layer G_3	Subspace (SC-ML)
Layer G_1	0	1.1100	1.3670	0.9456
Layer G_2	1.1100	0	1.3354	1.0452
Layer G_3	1.3670	1.3354	0	1.0788

quality, and that the distance between the corresponding subspaces is smaller. However, the vertex connectivity in layer G_3 is not very consistent with the groundtruth clusters and the corresponding subspace is further away from the ones from G_1 and G_2. In this case, the solution found by **SC-ML** is enforced to be close to the consistent subspaces from G_1 and G_2, hence provides satisfactory clustering results ($NMI = 1$ represents perfect recovery of groundtruth clusters). Let us now consider a second toy example, as illustrated in Fig. 3.5. In this example we have two layers G_2 and G_3 with relatively low quality information with respect to the groundtruth clustering of the vertices. As we see in Table 3.2(b), their corresponding subspaces are close to each other on the Grassmann manifold. The most informative layer G_1, however, represents a subspace that is quite far away from the ones from G_2 and G_3. At the same time, we see in Table 3.2(a) that the clustering results are better for the first layer than for the other two less informative layers. If the quality of the information in the different layers is not considered in computing the representative subspace, **SC-ML** enforces the solution to be closer to two layers of relatively lower quality, which results in unsatisfactory clustering performance in this case.

The analysis above implies that the proposed clustering algorithm works well under the following assumptions: (i) the majority of the individual subspaces are relatively informative, namely, they are helpful for recovering the groundtruth clustering, and (ii) they are reasonably close to each other on the Grassmann manifold, namely, they are expected to provide *complementary* but not

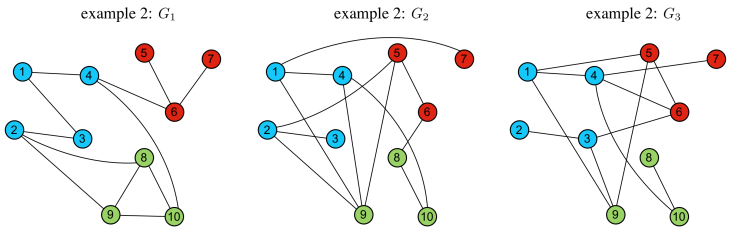


Figure 3.5: A 3-layer graph with unit edge weights for toy example 2. The colors indicate the groundtruth clusters.

Table 3.2: Analysis of toy example 2.

(a) clustering performances for toy example 2

	Layer G_1	Layer G_2	Layer G_3	SC-ML
NMI	0.7934	0.2673	0.4728	0.5300

(b) subspace distances for toy example 2

	Layer G_1	Layer G_2	Layer G_3	Subspace (SC-ML)
Layer G_1	0	1.3098	1.2296	1.0311
Layer G_2	1.3098	0	0.9343	0.8828
Layer G_3	1.2296	0.9343	0	0.5058

contradictory information. When this is the case, the majority of informative views tend to agree with each other, and the information contained in these informative views is likely to be mainly captured by k-dimensional subspaces. The global clustering structure will then mainly be defined by the informative views that admit a value of k for clustering, even though k might not be optimal for each individual view independently (in fact we do not assume that k is optimal for each individual view. Therefore, without loss of generality, it is reasonable for the proposed framework to consider a universal k for the subspace dimension across different views).

In Algorithm 2 we assume that the target number of clusters k is known a priori. Although this is a reasonable assumption done in many popular clustering algorithms, there are practical situations where k is not defined a priori. Traditionally, in spectral methods we could use the eigen-gap of the graph Laplacian matrices as a heuristic to choose the number of clusters [134]. In the case of Algorithm 2, if the majority of the informative views agree with each other, one could estimate the number of clusters k such that the gap between the k-th and the $(k+1)$-th eigenvalues is reasonably large for all these views. We remark, however, that after the merging of multiple layers a particular value of k could emerge as a good choice, which is not necessarily optimal for all the individual views, as in the first toy example illustrated in Fig. 3.6.

Finally, we note that we could use the information about the disagreement between views to tune the regularization parameters in the optimization problem to promote better final clustering

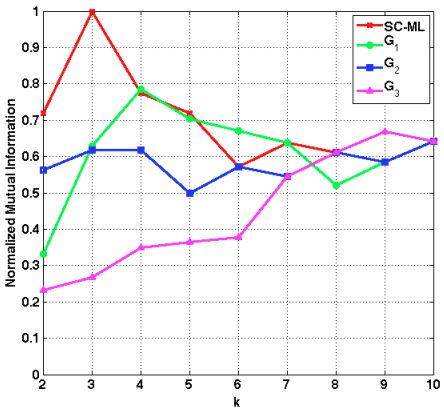


Figure 3.6: Clustering performances of Algorithm 1 on the individual graph layers and Algorithm 2 on the multi-layer graph in the first toy example, under different values of k (number of clusters).

quality. For example, if one view is significantly different or contradictory from other views, we tend to discard it or choose a rather small regularization parameter to attenuate its influence on the final clustering quality.

3.5 Experimental results

In this section, we evaluate the performance of the **SC-ML** algorithm presented in Section 3.4 on one synthetic and two real world datasets. We first describe the datasets that we use for the evaluation, and then explain the various clustering algorithms that we adopt in the performance comparisons. We finally present the results in terms of three evaluation criteria as well as some discussions.

3.5.1 Datasets

The first dataset that we use is a synthetic dataset, where we have three point clouds in $\mathbb{R}^2$ forming the English letters "N", "R" and "C" (shown in Fig. 3.7). Each point cloud is generated from a five-component Gaussian mixture model with different values for the mean and covariance of the Gaussian distributions[10], where each component represents a class of 500 points with specific color. A 5-nearest neighbor graph is then constructed for each point cloud by assigning the weight of the edges connecting two vertices (points) as the reciprocal of the Euclidean distance between them. This gives us a 3-layer graph of 2500 vertices, where each graph layer is from a point cloud forming

[10]For letter "N", the mean and covariance of the five components are [-1 0], [1 3], [3 2], [6 3], [4 0], and [1 0.3; 0.3 1], [0.6 0.1; 0.1 0.5], [0.5 -0.1; -0.1 1.5], [0.8 0.3; 0.3 0.4], [0.5 0.2; 0.2 1.5], respectively. For letter "R", the mean and covariance of the five components are [1 0], [0 2], [2 4], [4 3], [4 0], and [0.8 -0.2; -0.2 0.8], [0.3 0.1; 0.1 0.7], [1 0.3; 0.3 0.2], [0.5 -0.1; -0.1 1.5], [1.2 -0.4; -0.4 0.6], respectively. For letter "C", the mean and covariance of the five components are [1 0], [0 1], [2 3], [4 3], [5 -1], and [1.2 -0.6; -0.6 0.8], [0.6 0.1; 0.1 0.5], [1.2 0.3; 0.3 0.2], [1.5 -0.3; -0.3 0.5], [1.6 0.3; 0.3 0.2], respectively.

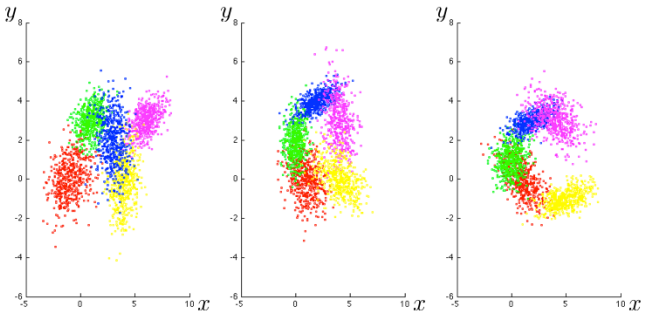


Figure 3.7: Three five-class point clouds in $\mathbb{R}^2$ forming English letters "N", "R" and "C".

a particular letter. The goal with this dataset is to recover the five clusters (indicated by five colors) of the 2500 vertices using the three graph layers constructed from the three point clouds.

The second dataset contains data collected during the Lausanne Data Collection Campaign [67] by the Nokia Research Center (NRC) in Lausanne. This dataset contains the mobile phone data of 136 users living and working in the Lake Léman region in Switzerland, recorded over a one-year period. Considering the users as vertices in the graph, we construct three graphs by measuring the proximities between these users in terms of GPS locations, Bluetooth scanning activities and phone communication. More specifically, for GPS locations and Bluetooth scans, we measure how many times two users are sufficiently close geographically (within a distance of roughly 1 km), and how many times two users' devices have detected the same Bluetooth devices, respectively, within 30-minute time windows. Aggregating these results for a one-year period leads to two weighted adjacency matrices that represent the physical proximities of the users measured with different modalities. In addition, an adjacency matrix for phone communication is generated by assigning edge weights depending on the number of calls between any pair of two users. These three adjacency matrices form a 3-layer graph of 136 vertices, where the goal is to recover the eight groundtruth clusters that have been constructed from the users' email affiliations.

The third dataset is a subset of the Cora bibliographic dataset[11]. This dataset contains 292 research papers from three different fields, namely, natural language processing, data mining and robotics. Considering papers as vertices in the graph, we construct the first two graphs by measuring the similarities among the title and the abstract of these papers. More clearly, for both title and abstract, we represent each paper by a vector of non-trivial words using the *Term Frequency-Inverse Document Frequency (TF-IDF)* [88] weighting scheme, and compute the cosine similarities between every pair of vectors as the edge weights in the graphs. Moreover, we add a third graph which reflects the citation relationships among the papers, namely, we assign an edge with unit weight between papers A and B if A has cited or been cited by B. This results in a 3-layer graph of 292 vertices, and the goal in this dataset is to recover the three clusters corresponding to the different fields the papers belong to.

The adjacency matrices of the graphs are visualized as the spy plots shown in Fig. 3.8(a), (b)

[11] Available online at "`http://people.cs.umass.edu/~mccallum/data.html`" under category "Cora Research Paper Classification".

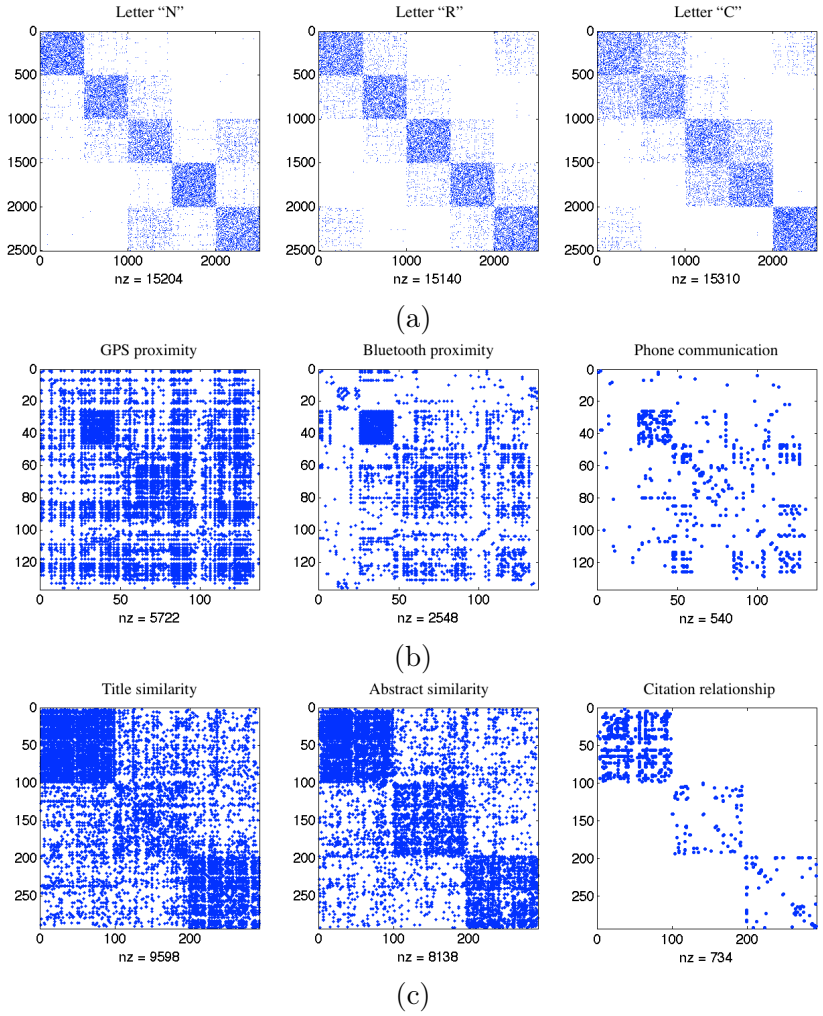


Figure 3.8: Spy plots of three adjacency matrices in (a) the synthetic dataset, (b) the NRC dataset, and (c) the Cora dataset.

and (c) for the synthetic, NRC and Cora dataset, respectively, where the orderings of the vertices are made consistent with the groundtruth clusters[12]. A spy plot is a global view of a matrix where every non-zero entry in the matrix is represented by a blue dot. As shown in these figures, we see clearly the clusters in the synthetic and Cora datasets, while the clusters in the NRC dataset are not very clear. The reason for this is that, in the NRC dataset, the email affiliations used to create the groundtruth clusters only provide approximative information.

[12]The adjacency matrix for GPS proximity in the NRC dataset is thresholded for better illustration.

3.5.2 Clustering algorithms

We now explain briefly the clustering algorithms in our comparative performance analysis along with some implementation details. We adopt three baseline algorithms as well as a state-of-the-art technique, namely the co-regularization approach introduced in [73]. As we shall see, there are interesting connections between the competitor clustering schemes and the proposed algorithm. First of all, we describe some implementation details of the proposed **SC-ML** algorithm and the co-regularization approach in [73]:

- **SC-ML**: Spectral Clustering on Multi-Layer graphs, as presented in Section 3.4. The implementation of **SC-ML** is pretty straightforward, and the only parameter to choose is the regularization parameter α in Eq. (3.11). In our experiments, we choose the value of α through multiple empirical trials and report the peak and average performances of 20 test runs[13]. We will discuss the choice of this parameter later in this section.

- **SC-CoR**: Spectral Clustering with Co-Regularization proposed in [73]. We follow the same practice as in [73] to choose the most informative graph layer to initialize the alternating optimization scheme in **SC-CoR**. The stopping criteria for the optimization process is chosen such that the optimization stops when changes in the objective function are smaller than 10^{-5}. Similarly, we choose the value of the regularization parameter α in **SC-CoR** through multiple empirical trials. As in [73], the parameter α is fixed in the optimization steps for all graph layers.

Next, we introduce three baseline comparative algorithms that work as follows:

- **SC-Single**: Spectral Clustering (Algorithm 1) applied on a single graph layer, where the graph is chosen to be the one that leads to the best clustering results.

- **SC-Sum**: Spectral clustering applied on a global matrix W that is the summation of the normalized adjacency matrices of the individual layers:

$$W = \sum_{i=1}^{M} D_i^{-\frac{1}{2}} W_i D_i^{-\frac{1}{2}}. \tag{3.19}$$

- **SC-KSum**: Spectral clustering applied on the summation K of the spectral kernels [128] of the adjacency matrices:

$$K = \sum_{i=1}^{M} K_i \quad \text{with} \quad K_i = \sum_{m=1}^{d} u_{im} {u_{im}}^T, \tag{3.20}$$

  where n is the number of vertices, $d \ll n$ is the number of eigenvectors used in the definition of the spectral kernels K_i, and u_{im} represents the m-th eigenvector of the Laplacian L_{sym_i} for graph G_i. To make it more comparable with spectral clustering, we choose d to be the target number of clusters in our experiments.

[13]The values of α that achieve the peak performances are 0.695, 0.42 and 0.44 for the synthetic, NRC and Cora datasets, respectively.

3.5.3 Clustering results

We evaluate the performance of the different clustering algorithms with three different criteria, namely *Purity*, *Normalized Mutual Information (NMI)* and *Rand Index (RI)* [88]. Specifically, let $\Omega = \{\omega_1, \ldots, \omega_k\}$ be the computed clusters and $C = \{c_1, \ldots, c_k\}$ be the intended groundtruth classes. First, *Purity* is defined as:

$$Purity(\Omega, C) = \frac{1}{n} \sum_{k} \max_{j} |\omega_k \cap c_j|, \tag{3.21}$$

where n is the total number of entities, and $|\omega_k \cap c_j|$ denotes the number of entities in the intersection of ω_k and c_j. Next, *Normalized Mutual Information* is defined as:

$$NMI(\Omega, C) = \frac{I(\Omega; C)}{[H(\Omega) + H(C)]/2}, \tag{3.22}$$

where I is the mutual information between clusters Ω and classes C , and $H(\Omega)$ and $H(C)$ represent the entropies of the clusters and classes, respectively. Finally, when interpreting clustering as a series of binary decisions on each pair of entities, *Rand Index* is defined as:

$$RI(\Omega, C) = \frac{TP + TN}{TP + FP + FN + TN}, \tag{3.23}$$

where TP, TN, FP, FN represent true positive, true negative, false positive and false negative decisions, respectively.

The clustering results are summarized in Table 3.3 and Table 3.4 for the peak and average performances of all the algorithms out of 20 test runs, respectively. For each scenario, the best result is highlighted in bold font. First, as expected, we see that the clustering performances for the synthetic and Cora datasets are higher than that for the NRC dataset, which indicates that the latter one is indeed more challenging due to the approximative groundtruth information. Second, it is clear that **SC-ML** and **SC-CoR** generally outperform the baseline approaches for the three datasets. More specifically, although both **SC-Sum** and **SC-KSum** indeed improve the clustering quality compared to clustering with individual graph layers, they only provide limited improvement, and the potential drawback for both of the summation methods is that they can be considered as similar to building a simple average graph for representing the different layers of information. Therefore, depending on data characteristics in specific datasets, this might smooth out the particular information provided by individual layers, and thus penalize the clustering performance. In comparison, **SC-ML** and **SC-CoR** always achieve significant improvements in the clustering quality compared to clustering using individual graph layers. Third, **SC-ML** achieves very competitive performance compared to **SC-CoR** for all the three evaluation criteria on the three datasets, with a much simpler implementation scheme and lower computational complexity, which we will explain in more details in the following section. Moreover, the standard deviations shown in Table 3.4 help demonstrate the statistical significance of the clustering results obtained by **SC-ML**, especially in the case of the NRC and Cora datasets where the graphs are of relatively smaller sizes.

In addition to the clustering results provided above, we have computed and shown in Table 3.5

Table 3.3: Peak performances of different clustering algorithms out of 20 test runs on one synthetic and two real world datasets.

Algorithm	Purity (%)			NMI (%)			RI (%)		
	Synthetic	NRC	Cora	Synthetic	NRC	Cora	Synthetic	NRC	Cora
SC-Single	85.8	51.5	95.6	72.7	31.3	83.1	90.2	73.3	94.3
SC-Sum	97.3	54.4	96.9	91.8	36.1	86.6	97.9	77.1	96.0
SC-KSum	97.7	53.7	95.2	92.8	34.1	81.2	98.2	76.7	93.8
SC-CoR	97.8	58.1	**98.3**	92.8	40.6	**93.0**	98.3	78.8	**97.8**
SC-ML	**98.3**	**61.0**	**98.3**	**94.1**	**41.6**	91.8	**98.6**	**79.3**	**97.8**

Table 3.4: Average performances of different clustering algorithms out of 20 test runs on one synthetic and two real world datasets. The numbers in parenthesis are the standard deviations.

Algorithm	Purity (%)			NMI (%)			RI (%)		
	Synthetic	NRC	Cora	Synthetic	NRC	Cora	Synthetic	NRC	Cora
SC-Single	84.2(5.0)	50.3(2.6)	95.6(0.0)	71.3(4.2)	32.7(2.1)	83.1(0.0)	89.4(2.4)	74.4(1.6)	94.3(0.0)
SC-Sum	96.4(4.2)	52.7(2.7)	93.7(9.8)	91.3(2.3)	34.7(1.9)	83.4(9.8)	97.5(1.9)	76.7(1.0)	93.6(7.5)
SC-KSum	95.8(6.0)	53.1(1.9)	95.2(0.0)	91.7(3.4)	33.9(1.6)	81.2(0.0)	97.3(2.7)	76.8(1.0)	93.8(0.0)
SC-CoR	96.7(4.6)	57.8(1.9)	**98.3(0.0)**	91.9(3.1)	**39.7(1.9)**	**93.0(0.0)**	97.8(2.0)	77.9(1.4)	**97.8(0.0)**
SC-ML	**98.2(0.0)**	**58.0(1.5)**	**98.3(0.0)**	**93.8(0.0)**	38.7(1.4)	91.8(0.0)	**98.6(0.0)**	**78.4(0.6)**	**97.8(0.0)**

the confusion matrices based on the outcomes of the five clustering algorithms on the NRC dataset, as an illustrative example of the clustering qualities. The columns of the confusion matrices represent the predicted clusters while the rows represent the intended classes. The diagonal entries represent the numbers of entities that have been correctly identified for each class. By summing up the diagonal entries, it is clear that overall **SC-ML** best reveals the eight classes in the groundtruth data.

Finally, the eigen-gap is considered as a heuristic indicator of the clusterability of the vertices into k subsets. To understand better the benefits of multi-layer graph clustering compared to clustering with individual graph layers, we have computed the gap between the k-th and the $(k+1)$-th eigenvalues of the graph Laplacian matrices corresponding to individual and merged graph layers, where k is the target number of clusters. Specifically, although the proposed merging framework does not lead directly to a graph topology, but rather a representative subspace, we created a "virtual" graph by using a Gaussian kernel together with the Euclidean distances between the low dimensional representations of every pair of vertices in the representative subspace. The same method can be used to create a "virtual" graph in the co-regularization approach. We then compared the eigen-gaps of the individual graph Laplacian matrices and the eigen-gaps computed using the "virtual" graphs, and the results for different datasets are shown in Table 3.6. As we can see, clustering on multi-layer graphs always leads to larger eigen-gaps, which is indicative of better clustering structures.

Table 3.5: Confusion matrices for different clustering algorithms on the NRC dataset.

SC-Single

3	4	0	1	2	0	0	9
1	**3**	0	0	0	0	0	2
1	0	**14**	0	0	0	6	0
0	1	0	**4**	4	0	0	4
0	1	0	1	**17**	3	0	12
0	0	0	0	1	**0**	0	1
0	0	0	0	0	0	**0**	2
1	7	1	5	5	2	3	**15**

diagonal sums up to 56

SC-Sum

8	1	2	0	2	5	0	1
3	**1**	0	0	0	0	1	1
0	0	**21**	0	0	0	0	0
1	1	0	**9**	1	0	0	1
9	0	0	3	**12**	0	1	9
0	1	0	0	1	**0**	0	0
0	0	0	0	1	0	**1**	0
8	6	5	10	4	0	2	**4**

diagonal sums up to 56

SC-KSum

3	3	0	1	4	1	0	7
0	**3**	0	0	0	0	0	3
0	0	**20**	0	1	0	0	0
1	1	0	**9**	1	0	0	1
0	1	0	3	**13**	6	4	7
1	0	0	0	1	**0**	0	0
0	1	0	0	1	0	**0**	0
1	7	4	10	4	3	0	**10**

diagonal sums up to 58

SC-ML

7	4	1	0	1	3	1	2
1	**3**	0	0	0	0	0	2
0	0	**20**	0	0	1	0	0
1	0	0	**10**	1	0	1	0
0	1	0	5	**20**	0	8	0
1	0	0	0	1	**0**	0	0
0	1	0	0	1	0	**0**	0
2	5	5	13	4	0	3	**7**

diagonal sums up to 67

SC-CoR

11	2	2	0	1	2	1	0
3	**1**	0	0	0	1	0	1
0	0	**21**	0	0	0	0	0
1	0	0	**10**	1	0	1	0
0	0	0	3	**12**	1	8	10
1	0	0	0	0	**0**	0	1
0	0	0	0	0	0	**0**	2
6	5	5	10	1	2	3	**7**

diagonal sums up to 62

3.5.4 Further discussions

We now present some discussions on parameter selection in **SC-ML** and its connections to the competitor clustering schemes. First of all, we discuss the influence of the choice of the regularization parameter α on the performance of **SC-ML**. In Fig. 3.9, we compare the performances of **SC-ML** and **SC-CoR** in terms of *NMI* under different values of parameter α in the corresponding implementations. As we can see, in our experiments, **SC-ML** achieves the best performances when α is chosen between 0.4 and 0.6, and it outperforms **SC-CoR** for a large range of α for the synthetic and NRC datasets. For the Cora dataset, the two algorithms achieve very similar performances at different values of α, but **SC-ML** permits a larger range of parameter selection. Furthermore, it is worth noting that the optimal values for α in **SC-ML** lie in similar ranges across different datasets, thanks to the adoption of the normalized graph Laplacian matrix whose spectral norm is upper bounded by 2. In summary, this shows that the performance of **SC-ML** is reasonably stable with respect to the parameter selection.

The role of the parameter α can also be understood by comparing **SC-ML** with **SC-Sum** and **SC-KSum**. Specifically, **SC-Sum** can be considered as taking an average of the graph Laplacian matrices of the individual graph layers, while **SC-KSum** takes the average of the corresponding low-dimensional spectral representations. These are exactly the two parts of the objective function in the optimization problem of Eq. (3.11), whose relative importance is weighted by α. Theoretically,

Table 3.6: Eigen-gap between the k-th and the $(k+1)$-th eigenvalues of the graph Laplacian matrices from the individual graph layers and the "virtual" graphs computed by **SC-CoR** and **SC-ML** on different datasets.

Eigen-gap between the k-th and (k+1)-th eigenvalues	Layer G_1	Layer G_2	Layer G_3	SC-CoR	SC-ML
Synthetic dataset	0.0010	0.0005	0.0008	0.2140	0.2146
NRC dataset	0.0116	0.0084	0.0087	0.0125	0.0250
Cora dataset	0.0384	0.1163	0.0000	0.2792	0.2954

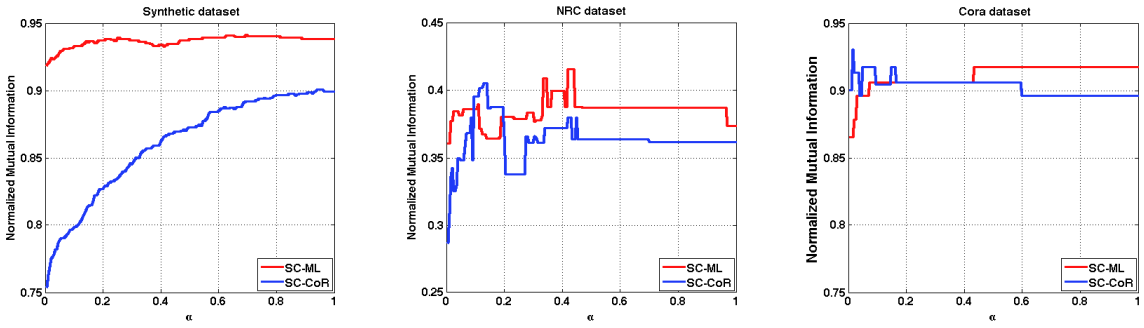


Figure 3.9: Comparison of performances (in terms of *NMI*) of **SC-ML** and **SC-CoR** under different values of parameter α in the corresponding implementations.

on the one hand, if we set α to be zero, the solution of the problem becomes equivalent to the one found by **SC-Sum**; on the other hand, if we let α go to infinity, then the solution becomes equivalent to the one found by **SC-KSum**. This intuition is confirmed by the results shown in Fig. 3.10. As we can see, **SC-ML** achieves the same performances as **SC-Sum** and **SC-KSum** when α is chosen to be 0 and very large, respectively. More importantly, for a wide and stable range of choices of α, it leads to better clustering performance than these two baseline schemes.

Finally, we take a closer look at the comparisons between **SC-ML** and **SC-CoR**. Although the latter is not developed from the viewpoint of subspace analysis on the Grassmann manifold, it can actually be interpreted as a process in which individual subspace representations are updated based on the same distance analysis as in our framework. In this sense, **SC-CoR** uses the same distance as ours to measure similarities between subspaces. The merging solution however leads to a different optimization problem than that of Eq. (3.11), which is based on a slightly different merging philosophy. Specifically, it enforces the information contained in the individual subspace representations to be consistent with each other. An alternating optimization scheme optimizes, at each step, one subspace representation, while fixing the others. This can be interpreted as a process in which one subspace at each step becomes closer to other subspaces in term of the projection distance on the Grassmann manifold. Upon convergence, all initial subspaces are "brought" closer to each other and the final subspace representation from the most informative graph layer is considered

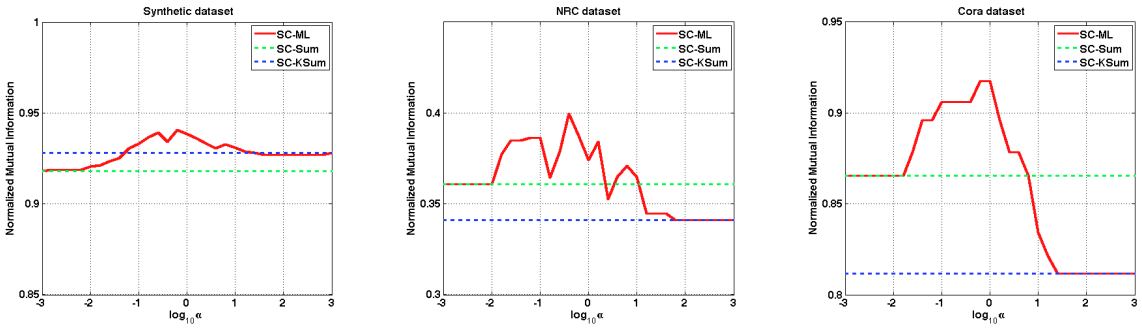


Figure 3.10: Comparison between performances (in terms of *NMI*) of **SC-ML**, **SC-Sum** and **SC-KSum** under different values of parameter α.

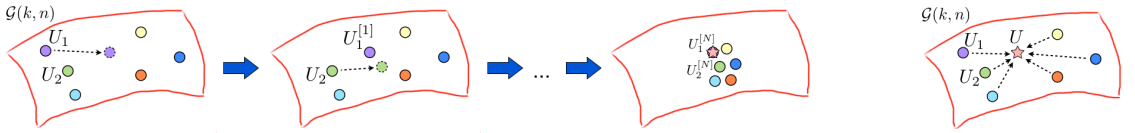


Figure 3.11: Illustrations of graph layer merging. (a) **SC-CoR**: iterative update of the individual subspace representations. The superscript $[N]$ represents the number of iterative steps on each individual subspace representation. The final update of the subspace representation for the most informative graph layer ($U_1^{[N]}$, shown as a star) is considered as a good merging solution; (b) **SC-ML**: the representative subspace (U, shown as a star) is found in one step.

as the one that combines information from all the graph layers efficiently. Two illustrations of **SC-CoR** and **SC-ML** are shown in Fig. 3.11(a) and (b), respectively. Therefore, on the one hand, results for both approaches demonstrate the benefit of using our distance analysis on the Grassmann manifold for merging information in multi-layer graphs. Indeed, for both approaches, since the distances between the solutions and the individual subspaces are minimized without sacrificing too much of the information from individual graph layers, the resulting combinations can be considered as good summarizations of the multiple graph layers. On the other hand, however, **SC-ML** differs from **SC-CoR** mainly in the following aspects. First, the alternating optimization scheme in **SC-CoR** focuses only on optimizing one subspace representation at each step, and it requires a sensible initialization to guarantee that the algorithm ends up at a good local minimum for the optimization problem; it also does not guarantee that all the subspace representations converge to one point on the Grassmann manifold (it uses the final update of the most informative layer for clustering)[14]. In contrast, **SC-ML** directly finds a single representation through a unique optimization of the representative subspace with respect to all graph layers jointly, which does not need alternating optimization steps and careful initializations. These are the possible reasons why **SC-ML** performs

[14]In [73], the authors have also proposed a "centroid-based co-regularization approach" that introduces a consensus representation. However, such a representation is still computed via an alternating optimization scheme, which needs a sensible initialization and keeps the same iterative nature.

slightly better than **SC-CoR** for the synthetic and NRC datasets in our experiments. Second, it is worth noting from a computational point of view that, the performance improvements are achieved with lower computational complexity, since the optimization process involved in **SC-ML** is much simpler than that in **SC-CoR**. Specifically, the iterative nature of **SC-CoR** requires solving an eigenvalue problem for MN times, where M and N are the number of individual graphs and the number of iterations needed for the algorithm to converge, respectively. In contrast, since **SC-ML** aims at finding a globally representative subspace without modifying the individual ones, it needs to solve an eigenvalue problem only once.

3.6 Conclusions

In this chapter, we provided a framework for analyzing information provided by multi-layer graphs and for clustering vertices of graphs in rich datasets. Our generic approach is based on the transformation of information contained in the individual graph layers into subspaces on the Grassmann manifold. The estimation of a representative subspace can then be essentially considered as the problem of finding a good summarization of multiple subspaces using distance analysis on the Grassmann manifold. The proposed framework can be applied to various learning tasks where multiple subspace representations are involved. Under appropriate and realistic assumptions, we show that it leads to a novel clustering algorithm on multi-layer graphs that is competitive to the state-of-the-art techniques.

The analysis presented in this chapter can be extended in the following directions. First, the subspace representation inspired by spectral clustering is certainly not the only valid representation for the graph information. For example, one alternative can be the subspace derived from the modularity matrix of the graph as suggested in [97, 98]. Other possibilities can be subspaces defined via various kernels or dictionaries defined on graphs based on specific low-pass and high-pass filterings of the eigenvalues of the graph Laplacian in the graph spectral domain. The latter choice is more from the viewpoint of signal processing on graphs, which we will discuss in more details in Chapter 5. It is thus an interesting problem to find the most appropriate subspace representation for the data available. Second, instead of finding a k-dimensional representative subspace in one shot by solving an optimization problem (in this chapter the problem of Eq. (3.11)), it might be possible to define such a representative subspace in an iterative fashion, where k 1-dimensional subspaces are defined recursively, each one being orthogonal to the previously chosen subspace(s). Third, although the proposed framework in this chapter can be easily extended to take into account prior knowledge about the respective importance of the individual layers, these layers are still treated in an equal manner in the optimization problem. It would be interesting to see whether one can define a framework where the individual layers play different roles in the merging framework. In the next chapter, we propose an alternative solution that goes along this third direction.

Chapter 4

Learning with multi-layer graphs: Spectral regularization

4.1 Introduction

In the previous chapter, we proposed a framework for merging the information from individual layers of the multi-layer graph, based on the analysis of subspace distances on the Grassmann manifold. This can be considered as a machine learning approach to the problem of learning of multi-layer graphs. As we have seen before in the optimization problem of Eq. (3.11), different layers are treated in an equal manner and are combined in a linear fashion, where the respective importance of the individual layers is controlled by the regularization parameter for each layer in the linear combination. This approach however might be limited in challenging scenarios where individual layers do not have strong agreement with each other and an optimal solution could be difficult to find by a weighted averaging scheme. In such cases, we may need a better way to take into account the respective importance of the layers. This motivates us to propose in this chapter an alternative solution to the problem of merging multiple graph layers, in which individual layers play different roles in the merging framework based on their respective importance.

To enable such a framework, we propose to model the information contained in the different layers in different forms. Our idea is based on a more detailed understanding of the behaviors of the eigenvectors of the graph Laplacian matrix from a signal processing perspective. Specifically, we observe that the first k eigenvectors of the graph Laplacian contain the graph information, and at the same time they can be considered as smooth signals defined on the vertex set of the graph. This inspired us to seek for a set of smooth signals that are jointly smooth on the individual layers, hence capture the information contained in all the layers. In more details, given two graph layers, we propose a spectral regularization framework in which we consider the eigenvectors of the Laplacian matrix from one graph layer as signals defined on another layer. By enforcing the "smoothness" of such signals on the graph through a novel regularization framework, we are able to find a set of "joint eigenvectors" that forms a joint low dimensional embedding of the vertices in the graph, which captures the characteristics of both layers. We call such an embedding a "joint spectrum"[1]

[1]Strictly speaking, the spectrum of the graph is usually defined as the set of eigenvalues of the Laplacian matrix or the adjacency matrix of the graph [29]. In this chapter, we use the intuitive terminology of a "joint spectrum" to

of the two individual layers. We then propose an information-theoretic approach to generalize this framework to the case in which we have more than two graph layers, where the multiple layers are combined in a recursive way.

The concept of a "joint spectrum" is generally helpful in the analysis of data that can be conveniently modeled as a multi-layer graph. For example, it can lead to the generalization of the classical spectral analysis frameworks to multi-dimensional cases. In this chapter, we introduce a novel method based on the proposed spectral regularization framework for clustering vertices in the multi-layer graph. We evaluate the performance of the proposed clustering method on the same datasets used in the previous chapter, and demonstrate its competitive clustering performance. The main difference with the approach proposed in the previous chapter is that, the spectral regularization framework permits to taking into account the respective importance of individual layers in a unique and convincing way. This is potentially helpful in challenging clustering tasks on multi-layer graphs where an optimal clustering solution might be difficult to find by a weighted averaging scheme.

The rest of the chapter is organized as follows. In Section 4.2, we describe in details the proposed spectral regularization approach for merging multiple graph layers. In Section 4.3, we introduce our novel clustering method based on spectral regularization. We show experimental results of our clustering method in Section 4.4 and conclude the chapter in Section 4.5.

4.2 Spectral regularization

In this section, we describe the proposed spectral regularization framework for merging individual layers in the multi-layer graph, where we treat the individual layers based on their respective importance. As a consequence, this method helps preserve specificities of each layer in the merging framework.

4.2.1 Intuition

Our idea is based on a more detailed understanding of the behavior of the eigenvectors of the graph Laplacian matrix. Consider a weighted and connected graph G with n vertices.. From spectral graph theory [29], we know that the eigenvectors $u_1, \ldots, u_n$ of the graph Laplacian matrix L has the following properties:

1. The first eigenvalue λ_1 is 0 and the corresponding eigenvector u_1 is a constant vector;
2. For $i = 2, \ldots, n$, u_i satisfies: $u_i \perp \mathbf{1}$ and $||u_i|| = 1$.

Now consider the problem of mapping the vertices of graph G on a 1-dimensional line such that connected vertices stay as close as possible on the line, with the condition that the mapping vector satisfies the second property above. In other words, we want to find a 1-dimensional mapping,

denote a joint spectral embedding of the individual graph layers.

namely, a scalar signal $f : V \to \mathbb{R}^n$, that minimizes the following term:

$$\arg\min_{f\in\mathbb{R}^n} \Big\{ \sum_{i,j}^{n} w_{i,j}(f(i) - f(j))^2 \Big\}, \quad \text{s.t.} \quad f \perp \mathbf{1},\ ||f|| = 1, \tag{4.1}$$

where $f(i)$ and $f(j)$ represent the mappings of vertices i and j respectively, and $w_{i,j}$ is the weight of the edge between the two vertices. The constraints on the norm of f and the orthogonality to the constant one vector $\mathbf{1}$ are introduced to make the solution nontrivial and unique, and can be explained from a graph-cut point of view [134]. Moreover, since eigenvectors of the Laplacian matrix can be viewed as signals defined on the vertices of the graph, these conditions suggest that they can be considered as candidate solutions to the problem of Eq. (4.1). In fact, we can rewrite Eq. (4.1) in terms of the graph Laplacian matrix L so that an equivalent problem is:

$$\arg\min_{f\in\mathbb{R}^n} f^T L f, \quad \text{s.t.} \quad f \perp \mathbf{1},\ ||f|| = 1. \tag{4.2}$$

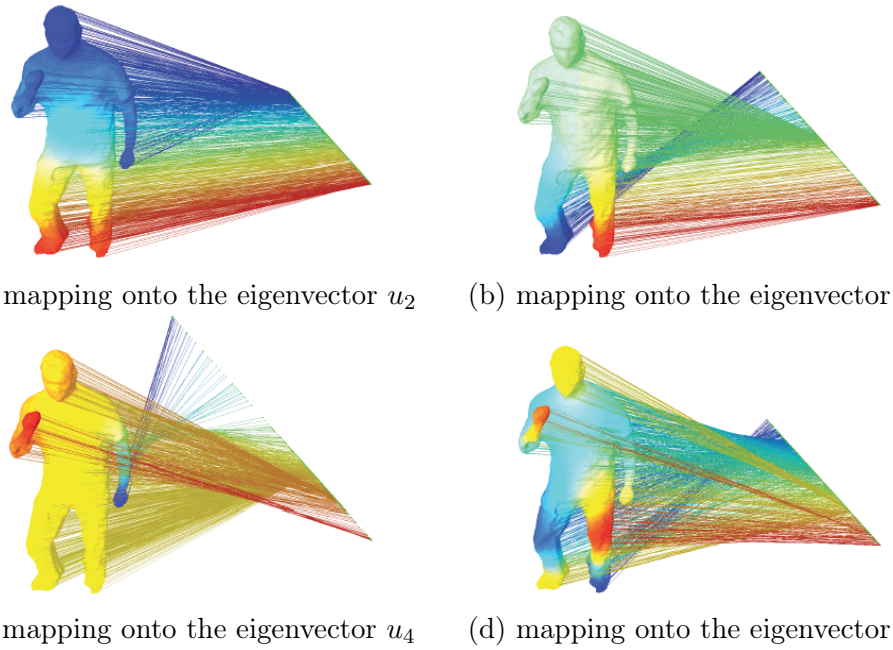
It can be shown by the Rayleigh-Ritz theorem [134] that the solution to the problem of Eq. (4.2) is u_2, the eigenvector that corresponds to the second smallest eigenvalue of L, which is usually called the Fiedler vector of the graph.

As an illustrative example of such a mapping, a weighted graph G constructed from a 3-dimensional point cloud and its mapping onto the Fiedler vector u_2 are shown in Fig. 4.1(a) [63, 68, 113]. It can be seen that this mapping indeed keeps the strongly connected vertices as close as possible on the line. More importantly, it is shown in Zhou et al. [144] that the quadratic objective in Eq. (4.2) can be viewed as a smoothness measure of the signal f defined on the vertices of the graph G, that is, f has similar values on the vertices that are strongly connected in the graph. Therefore, the fact that it minimizes this objective implies that the Fiedler vector u_2 is a smooth signal on the graph. In fact, since we have

$$u_i^T L u_i = \lambda_i, \quad \text{for} \quad i = 1, \ldots, n, \tag{4.3}$$

all the first k eigenvectors tend to be smooth on the graph G provided that the first k eigenvalues are sufficiently small. This is illustrated in Fig. 4.1(b), (c), (d) for u_3, u_4 and u_8, and we can see that closely related points stay quite close on the mappings they represent. Since these first k eigenvectors are used to form the low dimensional embedding U in the spectral clustering algorithm, such smoothness property implies that a set of smooth signals on the graph, such as eigenvectors of the graph Laplacian matrix, can well represent the graph connectivity.

This inspires us to combine information from multiple graph layers, with the help of a set of joint eigenvectors that are smooth on all the layers, hence capture all their characteristics. However, instead of treating all the layers equally, we try to highlight the specificities of different layers. Therefore, we propose the following methodology. Consider two individual graph layers G_1 and G_2. From the smoothness analysis above, we observe that the eigenvectors of the Laplacian matrix from G_1 are smooth signals on G_1; in the meantime, since they can also be viewed as signals defined on G_2, we try to enforce their smoothness on G_2 as well. This leads to a set of eigenvectors that are smooth on both layers, namely a jointly smooth spectrum shared by G_1 and G_2; this spectrum



(a) mapping onto the eigenvector u_2 (b) mapping onto the eigenvector u_3

(c) mapping onto the eigenvector u_4 (d) mapping onto the eigenvector u_8

Figure 4.1: Examples of 1-dimensional mappings of the vertices of a 3-dimensional point cloud graph onto four eigenvectors of the graph Laplacian (figure from [63]).

captures the characteristics of both graph layers.

4.2.2 Jointly smooth spectrum computation

We propose a spectral regularization framework to compute a jointly smooth spectrum of two graph layers G_1 and G_2 by solving the following optimization problem:

$$\arg\min_{f_i \in \mathbb{R}^n} \left\{ ||f_i - u_i||_2^2 + \alpha \cdot \Phi_{f_i} \right\} \quad \text{for} \quad i = 2, \ldots, k, \tag{4.4}$$

where $f_i : V \to \mathbb{R}^n$ is a signal defined on the graph, u_i is the i-th eigenvector from G_1, $\Phi_{f_i} = f_i^T L_{\text{sym}_2} f_i$ is a quadratic term from G_2 which measures the smoothness of f_i on G_2, and the operator $||\cdot||_2$ denote the L^2-norm. We provide two remarks here. First, since the smoothness analysis in Section 4.2.1 can be easily generalized from L to L_{sym}, we follow Zhou et al. [144] to use L_{sym} instead of L to define the quadratic smoothness term Φ_{f_i}. Second, we propose to define u_i as the i-th eigenvector of the random walk graph Laplacian $L_{\text{rw}1}$ from G_1. The reason behind this choice is that the first k eigenvectors of both L and L_{rw} are the cluster indicator vectors, where those of L_{sym} are not [134]. Therefore, although L_{sym} can also be used here, we choose L_{rw} to compute u_i, which is more desirable in our framework (more discussions on this will be presented in Section 4.3). After all, such a choice will not incur an extra eigen-decomposition process, since the eigenvectors of L_{rw} is a rotation of those of L_{sym} by $D^{-\frac{1}{2}}$, where D is the degree matrix of the graph.

In the problem of Eq. (4.4), we seek for a signal f_i such that it is not only close to the eigenvector u_i that comes from G_1, but also sufficiently smooth on G_2 in terms of the quadratic smoothness measure. This promotes the smoothness property of our solution f_i on both of the layers, so that f_i can be considered as a joint eigenvector of G_1 and G_2. The regularization parameter α is used to balance the trade-off between the data fidelity term and the regularization term in the objective function. The problem of Eq. (4.4) can be solved efficiently, since it is shown in [144] that it has a closed form solution:

$$f_i^* = \mu(L_{\text{sym}_2} + \mu I_n)^{-1} u_i, \tag{4.5}$$

where $\mu = \frac{1}{\alpha}$ and I_n represents the identity matrix of dimension n. Furthermore, notice that for each $\{u_i\}_{i=2}^k$ there is an associated optimization problem (except for $i = 1$ since the first eigenvector u_1 of L_{rw} is a constant vector), hence by solving all these problems we get a set of approximate joint eigenvectors[2] f_i for $i = 2, \ldots, k$. Therefore, they together can be viewed as a jointly smooth (or a joint) spectrum of G_1 and G_2. Such a jointly smooth spectrum captures characteristics of both graph layers.

It is worth noting that G_1 and G_2 play different roles in our merging framework. Specifically, G_1 is used for the eigen-decomposition process to get the eigenvectors, and G_2 is used as the graph structure for the regularization process[3]. It is natural to choose the more informative layer as G_1. Therefore, the proposed framework is able to promote specificities of the individual layers in the merging process by treating them differently. Moreover, we can generalize the above framework to multi-layer graphs with more than two layers. Specifically, we propose to start with the most informative graph layer G_1, and search for the next layer G_2 that maximizes the mutual information between G_1 and G_2. More clearly, the mutual information between two graph layers is introduced by interpreting clustering from each individual layer as a discrete distribution of the cluster memberships of the vertices. Therefore, it can be calculated by measuring the mutual information between two distributions using Eq. (3.22). Next, after having the combination of the first two layers, we can repeat the process by maximizing the mutual information between the current combination and the next selected layer, until we include all the layers in the end. This provides a greedy approach to compute a jointly smooth spectrum of all layers in the multi-layer graph.

Finally, compared to the approach proposed in Chapter 3, which is along the lines of machine learning approaches, it is clear that the spectral regularization framework is designed from a signal processing perspective. Specifically, the graph information in different layers are considered as signals defined on graphs, and the combination is done by enforcing some desired properties of

[2]We adopt here the terminology of a set of "approximate joint eigenvectors", since the precise definition for "joint eigenvectors" are usually only for commuting matrices with simple spectrum; we also do not require f_i to be orthogonal to each other in our framework.

[3]As a result, the solution of the problem of Eq. (4.4) would change if we exchange the roles of G_1 and G_2. However, if L_{sym_1} is used for computing u_i, then, with the additional constraint that f_i are orthogonal to each other, one can also interpret the problem of Eq. (4.4) as a kind of joint diagonalization, where the exchange of the roles of G_1 and G_2 would not bring much difference. To see this more clearly, denote $F \in \mathbb{R}^{n\times k}$ and $U_1 \in \mathbb{R}^{n\times k}$ as the matrices that contain f_i and u_i as columns, and suppose that there exists an orthogonal matrix $A \in \mathbb{R}^{k\times k}$ that satisfies $F = U_2 A$, where $U_2 \in \mathbb{R}^{n\times k}$ contains the top k eigenvectors of L_{sym_2}. In this case, since $tr(F^T L_{\text{sym}_2} F) = tr(A^T {U_2}^T L_{\text{sym}_2} U_2 A) = tr(A^T \Lambda_2 A) = tr(\Lambda_2 A A^T)$ which is a constant, the second term of the objective in Eq. (4.4) can be dropped and one would find an orthogonal matrix A that minimizes $||U_2 A - U_1||_F^2$. This can thus be considered as an eigenvector alignment problem (which is a special case of the optimization problem in [69]), where exchanging U_1 and U_2 would not bring much difference.

the signals, in this case the smoothness property. In the next section, we apply the spectral regularization framework to the clustering problem considered in the previous chapter.

4.3 Clustering on multi-layer graphs via spectral regularization

The joint spectrum found by the proposed spectral regularization framework can be applied to various learning tasks on the multi-layer graph. Especially, it leads to a straightforward method for clustering the vertices in the multi-layer graph. More specifically, the joint eigenvectors can be used to define a joint spectral embedding matrix, similarly to that in the spectral clustering algorithms, based on which we perform clustering. The proposed clustering algorithm for two graph layers is summarized in Algorithm 3.

Algorithm 3 Spectral Clustering with Spectral Regularization (**SC-SR**)

1: **Input:**
$\{W_i\}_{i=1}^2$: $n \times n$ weighted adjacency matrices of individual graph layers $\{G_i\}_{i=1}^2$
k: target number of clusters
α: regularization parameter
2: Compute the random walk graph Laplacian $L_{\text{rw1}} = {D_1}^{-1}(D_1 - W_1)$.
3: Compute $U \in \mathbb{R}^{n \times k}$ that is the matrix containing the first k eigenvectors $u_1, \ldots, u_k$ of L_{rw1} as columns.
4: For $i = 2, \ldots, k$, solve the spectral regularization problem of Eq. (4.4) for each u_i and replace it with the solution f_i in U to form the new low dimensional embedding U'.
5: Let $y_j \in \mathbb{R}^k$ $(j = 1, \ldots, n)$ be the transpose of the j-th row of U'.
6: Cluster y_i in $\mathbb{R}^k$ into $C_1, \ldots, C_k$ using the k-means algorithm.
7: **Output:**
$C_1, \ldots, C_k$: The cluster assignment

In practice, given more than two graph layers, we use the procedure described in Section 4.2.2 to combine all the layers in a recursive way. It can be seen that our clustering algorithm is a direct generalization of the spectral clustering algorithm proposed in [116], where, instead of operating on a single graph, it computes a joint low dimensional embedding of all the individual graph layers via the spectral regularization framework. To show that the proposed method can indeed lead to improved clustering performance, in addition to the intuition provided in the previous section, we further explain as follows the reasons why the spectral regularization framework can be considered as a good way of combining the spectrum of individual graph layers for clustering.

We first interpret the combination of multiple layers from the viewpoint of label propagation [146, 151, 16, 147], which is proven to be an effective approach for graph-based semi-supervised learning. In label propagation, one usually has a similarity graph whose vertices represent entities and edges reflect the pairwise relationships between them. We let the initial labels of the vertices propagate towards their neighboring vertices to make inference, based on the strength of relationships between them and their neighbors. This is exactly what the spectral regularization framework in Eq. (4.4) does. More clearly, the optimization problem of Eq. (4.4) can be solved through an

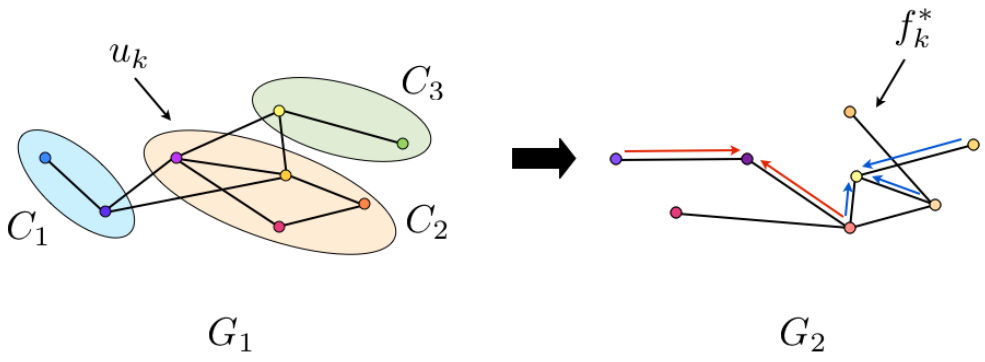


Figure 4.2: Illustration of the label propagation interpretation of the spectral regularization framework: The solution ${f_k}^*$ is found by letting the values of the eigenvector u_k from G_1, which is a cluster indicator vector, propagate linearly on the topology of G_2. In this figure, the values of u_k and ${f_k}^*$ associated with the vertices are color-coded.

iterative process, where in each iteration we have for every vertex $v \in V$:

$$(f_i(v))^{[t+1]} \leftarrow \beta((I_n - L_{\text{sym}_2})f_i^{[t]})(v) + (1-\beta)u_i(v), \tag{4.6}$$

where u_i contains the initial values at the vertices and $f_i^{[t]}$ represents f_i at iteration t [144]. The parameter β is defined as $\beta = \frac{\alpha}{1+\alpha}$ while α is the regularization parameter in Eq. (4.4). In other words, the value at each vertex is updated by a convex combination of the initial value $u_i(v)$ and the values at its neighboring vertices at the current iteration, where the parameter β balances the trade-off between the two. Notice that the initial values $\{u_i\}_{i=1}^k$ from G_1 form the continuous-valued solver of a relaxed discrete graph-cut problem [134]. Therefore, $\{u_i\}_{i=1}^k$ can be viewed as labels indicating the cluster memberships derived from G_1. Consequently, the spectral regularization framework in Eq. (4.4) can be interpreted as a label propagation process, where the cluster labels derived from G_1 are linearly propagated on G_2. In this way, both of the graph structures have been taken into account hence making the resulting combination meaningful. An illustration of this interpretation is shown in Fig. 4.2.

Another interpretation is based on disagreement minimization [33, 72], which has been proposed for the task of learning with multi-view data. The basic idea is to minimize the disagreement between information from the multiple views so that we get a good combination of all the views. For example, Kumar et al. [72] suggests a clustering algorithm that minimizes the disagreement between information from multiple graphs. Similarly, since we aim at finding a unified clustering result from multiple graph layers, it is natural to enforce the consistency between the clustering result and the information from all the graph layers, or in other words, to minimize the disagreement between them. Such a disagreement is again reflected in the objective function of the optimization problem of Eq. (4.4). More specifically, the data fidelity term explicitly measures the disagreement between the solution f_i and the initial vector u_i that comes from G_1, while the regularization term implicitly represents the inconsistency of the information contained in f_i with the structure of G_2.

Indeed, the regularization term Φ_f can be expressed in the following form:

$$\Phi_f = \frac{1}{2}\sum_{i,j}^{n} w_{i,j}\Big(\frac{f(i)}{\sqrt{d(i)}} - \frac{f(j)}{\sqrt{d(j)}}\Big)^2, \tag{4.7}$$

where $d(i)$ and $d(j)$ are the degrees of i and j, respectively. This means that Φ_{f_i} will only be small if the two end-point vertices of a large-weight edge in G_2 have similar signal values normalized by their respective degree. Therefore, minimizing the objective function in Eq. (4.4) can be considered as minimizing the overall disagreement between the solution f_i and the information from the two graph layers. Notice that in this formation the disagreement is modeled from two different viewpoints for the two individual layers, whose respective importance is controlled by the parameter α.

4.4 Experimental results

In this section, we test the performance of the proposed clustering algorithm **SC-SR** using the same evaluation criteria, namely *Purity*, *Normalized Mutual Information (NMI)*, and *Rand Index (RI)*, on the same datasets presented in Chapter 3. We first describe some implementation details of **SC-SR**. Since **SC-SR** is a recursive approach, we need to select two graph layers to fit in the regularization framework at each time. As discussed in Section 4.2.2, we investigate the mutual information between different graph layers. As an example, in the Cora dataset, the "Title" and "Abstract" layers have the highest mutual information between them. Therefore we choose to first combine these two layers. We select the "Abstract" layer to act as G_1 in the spectral regularization framework, as it is considered more informative than the "Title" layer. After the first combination, the third layer "Citation" is incorporated to get the final solution. In addition, at each combination step, there is a regularization parameter α in the optimization problem of Eq. (4.4) to control the relative importance of the two graph layers. Intuitively, the choice of this parameter at each step should loosely reflect the mutual information between the two layers being considered. We use this as a rule of thumb to set the parameters in the first and second combination steps, which are denoted by α_1 and α_2, respectively. As an example, we set $\alpha_1 = 1.9$ and $\alpha_2 = 0.9$ for the Cora dataset.

The clustering results for the baseline methods and **SC-ML** in Section 3.5.2, as well as for **SC-SR** using the first two layers and using all the three layers in each dataset, are summarized in Table 4.1 and Table 4.2 for the peak and average performances of all the algorithms out of 20 test runs, respectively. For each scenario, the two best results are highlighted in bold font. First of all, it is clearly shown that proper combinations of the individual graph layers via **SC-SR** indeed lead to improved clustering results compared to the baselines. In most of the scenarios, **SC-SR** outperforms the two baseline methods **SC-Sum** and **SC-KSum** on the evaluation criteria we consider. On the real world NRC and Cora datasets, **SC-SR** achieves overall better performance than the two baseline methods even when only two layers are considered. This demonstrates the advantages of **SC-SR**, which is able to preserve specificities of the individual layers via the regularization framework, over these baselines that are based on simple averaging of the information from different layers. Second, we can see that the regularized combinations in **SC-SR** consistently lead to better clustering results as more layers are combined, particularly in terms of the *NMI*

Table 4.1: Peak performances of different clustering algorithms out of 20 test runs on one synthetic and two real world datasets.

Algorithm	Purity (%)			NMI (%)			RI (%)		
	Synthetic	NRC	Cora	Synthetic	NRC	Cora	Synthetic	NRC	Cora
SC-Sum	97.3	54.4	**96.9**	91.8	36.1	86.6	97.9	**77.1**	**96.0**
SC-KSum	97.7	53.7	95.2	92.8	34.1	81.2	98.2	76.7	93.8
SC-ML	**98.3**	**61.0**	**98.3**	**94.1**	**41.6**	**91.8**	**98.6**	**79.3**	**97.8**
SC-SR (2)	95.7	57.4	96.2	87.2	40.1	84.9	96.7	70.7	95.1
SC-SR (3)	**98.3**	**58.1**	**96.9**	**94.1**	**42.9**	**87.9**	**98.6**	75.2	**96.0**

Table 4.2: Average performances of different clustering algorithms out of 20 test runs on one synthetic and two real world datasets. The numbers in parenthesis are the standard deviations.

Algorithm	Purity (%)			NMI (%)			RI (%)		
	Synthetic	NRC	Cora	Synthetic	NRC	Cora	Synthetic	NRC	Cora
SC-Sum	**96.4(4.2)**	52.7(2.7)	93.7(9.8)	91.3(2.3)	34.7(1.9)	83.4(9.8)	**97.5(1.9)**	76.7(1.0)	93.6(7.5)
SC-KSum	95.8(6.0)	53.1(1.9)	95.2(0.0)	**91.7(3.4)**	33.9(1.6)	81.2(0.0)	97.3(2.7)	**76.8(1.0)**	93.8(0.0)
SC-ML	**98.2(0.0)**	**58.0(1.5)**	**98.3(0.0)**	**93.8(0.0)**	38.7(1.4)	**91.8(0.0)**	**98.6(0.0)**	**78.4(0.6)**	**97.8(0.0)**
SC-SR (2)	93.8(5.6)	53.2(1.6)	96.2(0.0)	85.8(3.6)	**39.9(1.8)**	84.9(0.0)	95.8(2.6)	70.2(2.2)	95.1(0.0)
SC-SR (3)	94.3(8.1)	**54.9(1.8)**	**96.9(0.0)**	91.2(5.6)	**41.0(1.4)**	**87.9(0.0)**	96.7(4.0)	71.1(2.0)	**96.0(0.0)**

scores.

Compared to **SC-ML**, **SC-SR** does not always perform as well but it still achieves impressive clustering results, especially in terms of the peak performance in *NMI*. To further compare the clustering results obtained by these two methods, we show in Table 4.3 the confusion matrices for them on the three datasets, based on the peak performance out of 20 test runs. The rows of the confusion matrices represent the groundtruth classes and the columns represent the predicted clusters. As we can see from the diagonal entries of the confusion matrices, which are the number of entities that have been correctly identified for each class, the performance of **SC-SR** is comparable to that of **SC-ML** on all the three datasets. In particular, the former outperforms the latter on the NRC dataset. One reason for this is that, the quality of the graph layers in this dataset is lower than that in the other two datasets, and the agreement between different layers is weaker as well, as we can observe from the spy plots shown in Fig. 3.8(b). In such cases, we might need to pay more attention to the specficities of different graph layers to get the best combination, rather than weighting the importance of each layer in a linear fashion. This is where the regularization scheme in **SC-SR** could help.

However, we also notice from the standard deviations shown in Table 4.2 that, the performance of **SC-SR** is statistically less stable than that of **SC-ML**. One explanation for this behavior is that, there are more flexibilities in the implementation of **SC-SR** with more parameter selection procedures involved. Indeed, compared to **SC-ML**, we observe in our experiments that the performance of **SC-SR** is more sensitive to parameter selections. In a scenario where we wish to use

Table 4.3: Confusion matrices for **SC-ML** and **SC-SR** on the (a,d) Synthetic, (b,e) NRC, and (c,f) Cora dataset.

(a)

SC-ML

493	7	0	0	0
6	**491**	3	0	0
0	3	**490**	5	2
0	0	10	**489**	1
6	0	0	0	**494**

diagonal sums up to 2457

(b)

SC-ML

7	4	1	0	1	3	1	2
1	**3**	0	0	0	0	0	2
0	0	**20**	0	0	1	0	0
1	0	0	**10**	1	0	1	0
0	1	0	5	**20**	0	8	0
1	0	0	0	1	**0**	0	0
0	1	0	0	1	0	**0**	0
2	5	5	13	4	0	3	**7**

diagonal sums up to 67

(c)

SC-ML

97	1	0
0	**95**	2
1	1	**95**

diagonal sums up to 287

(d)

SC-SR

491	8	0	0	1
3	**494**	3	0	0
0	7	**486**	7	0
0	0	7	**489**	4
1	0	1	0	**498**

diagonal sums up to 2458

(e)

SC-SR

8	0	0	0	7	0	2	2
4	**1**	0	0	1	0	0	0
0	0	**21**	0	0	0	0	0
0	0	0	**3**	1	1	0	8
0	0	0	0	**29**	2	0	3
0	0	0	0	1	**0**	0	1
0	0	0	0	2	0	**0**	0
5	2	4	2	14	0	0	**12**

diagonal sums up to 74

(f)

SC-SR

98	0	0
1	**91**	5
3	0	**94**

diagonal sums up to 283

cross validation to select the optimal set of parameters, **SC-SR** might thus be more costly than **SC-ML**. Nevertheless, the experimental results demonstrate that **SC-SR** provides an interesting alternative to **SC-ML** in merging multiple graph layers.

Finally, to investigate the potential influence of different orderings of the individual layers in the recursive combination on the clustering performance, we test the performance of **SC-SR** on the synthetic and the Cora datasets with all the six possible orderings of the layers. In more details, in Table 4.4, we first combine layer 1 and layer 2, where layer 1 is used for computing the eigenvectors (G_1 in **SC-SR**), and then include layer 3 in the combination. We implement **SC-SR** with both regularization parameters α_1 and α_2 chosen as $10^{[-2:0.5:2]}$, and report the best performance[4] in Table 4.4. As we can see, in both cases, the ordering of the layers does influence the quality of the result of the recursive combination, particularly so when the layers are of more different qualities, as in the case of the Cora dataset. Moreover, it seems that the choice of layer 1 is a more important factor than the choices of layer 2 and 3. For example, the choice of the "Citation" layer as layer 1 leads to suboptimal performance due to the lower quality of this layer

[4]These results are only based on the discrete values of $10^{[-2:0.5:2]}$ for α_1 and α_2, and are thus illustrative examples that are not necessarily optimal.

Table 4.4: Performance of **SC-SR** on the (a) Synthetic and (b) Cora dataset with different orderings of the layers in the recursive combination.

(a)

Layer 1	Layer 2	Layer 3	Purity (%)	NMI (%)	RI (%)
"R"	"N"	"C"	98.2	94.0	98.6
"N"	"R"	"C"	97.0	91.4	97.7
"C"	"N"	"R"	97.9	92.9	98.4
"N"	"C"	"R"	97.2	91.3	97.8
"C"	"R"	"N"	97.9	92.9	98.3
"R"	"C"	"N"	98.1	93.4	98.5

(b)

Layer 1	Layer 2	Layer 3	Purity (%)	NMI (%)	RI (%)
Abstract	Title	Citation	96.2	84.4	95.1
Title	Abstract	Citation	94.9	83.6	93.5
Citation	Title	Abstract	57.5	21.0	61.9
Title	Citation	Abstract	92.5	77.3	90.4
Citation	Abstract	Title	56.2	21.1	61.7
Abstract	Citation	Title	95.9	83.6	94.7

because of the extreme sparsity of the graph (as shown in Fig. 3.8(c)). These observations could help guide the implementation of **SC-SR** in practice.

4.5 Conclusion

In this chapter, we study the problem of merging multiple graph layers from a signal processing perspective. Specifically, we have designed a framework based on a more detailed understanding of the behavior of the eigenvectors of the graph Laplacian matrix, where information from different layers are combined in a graph regularization process. This way, we are able to treat individual graph layers differently based on their respective importance in a unique and convincing way, which is a potential advantage compared to the approach in Chapter 3. In addition to the clustering performance we achieve in the experiments, we believe that (i) the concept of a joint spectrum shared by multiple graphs is of broad interest in graph-based multi-view data processing tasks, and (ii) the proposed approach is an example that shows how traditional machine learning problems can be approached from a signal processing perspective. In the next chapter, we will follow this direction to study specifically learning problems in the field of signal processing on graphs.

Chapter 5

Graph signal classification: A wavelet-based approach

5.1 Introduction

We consider in this chapter the processing and learning of signals defined on graphs, which we referred to as graph signals. Broadly speaking, a graph signal assigns a scalar value to each vertex (entity) of a graph, based on some information of interest that is related to the entities. The edges of the graph capture the pairwise relationships between these entities. In many applications, various patterns are present in the graph signals; these patterns often carry relevant information about the data. For example, in Fig. 1.2 we see that the graph signals observed in different local regions of the geographical graph correspond to different mobility patterns. In this chapter, we study the problem of classification of graph signals in order to distinguish different information patterns. That is, given a set of signals, each of which is defined on a different weighted and undirected graph, we would like to classify the graph signal patterns into several categories corresponding to different behavior or type of data on the graph.

While the classification problem has been extensively studied in the signal processing, machine learning and pattern recognition communities, the classification of graph signals brings new research challenges: (i) since the graph signals reside in a structured and irregular space, we need to adapt traditional signal processing and learning techniques for efficient signal representation and feature construction on irregular signal structures; (ii) graph signals are less studied than traditional signals such as images and videos and it is necessary to introduce appropriate and meaningful models of graph signals for theoretical analysis of classification problems. From the application side, we also need to demonstrate in real world applications the meaningfulness and efficiency of graph signal classification. These are exactly the challenges that we address in this chapter.

We propose in this chapter to design classification approaches that exploit the characteristics of graph signals in the graph spectral domain. More specifically, we analyze the benefits of adopting spectral representations in graph signal classification. We focus on spectral graph wavelets [62], which are particularly attractive because of their ability to detect local signal variations on graphs. We focus our studies on a generic model for graph signals, namely, the random walk model, and show that the spectral graph wavelets are able to capture local properties of graph signals, which

are of crucial importance in the classification of data that follow this model. As a concrete example of graph signal classification, we then focus on the problem of inference of mobility patterns. Specifically, we design mobility models, which can be considered as specific instances of the generic random walk model. These are tailored for graph signals that correspond to different mobility patterns in local regions of the graph. We then distinguish these patterns by applying the proposed classification framework in the spectral domain. We later present experiments on two sets of real world data, namely, freeway traffic congestion in Alameda County in California and activities of Flickr users in the central area of London. We show that the proposed approach permits to efficiently infer the mobility patterns of both freeway vehicles and Flickr users, by classifying the movements in different geographical locations into one of the proposed mobility patterns. In both synthetic and real world experiments, the proposed spectral features outperform similar features defined using information from the spatial domain only.

The graph signal classification framework proposed in this chapter sheds new lights in studying learning problems related to graph signals, and will benefit in general the understanding and analysis of structured data. It will also open possibilities in a wide range of real world applications, such as mobility, social media and urban dynamics analysis.

The rest of the chapter is organized as follows. In Section 5.2, we formally define the classification problem of graph signals. In Section 5.3, we present the proposed classification framework based on signal representations defined in the graph spectral domain. In Section 5.4, we analyze the efficiency of adopting the representation of spectral graph wavelets for feature construction in the classification of graph signals that follow the random walk model. We start with the generic random walk model, and move onto synthetic models for the mobility patterns. We then present experimental results on synthetic data. In Section 5.5, we apply the proposed classification approach to the problem of inference of mobility patterns on real world traffic and social media data. We conclude the chapter in Section 5.6.

5.2 Problem formulation

We first define formally the graph signal classification problem of this chapter. Specifically, we consider the following problem:

Graph signal classification. *Given a set of graph signals $\{f_i, G_i\}_{i=1}^{M}$ where the signal f_i is defined on the vertex set of weighted and undirected graph G_i, and a set of classes $\{C_s\}_{s=1}^{S}$ that correspond to S different graph signal patterns, the graph signal classification problem is to classify each graph signal $\{f_i, G_i\}$ into one of the classes, C_s, such that it corresponds to the pattern indicated by C_s.*

An illustrative example of the graph signal classification problem is shown in Fig. 5.1, where we are given three graph signals, and the task is to classify them into two classes: the class of smooth graph signals and non-smooth graph signals, respectively. In this case, it is clear that the signal defined on G_1 is a smooth graph signal, since its values vary smoothly along the edges. The other two graph signals are less smooth, because there exist edges between vertices that have quite different signal values. As we can see from this example, we need to take into account the irregular support of the signals, and analyze both the signal values and the graph topologies to do proper classification. To this end, we choose to work with signal representation defined in the graph spectral domain, which

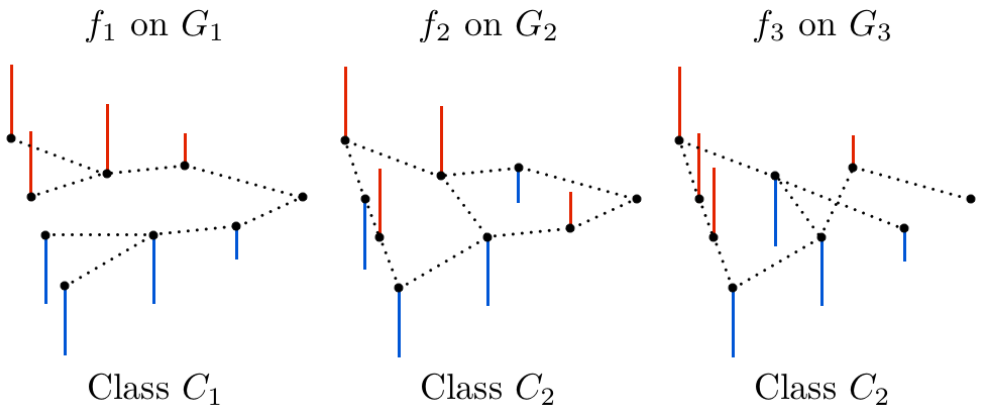


Figure 5.1: Illustration of the graph signal classification problem (this figure is inspired by the one in [119]). In these figures, the red bars pointing upwards and the blue bars pointing downwards represent positive and negative signal values, respectively. The length of the bars reflects the magnitude of the values. In this example, we would like to classify three graph signals into two classes C_1 and C_2, namely, smooth and non-smooth graph signals, respectively.

is able to capture the local variations of the signal with respect to the graph topology. Our goal in this chapter is therefore to build robust and discriminative spectral feature representations that help classify the graph signals according to their similarities with different patterns of interest.

It is important to notice that we consider a very generic setting of the graph signal classification problem, namely, we would like to distinguish a set of graph signals $\{f_i, G_i\}_{i=1}^{M}$ in which both the signal values and the graph topologies can vary. Real world applications usually lead us to deal with graph signals that differ in both signal values and graph topologies, or even those of different dimensions, as we see in the example shown in Fig. 1.2 and in the real world experiments presented later on. Therefore, the feature representations for different graph signals should be defined in a generic way, that does not require the graphs to be identical, and does not depend on the exact dimensions of the signals.

5.3 Framework

5.3.1 Spectral representations of graph signals

Efficient representations of graph signals is of key importance in developing signal processing and learning techniques on graphs. While traditional techniques cannot be directly applied in the graph spatial domain due to its structured and irregular nature, recent development of spectral graph theory [29] permits to define the notion of frequency spectrum for signals defined on graphs. This provides a convenient way to study the characteristics of graph signals, since it allows the generalization of well-understood concepts in signal processing from the traditional setting to the graph setting. In this section, we briefly review the representation of graph signals in the graph spectral domain.

Let us consider a weighted and undirected graph G of n vertices with the unnormalized graph

Laplacian L. Since L is a real and symmetric matrix, it has a complete set of orthonormal eigenvectors[1] $\{\chi_l\}_{l=0}^{n-1}$ (without loss of generality we assume that the eigenvectors are real), with the associated eigenvalues $0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{n-1}$. This enables the generalization of the classical Fourier transform for graph signals. Specifically, let $f : V \to \mathbb{R}^n$ be a signal defined on V, where each entry of the vector f corresponds to the signal value associated with a vertex in V. The graph Fourier transform (GFT) [62] can then be defined as the inner product of the signal f and the eigenvectors $\{\chi_l\}_{l=0}^{n-1}$ of the graph Laplacian L:

$$\hat{f}(\lambda_l) = \langle f, \chi_l \rangle = \sum_{i=1}^{n} \chi_l(i) f(i). \tag{5.1}$$

As we can see, similarly to the classical setting, the eigenvalues and eigenvectors of the graph Laplacian carry a notion of frequency for graph signals, and the GFT decomposes the graph signals into different frequency components, where low frequency components correspond to less oscillation with respect to the graph topology than high frequency components.

Based on the frequency notion and the GFT, it becomes possible to generalize well-developed signal processing techniques in the traditional setting, such as the classical wavelet transform, for graph signals. Several works in the literature have proposed wavelet transforms defined on graphs, for example the lifting-based and tree-based transforms proposed in [93] and [50], respectively, the diffusion wavelets proposed in [30] which are constructed based on a diffusion operator, and the graph-QMF filterbanks proposed in [94] which are constructed based on two-channel filterbanks on graphs. However, of particular interest is the recently proposed spectral graph wavelet transform (SGWT) [62], which has several convenient properties for the analysis proposed in this chapter. First, the SGWT is defined in precise analogy to the classical continuous wavelet transform, which is easy to understand and implement. Second, the SGWT brings an explicit notion of scales in the signal decomposition. Third, the SGWT can be computed in a computationally efficient manner, without the need for computing a full eigendecomposition of the graph Laplacian matrix. This is crucial for dealing with large scale graphs. Specifically, for a bandpass filter g that satisfies the admissibility condition for wavelets, the spectral graph wavelets defined at scale $\{t_j\}_{j=2}^{J}$ can be written as:

$$\Psi_{t_j} = \chi g(t_j \Lambda) \chi^T, \tag{5.2}$$

where Λ is the diagonal matrix that contains the eigenvalues $\{\lambda_l\}_{l=0}^{n-1}$ of the graph Laplacian along the diagonal, χ is the matrix that contains the eigenvectors $\{\chi_l\}_{l=0}^{n-1}$ as columns, and J is the total number of scales. For a discrete set of scales[2] $\{t_j\}_{j=1}^{J}$, the spectral graph wavelets constitute a frame in $\mathbb{R}^n$. The SGWT is then computed by taking the inner product between the signal and each atom in the wavelet frame.

[1] From this chapter onwards we will denote χ as the eigenvectors of the graph Laplacian, and use the indexes from 0 to $n-1$ for the eigenvectors and eigenvalues, which are more commonly adopted in signal processing.

[2] In the construction of SGWT, the wavelets at scale t_1 (which are called spectral graph scaling functions in [62]) are defined via a low pass filter h, which covers the low frequencies in the spectrum.

5.3.2 Graph signal classification framework

The key ingredient of the proposed classification framework is the representation of graph signals using features extracted in the graph spectral domain. Once spectral features are defined and computed, the classification is implemented via traditional learning models such as the support vector machines (SVMs). The proposed framework is summarized in Algorithm 4.

Algorithm 4 Spectral Classification of Graph Signals

1: **Input:**
$\{f_m, G_m\}_{m=1}^M$: M graph signals, where f_m is defined on the vertex set of weighted and undirected graph G_m of n vertices
$\mathcal{I}_1$: Training indexes
$\mathcal{L}$: Training labels
$\mathcal{I}_2$: Testing indexes
2: **Training phase:** For each $i \in \mathcal{I}_1$, compute the spectral features ϕ_i using f_i and G_i.
3: Train the classifier with the features $\{\phi_i\}$ and $\mathcal{L}$.
4: **Testing phase:** For each $j \in \mathcal{I}_2$, compute the spectral features ϕ_j using f_j and G_j.
5: Apply the trained classifier to the features $\{\phi_j\}$ to classify the testing signals $\{f_j, G_j\}$.
6: **Output:** Predicted class for each graph signal $\{f_j, G_j\}$ where $j \in \mathcal{I}_2$.

As we have explained, both the GFT and the SGWT coefficients at different scales can be considered for representation of the graph signal in the graph spectral domain. However, for the classification of graph signals, we would like to build discriminative features ϕ from the spectral representation for separating signals belonging to different classes. For signals that have localized information, SGWT is a more attractive representation than GFT because of its ability to detect local signal variations on graphs. In the next section, we analyze the efficiency of adopting the spectral representation of SGWT for feature construction in graph signal classification applications, for graph signals that follow a random walk model. We will later propose specific forms of features ϕ in our classification experiments, which are conducted for synthetic signals as well as real world data.

5.4 Analysis of the random walk model

To analyze the efficiency of adopting spectral representations and SGWT for feature design in classification tasks, we first need to introduce meaningful models for graph signals. In this section, we discuss the following random walk model. Given a graph signal $\{f, G\}$, we consider f to be the stationary distribution of a random walk process defined on a weighted and undirected graph G. A random walk on a graph can be considered as a Markov process whose transition probabilities are determined by the adjacency matrix of the graph. This model is interesting to study because (i) random walks on graphs are generic and common models, and (ii) they are closely related to diffusion processes associated with graphs in physics, and can potentially be applied to analyze information propagation in social networks and mobility patterns in transportation networks. In this section, we first present classification analysis of graph signals that follow the generic random

walk model, and then move onto analyzing specific instances of this model that are related to mobility patterns.

5.4.1 Classification analysis of the generic random walk model

Assuming that the state of the random walk, namely, the probabilities of staying at any of the n vertices in the graph, at a discrete time stamp t is captured by a vector $p(t) \in \mathbb{R}^n$. Then, the state at the next time stamp $t+1$ is computed as:

$$p(t+1) = {W_{\text{tran}}}^T p(t), \tag{5.3}$$

where $W_{\text{tran}} = D^{-1}W$ is the transition probability matrix. Denote the stationary distribution of a random walk on the graph as $\pi \in \mathbb{R}^n$, where $\pi(i)$ is the probability of staying at vertex i when the random walk process converges. We have the following lemma.

Lemma 1 (Chung [29]). *The stationary distribution $\pi \in \mathbb{R}^n$ of a random walk on a connected graph G is:*

$$\pi(i) = \frac{d(i)}{vol(G)}, \tag{5.4}$$

where $d(i)$ is the degree of vertex i, and $vol(G)$ is the volume of the graph (sum of degrees of all the vertices).

Note that the existence of a stationary distribution does not guarantee that the random walk will converge to that stationary distribution starting from any initial distribution. However, if the graph is non-bipartite, then the stationary distribution π in Eq. (5.4) is unique, and for any initial distribution, the random walk eventually converges to π, which is proportional to the degree sequence of the graph.

The stationary distribution of a random walk usually cannot be described by a single probability distribution, since the degree sequence of the graph can possibly follow many different probability distributions, depending on the connectivity and edge weights. In this case, the SGWT possesses important properties that leads to discriminative features in classification. Consider two graph signals $\{f_1, G_1\}$ and $\{f_2, G_2\}$, where f_1 and f_2 are defined as the degree sequence of G_1 and G_2, according to the random walk model. Consider that the only difference between the two graphs G_1 and G_2 is the edge weight between a specific vertex pair i and j. In this case, a local change, namely, the change of one edge weight between G_1 to G_2, leads to different local variations around vertices i and j in f_1 and f_2. To distinguish these two graph signals, it is thus important to adopt representations that are able to capture such local variations. From this viewpoint, the SGWT is attractive because of its ability to detect local signal variations with respect to the graph topology.

To see this more clearly, let us consider the example of two graph signals $\{f_1, G_1\}$ and $\{f_2, G_2\}$ shown in Fig. 5.2. These signals are defined as the degree sequences of the graphs, according to the random walk model. In G_1, apart from the unitary edge weights, there is an edge with a rather large weight of 3 between v_3 and v_4, while in G_2 there are three edges with an intermediate weight of 2. We would like to distinguish these two graph signals by analyzing their local properties. To this end, we compute the GFT coefficients, and the SGWT coefficients corresponding to high frequency

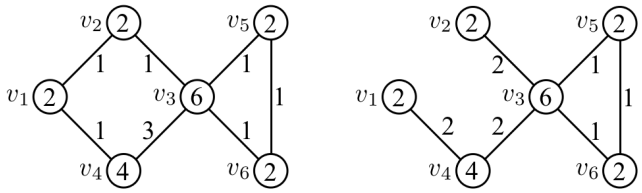


Figure 5.2: Two graph signals: (a) $\{f_1, G_1\}$ and (b) $\{f_2, G_2\}$. The signal values are shown in the circles, and the edge weights are shown alongside the edges.

Table 5.1: The GFT coefficients, and the SGWT coefficients corresponding to high frequency components in a 3-scale decomposition, for the signals f_1 and f_2 shown in Fig.(5.2).

	GFT coefficients global representation						**SGWT coefficients (high freq.)** v_1	v_2	v_3	v_4	v_5	v_6
f_1	7.35	0.07	2.20	2.12	0.55	2.09	0.05	0.32	1.78	0.92	0.30	0.30
f_2	7.35	0.51	0.63	0.00	2.84	2.30	0.12	0.84	2.15	0.48	0.36	0.36

components in a 3-scale decomposition[3], as shown in Table 5.1. We see clearly that in case of f_1, due to a large edge weight between v_3 and v_4, the signal variations are large between these vertices and their neighbors, resulting in much larger wavelet coefficients at v_3 (1.78) and v_4 (0.92) than at other vertices. This is an example of an edge with large weight leading to local signal variations. Compared to f_1, we see in f_2 that three edges of weight 2 lead to increasing local variations around v_1, v_2 and v_3, which are indicated by the larger wavelet coefficients at these vertices; in contrast, the variation around v_4 decreases relatively, as a smaller wavelet coefficient suggests. While the SGWT coefficients capture such information about local variations of the signals, it is neither reflected in the GFT coefficients, which only evaluates the signal variations in a global manner, nor in the signal values in the spatial domain, which happen to be exactly the same with two different graphs in Fig. 5.2. Therefore, to distinguish the two graph signals in this example, the SGWT coefficients is a better representation than the GFT coefficients or the signal values. This example demonstrates the importance of adopting representations that are able to detect local signal variations, such as the SGWT, in classifying graph signals that follow the random walk model.

5.4.2 Classification analysis of the mobility models

To analyze quantitatively the benefit of adopting the SGWT in graph signal classification, we focus now on specific instances of the generic random walk model, namely, models that are related to mobility patterns. We are interested in such mobility models because (i) they can be naturally defined based on the generic random walk model, since mobility patterns can be described as

[3]Throughout this chapter, we follow the configurations proposed in [62] to implement the SGWT.

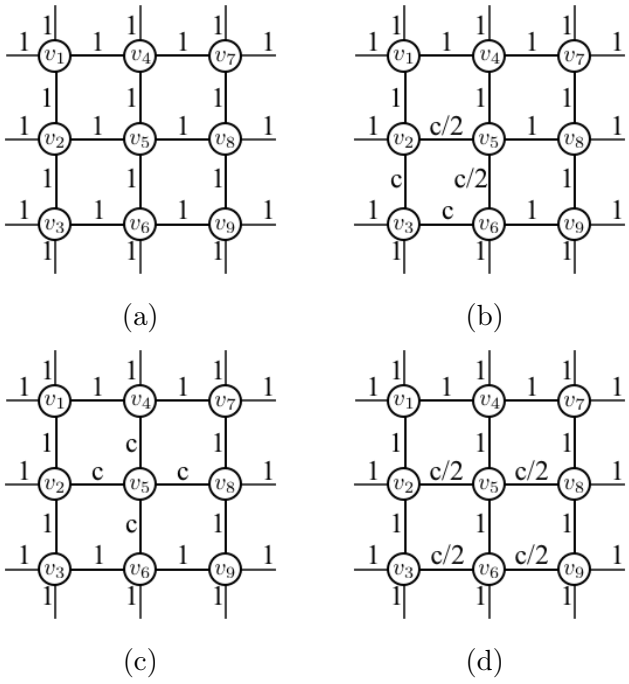


Figure 5.3: The four mobility models: (a) Random, (b) Spreading, (c) Gathering and (d) Bi-directional.

random walk processes on graphs, and (ii) they can be used for solving real world problems, such as mobility inference from traffic and social media data.

We propose below specific models that correspond to four mobility patterns, which are defined based on random walk processes on graphs, with different sets of edge weights determining the transition probabilities of different random walk processes. Specifically, we consider a local 3×3 region in a two-dimensional lattice (namely, a four-nearest neighbor regular grid), and we define four mobility patterns based on the graphs shown in Fig. 5.3. For the sake of simplicity, we only have one parameter c in our analysis. This parameter is assumed to be greater than 2 so that the transition probabilities in different random walk processes would be different[4]. In more details:

(a) the *Random* pattern represents a uniform random walk in all directions. This is suitable for areas where walkers follow any path at random.

(b) the *Spreading* pattern represents flows increasing smoothly from top right vertex v_7 to bottom left vertex v_3. This pattern represents situations where walkers emerge at certain locations and move towards a specific point nearby.

(c) the *Gathering* pattern represents strong movements towards the central vertex v_5. This represents the case where walkers converge to the same location from different directions.

[4]We note that the settings shown in Fig. 5.3 are illustrative examples, which have been extended in the experiments in Section 5.4.3.

(d) the *Bi-directional* pattern represents dominant movements along two lines connecting vertices v_2, v_5, v_8 and vertices v_3, v_6 and v_9. Walkers in this case move preferably with clear directionalities.

The graphs shown in Fig. 5.3 are connected and non-bipartite. Therefore, according to Lemma 1, the random walk processes defined on these graphs will converge to different stationary distributions. These distributions are proportional to the respective degree sequences and can be expressed analytically in terms of the parameter c. For example, for the *Spreading* pattern, the stationary distribution is proportional to the degree sequence $[4\ 2+\frac{3}{2}c\ 2+2c\ 4\ 2+c\ 2+\frac{3}{2}c\ 4\ 4\ 4]^T$. Clearly, under the random walk model, the resulting graph signals become quite different for large values of c, and their spectral representations become quite distinctive. More specifically, Fig. 5.4 shows the signals in column 1 and their SGWT coefficients in three subbands in column 2 (low frequency), 3 (middle frequency), and 4 (high frequency), for the four proposed mobility patterns where $c = 10$. As expected, in the *Random* pattern, the spectrum is flat in all subbands, with most of the energy of the coefficients being contained in the low frequency subband. In the *Gathering* pattern, the peak of the signal is reflected by the peak in the high frequency subband. In *Spreading*, the smooth increase of the signal results in a spectrum in the low frequency subband that is not flat, and leads to a ramp-shape spectrum in the middle frequency subband, with increased energy in this subband. Finally, in *Bi-directional*, the movements along two preferred paths on the graph can be observed by the shape of the spectrum in the middle and high frequency subbands. In terms of energy distribution, the low and middle frequency subbands contain an intermediate percentage of energy compared to *Random* and *Spreading*, but the high frequency subband clearly contains a larger percentage of energy. In summary, for the four types of mobility patterns we consider, the SGWT provides a representation that is pretty different in the different patterns. Such differences mainly lie in the shape of the spectrum in different subbands, namely in the magnitude of the SGWT coefficients. This motivates us to consider this representation to design features for classification. Specifically, we consider spectral features that capture the behaviors of the SGWT coefficients in all the three subbands.

5.4.3 Experiments on synthetic data

In this section, we show the efficiency of features built on the SGWT representation for graph signal classification in synthetic data representing simple mobility patterns. Based on the spectral characteristics shown in Fig. 5.4, and on the understanding that statistical information about the SGWT coefficients can serve as discriminative features in the classification task, we propose to construct a 6-dimensional feature vector ϕ_{spectral}, which is computed with a 3-scale SGWT of the graph signal:

$$\phi_{\text{spectral}}(s) = \begin{cases} \text{mean}(C_{\text{low}})/C_{\max}, & s = 1, \\ \{\max(C_{\text{low}}) - \min(C_{\text{low}})\}/C_{\max}, & s = 2, \\ \text{mean}(C_{\text{mid}})/C_{\max}, & s = 3, \\ \{\max(C_{\text{mid}}) - \min(C_{\text{mid}})\}/C_{\max}, & s = 4, \\ \text{mean}(C_{\text{hig}})/C_{\max}, & s = 5, \\ \{\max(C_{\text{hig}}) - \min(C_{\text{hig}})\}/C_{\max}, & s = 6, \end{cases} \tag{5.5}$$

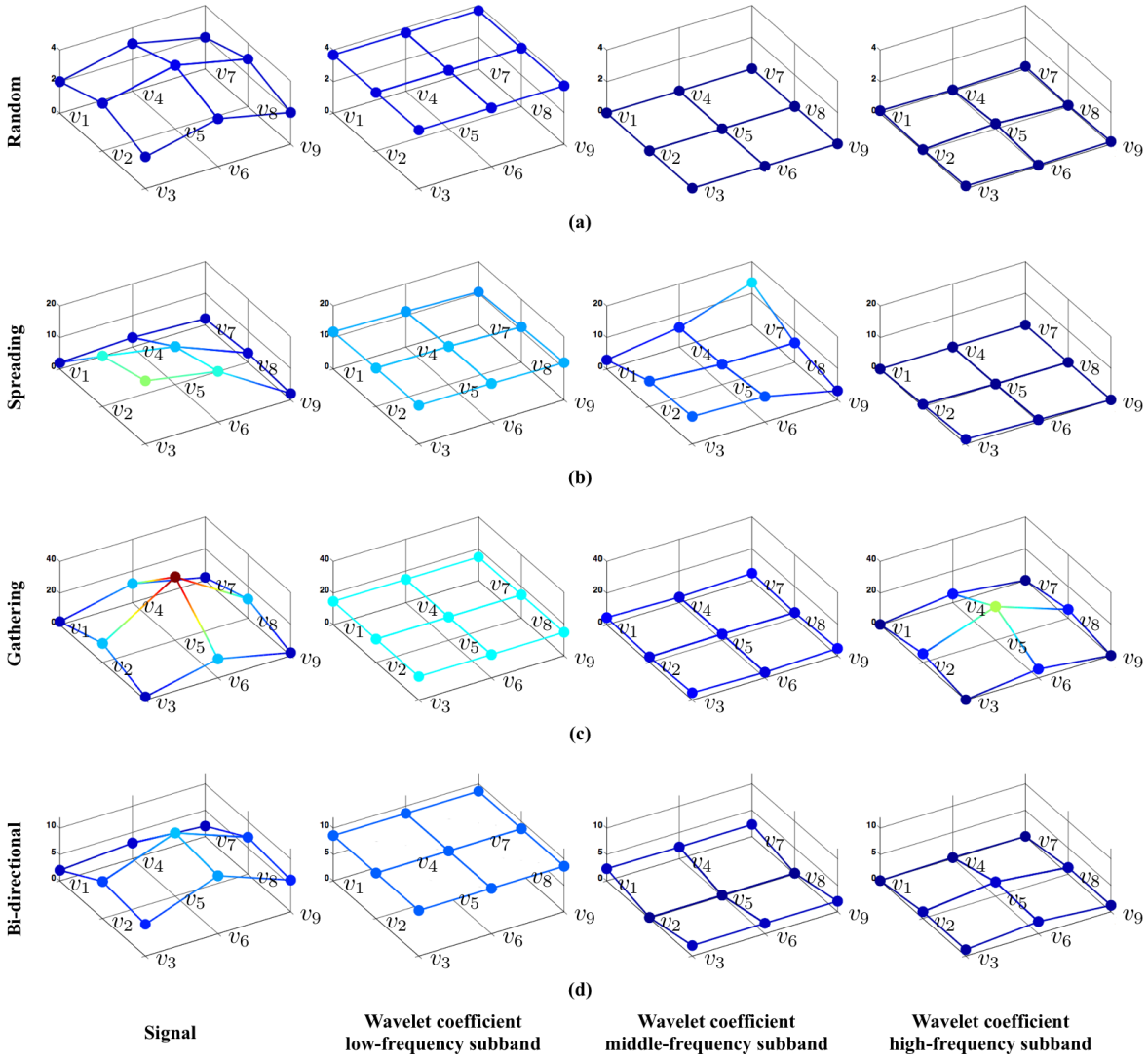


Figure 5.4: Random walk signals and their SGWT coefficients in three subbands for the four proposed mobility patterns: (a) Random, (b) Spreading, (c) Gathering and (d) Bi-directional. The vertices are color-coded according to the associated values of the signals or coefficients. Note that the colors of the edges are linear interpolations of the colors of their end vertices and do not represent the edge weights.

where C_{low}, C_{mid}, C_{hig} denote the magnitudes of wavelet coefficients in low, middle and high frequency subbands, respectively, $C_{\max}$ denotes the maximum magnitude of all wavelet coefficients, and mean$(\cdot)$, max$(\cdot)$ and min$(\cdot)$ denote the operators taking the mean, maximum and minimum values, respectively. We use a 3-scale SGWT since it can be considered as a standard choice where we capture the low, middle and high frequency components of the signal. Intuitively, the feature vector ϕ_{spectral} captures the variations of the SGWT coefficients in three different subbands, which together reflect both the global and local variations of the graph signals, hence it is likely to be discriminative in the classification task. We notice that the features defined as in Eq. (5.5) are an example choice that capture important characteristics of graph signals, namely, the local variations reflected by the magnitude changes of the SGWT coefficients in different subbands. Other features can certainly be useful if they are more suitable for the specific models of interest; however, feature optimization is not the main objective here.

In our experiments, we also propose one set of features defined similarly in the spatial domain. Specifically, although it is less straightforward to have the notion of scales in the spatial domain, we can define a 6-dimensional spatial feature vector as follows, for the purpose of comparison:

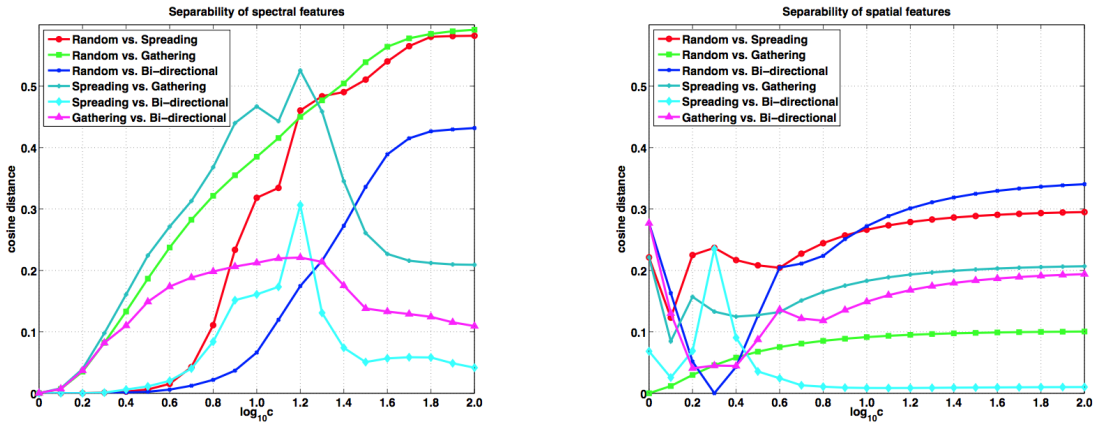
$$\phi_{\text{spatial}}(s) = \begin{cases} \text{mean}(\Delta_1)/\Delta_{\max}, & s = 1, \\ \{\max(\Delta_1) - \min(\Delta_1)\}/\Delta_{\max}, & s = 2, \\ \text{mean}(\Delta_2)/\Delta_{\max}, & s = 3, \\ \{\max(\Delta_2) - \min(\Delta_2)\}/\Delta_{\max}, & s = 4, \\ \text{mean}(\Delta_3)/\Delta_{\max}, & s = 5, \\ \{\max(\Delta_3) - \min(\Delta_3)\}/\Delta_{\max}, & s = 6, \end{cases} \tag{5.6}$$

where $\Delta_k \in \mathbb{R}^{d_k}$ is a vector that contains absolute differences in signal values between vertices that have a shortest path of length k (with d_k being the number of such vertex pairs)[5], and $\Delta_{\max}$ represents the maximum absolute difference in the values of the original signal. Therefore, ϕ_{spatial} is designed with a similar philosophy as ϕ_{spectral} to capture signal variations, which are important in classification as discussed above, but only using information obtained directly from the graph spatial domain. Furthermore, ϕ_{spatial} defined in this way do not have a frequency interpretation. It is important to notice that, since both ϕ_{spectral} and ϕ_{spatial} are specific choices, the comparison between them do not serve as a general comparison between any spectral or spatial features in an attempt to show that one is always better than the other. Our purpose here is rather to demonstrate the differences between the two types of features that are similarly defined in both domains. In addition to the proposed features ϕ_{spectral} and ϕ_{spatial}, we also include a 6-dimensional feature vector that is based on the Laplacian quadratic form:

$$\phi_{\text{spatialLQ}}(s) = f^T L^s f, \text{ for } s = 1, \ldots, 6. \tag{5.7}$$

$\phi_{\text{spatialLQ}}$ can be considered as a spatial feature vector that measures the s-hop signal variations with respect to the graph; however, it also has a frequency interpretation. Specifically, $f^T L^s f$ can be interpreted as the weighted sum of the squared GFT coefficients (since $f^T L^s f = f^T \chi \Lambda^s \chi^T f = \sum_{l=0}^{n-1} \lambda_l^s \hat{f}(\lambda_l)^2$). We nevertheless include it in our experiments for a more complete comparison.

[5]Notice here that we only consider the shortest-path distance between vertex pairs and do not consider the edge weights.



(a) (b)

Figure 5.5: Pairwise cosine distances between (a) the features ϕ_{spectral} and (b) the features ϕ_{spatial} of different mobility patterns.

Finally, we consider two standard PCA-based features. Specifically, we define $\phi^{\text{G}}_{\text{PCA}}$ and $\phi^{\text{W}}_{\text{PCA}}$ as the 6-dimensional projections of the GFT and SGWT coefficients of the graph signals, respectively, onto their principal components computed via PCA.

We first test the efficiency of the spectral features ϕ_{spectral} of Eq. (5.5) in classifying graph signals that correspond to the mobility models proposed above. We first investigate how the choice of parameter c in Fig. 5.3 influences the discriminating power of the spectral features ϕ_{spectral}. To this end, we analyze the separability of ϕ_{spectral} computed based on the four mobility models, for different values of the parameter c. Specifically, for a given c, we consider the graph signals as the stationary distributions of the random walks on the respective graphs. We then compute the spectral representation via a 3-scale SGWT and the feature vector ϕ_{spectral} for each signal class. Finally, we compute the pairwise cosine distances between feature vectors of different classes. The results are shown in Fig. 5.5(a). As expected, the separability of ϕ_{spectral} increases when the value of c increases, as the differences in the proposed patterns become prominent. Interestingly, while the distances between *Random* and other patterns keep increasing, the distances among the other three patterns start decreasing when c gets relatively large. One explanation for this behavior is that, when c gets relatively large, the edges with large weights become dominant and their specific locations within the graph become less relevant. As a comparison, similar separability performance results are shown in Fig. 5.5(b) for the spatial features ϕ_{spatial} of Eq. (5.6). Clearly, ϕ_{spectral} presents better separability between the four mobility patterns than ϕ_{spatial} when the value of c is reasonably large, that is, when the random walk signals get quite different.

Next, we test the performance of ϕ_{spectral} and ϕ_{spatial}, together with the other three features $\phi_{\text{spatialLQ}}$, $\phi^{\text{G}}_{\text{PCA}}$ and $\phi^{\text{W}}_{\text{PCA}}$, in simple classification tasks. In our experiments, we first extend the four basic mobility models defined in Fig. 5.3 to generate more realistic graph signals for classification. Specifically, we consider a 10×10 eight-nearest neighbor regular grid, where the edges have unitary weights for the *Random* pattern. For the *Spreading* pattern, we randomly select a local area in the

Table 5.2: Classification performances of the five classifiers in the synthetic experiments based on the random walk model.

	Accuracy	NMI	RI
ϕ_{spectral}	**0.9880**	**0.9612**	**0.9882**
ϕ_{spatial}	0.8680	0.7347	0.8925
$\phi_{\text{spatialLQ}}$	0.9560	0.8850	0.9595
$\phi^{\text{G}}_{\text{PCA}}$	0.7500	0.5631	0.8160
$\phi^{\text{W}}_{\text{PCA}}$	0.6160	0.3020	0.7382

graph with 10 vertices. We set the weights of the edges between these vertices to increase from 2 to 4 towards a certain direction. For the *Gathering* pattern, we define an edge weight of 5 for the eight edges surrounding a certain vertex chosen at random. Finally, for the *Bi-directional* pattern, we randomly choose two adjacent parallel paths in the graph, and we set the edges between the vertices to have a weight of 2, except for two vertices that have connecting edges with a weight of 4. The main motivation for the setting in this pattern is to enable more complex mobility patterns that might appear in real world applications. We create many instances of such graph models for both training and testing[6]. We also add zero-mean Gaussian noise with standard deviation 0.1 to all the edge weights.

We then associate random walk processes to the above graphs where the transition probabilities are defined as in Eq. (5.3). The synthetic graph signals are finally generated by aggregating 100 random walk instances of up to 50 steps with random starting points in the graphs. Such aggregate signals are more realistic than the stationary distributions of the random walks on graphs; real world observations do not necessarily correspond to the stationary state of the random walk process. Each synthetic signal counts how many times a random walker has passed by each vertex of the graph. In our experiments, we generate 1000 synthetic signals, 250 for each mobility model. In each case, we take 125 signals for training and the rest for testing.

We train five SVM classifiers on the above training signals using the feature vectors ϕ_{spectral}, ϕ_{spatial}, $\phi_{\text{spatialLQ}}$, $\phi^{\text{G}}_{\text{PCA}}$ and $\phi^{\text{W}}_{\text{PCA}}$, with the LIBSVM library [23]. For the latter three feature vectors, we scale each feature in the vector to the range of [0 1]. The hyperparameters in SVM are optimized in the training stage using 5-fold cross validation and grid search. We then apply the trained classifiers to the testing data and evaluate their performances. We adopt three evaluation criteria in our experiments, namely, *Accuracy*, *Normalized Mutual Information (NMI)* defined in Eq. (3.22) and *Rand Index (RI)* defined in Eq. (3.23). Specifically, *Accuracy* measures the percentage of the data samples that have been correctly classified, and *NMI* and *RI* evaluate the quality of the obtained labels from an information-theoretic and information retrieval point of view, respectively. The classification results are shown in Table 5.2 for the five different types of features. As we

[6]Notice that these graphs are example choices created for the four mobility patterns. Other ways of generating graphs are certainly possible.

Table 5.3: Confusion matrices for classifiers based on (a) ϕ_{spectral}, (b) ϕ_{spatial}, (c) $\phi_{\text{spatialLQ}}$, (d) $\phi_{\text{PCA}}^{\text{G}}$ and (e) $\phi_{\text{PCA}}^{\text{W}}$ in the synthetic experiments based on the random walk model.

(a) ϕ_{spectral}

	Predicted Random	Predicted Spreading	Predicted Gathering	Predicted Bi-directional
Groundtruth Random	**124**	0	0	1
Groundtruth Spreading	0	**125**	0	1
Groundtruth Gathering	0	0	**125**	0
Groundtruth Bi-directional	0	5	0	**120**

(b) ϕ_{spatial}

	Predicted Random	Predicted Spreading	Predicted Gathering	Predicted Bi-directional
Groundtruth Random	**116**	5	0	4
Groundtruth Spreading	0	**104**	0	21
Groundtruth Gathering	0	1	**124**	0
Groundtruth Bi-directional	2	33	0	**90**

(c) $\phi_{\text{spatialLQ}}$

	Predicted Random	Predicted Spreading	Predicted Gathering	Predicted Bi-directional
Groundtruth Random	**125**	0	0	1
Groundtruth Spreading	2	**114**	0	9
Groundtruth Gathering	0	0	**125**	0
Groundtruth Bi-directional	0	11	0	**114**

(d) $\phi_{\text{PCA}}^{\text{G}}$

	Predicted Random	Predicted Spreading	Predicted Gathering	Predicted Bi-directional
Groundtruth Random	**105**	20	0	0
Groundtruth Spreading	43	**67**	15	0
Groundtruth Gathering	2	32	**91**	0
Groundtruth Bi-directional	0	1	12	**112**

(e) $\phi_{\text{PCA}}^{\text{W}}$

	Predicted Random	Predicted Spreading	Predicted Gathering	Predicted Bi-directional
Groundtruth Random	**117**	1	1	6
Groundtruth Spreading	10	**64**	28	23
Groundtruth Gathering	9	27	**71**	18
Groundtruth Bi-directional	15	38	16	**56**

can see, the classifier based on the spectral features ϕ_{spectral} outperforms the other four classifiers in terms of all the three evaluation criteria. To further investigate the performance in terms of the individual classes, the confusion matrices for the five classifiers are shown in Table 5.3, whose columns represent the predicted classes and the rows represent the groundtruth classes. Looking at the diagonal of the confusion matrices, which are the number of signals that have been correctly identified for each class, we see the superiority of the spectral features ϕ_{spectral} over the other four features in the comparison. Next, the performance of the classifier based on the features $\phi_{\text{spatialLQ}}$ is slightly worse in Table 5.2 and Table 5.3, possibly due to the fact that the GFT coefficients only capture the global signal variations, while the SGWT coefficients are able to capture the local signal changes. Indeed, from Table 5.3 we see that the misclassified data samples by the classifier based on $\phi_{\text{spatialLQ}}$ are mainly those that correspond to the *Spreading* and *Bi-directional* patterns, which are the two mobility patterns that most likely have similar global signal variations. For the same reason, the performance of the spatial features ϕ_{spatial} is less satisfactory, which suggests that these

spatial features are inefficient in separating random walk signals that might have similar global but different local variations. Finally, the experimental results regarding the PCA-based features $\phi^{\mathrm{G}}_{\mathrm{PCA}}$ and $\phi^{\mathrm{W}}_{\mathrm{PCA}}$ suggest that, although the spectral representation of graph signals is meaningful in general, the feature construction process is still important in the classification applications. In particular, it is important that the features are adapted to the application and task at hand, like those of ϕ_{spectral} proposed in Eq. (5.5). PCA-based features, as in the traditional signal processing and learning problems, are efficient for the representation of the data, but not reliable and consistent enough for discrimination [39].

5.4.4 Discussion

The random walk model considered in this section is a model where we observe local variations of the graph signals. Such local properties are of crucial importance in the classification tasks. Wavelet-based features, such as the proposed features ϕ_{spectral} based on the SGWT, are particularly efficient in this case. Other signal models can be of interest in the classification studies as well. For example, we can also consider a Gaussian Markov Random Field (GMRF) model for the graph signal $\{f, G\}$, which assumes that f is a GMRF with respect to G, with the precision matrix in the GMRF chosen as the graph Laplacian L. This model essentially assumes that the graph signal follows a multivariate Gaussian distribution, and turns out to be a smooth signal model on graphs. In such case, the graph signals are globally smooth and do not have significant local variations, and the benefits of wavelet-based features are less obvious. We leave a detailed classification studies of this model in Appendix A.

5.5 Mobility inference experiments

5.5.1 Experimental settings

In this section, we evaluate the performance of the classifiers based on the spectral and spatial features in the classification of two real world datasets. The task that we propose to solve is the inference of mobility patterns at different geographical locations, by classifying the graph signals extracted from the neighborhood of each vertex in a geographical graph into one of the mobility models proposed earlier. The classification algorithm is summarized in Algorithm 5. It is worth noting that, first, since the K-hop neighborhoods around different vertices in an irregular graph are usually of different sizes, in Algorithm 5 we are able to deal with graph signals of different dimensions. This can be considered as a potential benefit of the proposed framework. Second, these graph signals are real world observations in the local regions of the geographical graph, hence do not necessarily follow strictly a random walk model. However, there are still links between these real world observations and the random walk signals that we considered in the synthetic experiments, as we will explain below. Third, Algorithm 5 can scale to real world graphs with large numbers of vertices, since the computation is done only on local patches of the graph. In our experiments, we compare Algorithm 5 with the same classification procedure but using the spatial features ϕ_{spatial}. We focus on the comparison between ϕ_{spectral} and ϕ_{spatial} since they are similarly defined in the respective domains. Similarly to the synthetic experiments, we use the LIBSVM

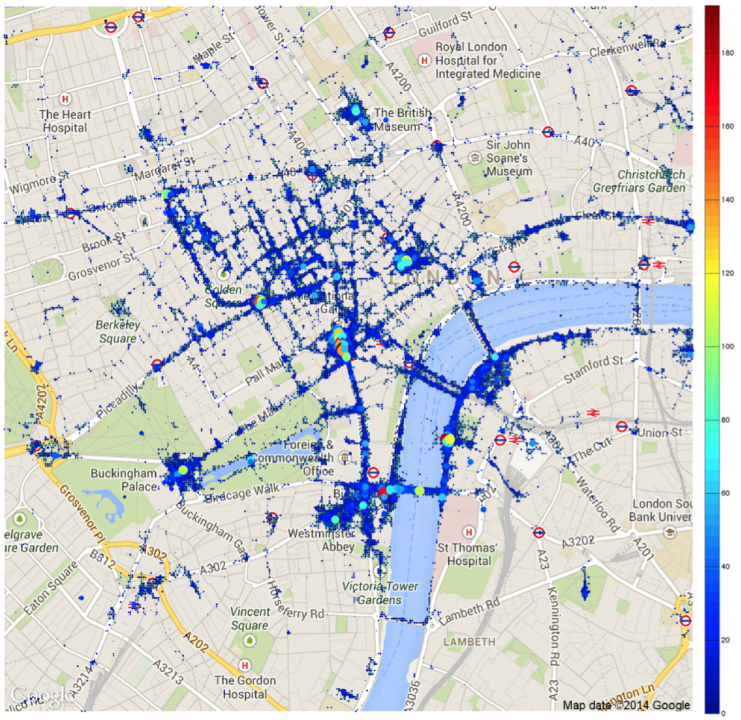
library for classification and the hyperparameters in SVM are optimized in the training stage using 5-fold cross validation and grid search.

Algorithm 5 Mobility Inference via Spectral Classification

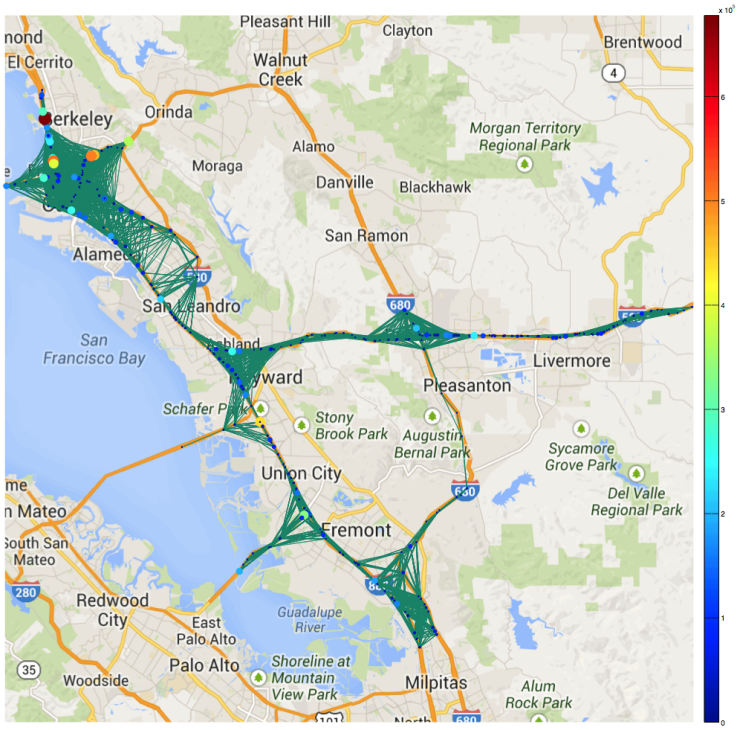
1: **Input:**
$\{f, G\}$: A graph signal where f is defined on the vertex set of a weighted and undirected geographical graph G of n vertices
$\mathcal{I}_1$: Training indexes
$\mathcal{L}$: Training labels (mobility patterns)
$\mathcal{I}_2$: Testing indexes
2: **Training phase:** For each vertex $i \in \mathcal{I}_1$ in G, generate a subgraph G_i that contains the K-hop neighbors of i. Generate the truncated signal f_i that corresponds to G_i.
3: Compute the spectral features ϕ_{spectral_i} using f_i and G_i.
4: Train the classifier with the features $\{\phi_{\text{spectral}_i}\}$ and $\mathcal{L}$.
5: **Testing phase:** For each vertex $j \in \mathcal{I}_2$ in G, generate a subgraph G_j that contains the K-hop neighbors of j. Generate the truncated signal f_j that corresponds to G_j.
6: Compute the spectral features ϕ_{spectral_j} using f_j and G_j.
7: Apply the trained classifier to the features $\{\phi_{\text{spectral}_j}\}$ to classify the testing signals $\{f_j, G_j\}$.
8: **Output:** Predicted mobility patterns for the K-hop neighborhood of each $j \in \mathcal{I}_2$.

We consider two sets of data from traffic networks and social media platforms, respectively. In the first dataset, the graph signals correspond to the aggregated numbers of distinct Flickr users that have taken photos at different geographical locations in the central area of London between January 2010 and June 2012. Specifically, the map of central London is fractioned into geographical cells of 10 meters by 10 meters, each of which corresponds to a vertex in our graph. The graph is constructed by assigning an edge between two vertices when the geographical distance between them is shorter than 30 meters; the edge weight is set to be inversely proportional to the distance. In the second dataset, the graph signals correspond to the accumulated traffic delays (in minutes) at different locations of the freeways in Alameda County in California between January 2007 and May 2013. Again, a geographical graph is built by connecting every pair of locations whose distance is smaller than a threshold, with the edge weight being inversely proportional to the geographical distance. The signals and graphs in both examples are shown in Fig. 5.6. It is clear that, in both cases, the observed graph signal is a global aggregation of individual mobility traces at different local regions of the geographical graph, thus it can be considered as a real world observation that is related to the mobility models proposed in Section 5.4.2. We remark that in the case of Flickr data, the "mobility" of a Flickr user is defined as a sequence of discrete points on the map where photos were taken; we acknowledge that this is only an approximation of the actual dynamics of the movement of the user.

It is important to notice that, contrarily to the random walk models, the edge weights in these real world graphs are defined based on geographical distances, which do not reflect directly the underlying transition probabilities. Nevertheless, we can still consider that these geographical graphs contain partial knowledge about the transition probabilities underlying the observed signals. In particular, transitions only happen between vertices that are connected by a path, for example road, street or bridge. We would like to remark that, (i) this setting still fits our generic classification



(a)



(b)

Figure 5.6: (a) The aggregated number of distinct Flickr users that have taken photos at different locations in the central area of London from January 2010 to June 2012. (b) The accumulated traffic delays (in minutes) at different locations of the freeways in Alameda County in California between January 2007 and May 2013. The size and color of the balls indicate the number of distinct users in (a) and the delays (in minutes) in (b) associated with the corresponding geographical positions.

problem, where we are given signals that are defined on some graphs, and we would like to classify them using the proposed features; (ii) unlike the synthetic settings, in real world applications a "perfectly" meaningful graph is usually not directly observable. In this case, experiments in this setting tests the efficiency of the proposed spectral features for real world graphs that are only partially meaningful.

5.5.2 Mobility inference from Flickr data

For the classification experiments on the Flickr data, we select 400 out of 9629 vertices in the graph and manually label the graph signals corresponding to their neighborhoods with one of the mobility models, namely *Random*, *Spreading*, *Gathering*, and *Bi-directional*[7]. We have 100 training signals for each model. We then apply Algorithm 5 with $K = 3$ to classify the 3-hop neighborhoods of the rest of the vertices, using both ϕ_{spectral} and ϕ_{spatial} as features. The results are shown in Fig. 5.7(a) and Fig. 5.7(b), respectively. In both figures, the *dark blue* labels represent vertices that have less than ten 3-hop neighbors, so that their neighborhoods are not of significant interest. The *light blue*, *green*, *orange* and *red* labels represent vertices whose neighborhoods are classified as those that have the *Random*, *Spreading*, *Gathering*, and *Bi-directional* mobility patterns, respectively. As we can see in Fig. 5.7(a), the red vertices are mostly streets or bridges over the river, which corresponds well to the assumptions we made in the *Bi-directional* pattern. The orange vertices clearly represent landmarks or locations of significant interest, whose neighborhoods fit well in the *Gathering* pattern. The green vertices stand for regions similar to the ones in the *Spreading* pattern, where we expect movements of Flickr users from certain directions to the points of interest nearby. Therefore, they are usually located in the areas surrounding the orange vertices. Finally, the light blue vertices represent areas where the numbers of Flickr users are quite uniform, for example squares with no particularly interesting landmarks. These results show that the labels obtained by the classifier based on spectral features are indeed meaningful. In contrast, the labels obtained by the classifier based on spatial features are visually less accurate. As we can see in Fig. 5.7(b), not all streets are correctly identified, with some of them labeled with the *Random* pattern. Moreover, some landmarks and their surrounding areas are not indicated by the orange and green vertices, especially for areas around the Piccadilly Circus tube station.

The quality of the labels obtained in Fig. 5.7 can be evaluated from another perspective. Specifically, once the features are computed, we can apply clustering techniques to the features to divide the signals into different classes in an unsupervised fashion. Fig. 5.8 shows the class labels obtained by such an unsupervised approach using both ϕ_{spectral} and ϕ_{spatial} as features and the k-means algorithm [86] for clustering. The number of clusters is chosen to be 4 in the k-means implementation, which is the same as the number of expected mobility patterns. As can be seen in Fig. 5.8, there is a reasonably good match between the labels obtained by supervised classification and unsupervised clustering using the spectral features, while the labels obtained by these two approaches using the spatial features are quite different. This implies that the proposed spectral features defined in

[7]We manually label the training signals based on the similarity between their spectral representations and the spectral representations of the mobility models shown in Fig. 5.4. An alternative is to use the classifier trained in the synthetic experiments to select a set of training signals for each mobility model with maximum confidence. However, we acknowledge that the labelling process is guided by the same features as those that are used later in classification, as we do not resort to domain experts for labelling the training signals.

Table 5.4: Quantitative evaluations of the performances of the spectral (ϕ_{spectral}) and spatial features (ϕ_{spatial}) on the (a) Flickr dataset and (b) traffic dataset.

(a)

	Best Cross-Validation Accuracy	Mismatched Labels Between Classification and Clustering
ϕ_{spectral}	0.9425	36%
ϕ_{spatial}	0.5475	55%

(b)

	Best Cross-Validation Accuracy	Mismatched Labels Between Classification and Clustering
ϕ_{spectral}	1.0000	4%
ϕ_{spatial}	0.7750	58%

Eq. (5.5) are sufficiently discriminative to separate these graph signals even without the need for supervision.

Finally, we propose a quantitative performance evaluation of the two feature choices. Specifically, we report the best 5-fold cross-validation accuracy in the training stage, as well as the percentage of mismatched labels in the classification and clustering approaches. As we can see in Table 5.4(a), the proposed spectral features achieve significantly higher cross-validation accuracy than the spatial features, which indicates that the classification labels shown in Fig. 5.7(a) are more likely to be accurate than those in Fig. 5.7(b). Moreover, there are much less mismatches between the labels obtained by classification and clustering for the spectral features, which suggests that the proposed spectral features are more stable and reliable in classification. This is consistent with the visualizations in Fig. 5.7 and Fig. 5.8.

5.5.3 Mobility inference from traffic data

We now perform experiments with the traffic dataset. We select 40 out of 437 vertices in the graph and manually label the graph signals corresponding to their neighborhoods with one of the mobility models. We have 10 training signals for each model. We then apply Algorithm 5 with $K = 2$ to classify the 2-hop neighborhoods of the rest of the vertices. The classification results based on ϕ_{spectral} and ϕ_{spatial} are shown in Fig. 5.9(a) and Fig. 5.9(b)[8], respectively. As can be seen in Fig. 5.9(a), the crosses, passages, and two main junctions through which vehicles move from Hayward (middle left region) to Berkeley (top left region), are identified by the orange, red, and green vertices, respectively. This matches our intuitions about the possible locations for the *Gathering*, *Bi-directional*, and *Spreading* patterns. In comparison, the labels obtained with classification using spatial features and shown in Fig. 5.9(b) are less consistent.

Similarly to the first dataset, we apply unsupervised clustering using both ϕ_{spectral} and ϕ_{spatial} as features and the k-means algorithm. We see in Fig. 5.9 and Fig. 5.10 that there is again a good

[8]For a more clear illustration, the color codes on the edges in Fig. 5.9 and Fig. 5.10 are based on a different colormap from that in Fig. 1.2 and Fig. 5.6. The color codes on the edges in all these figures are nevertheless not important.

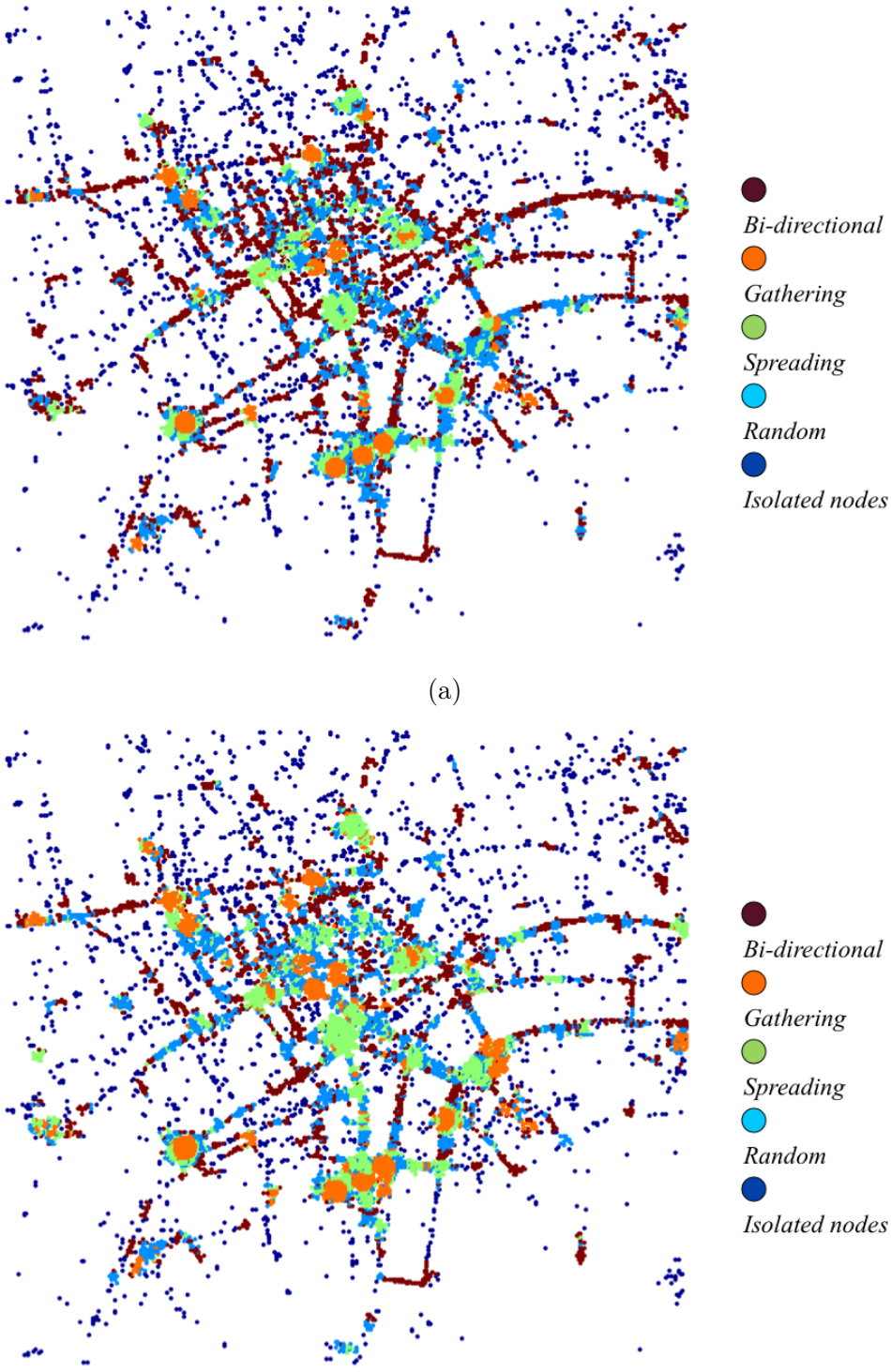


Figure 5.7: The class labels of mobility patterns for the central area of London, obtained by a classifier trained using the (a) spectral features ϕ_{spectral} and (b) spatial features ϕ_{spatial}.

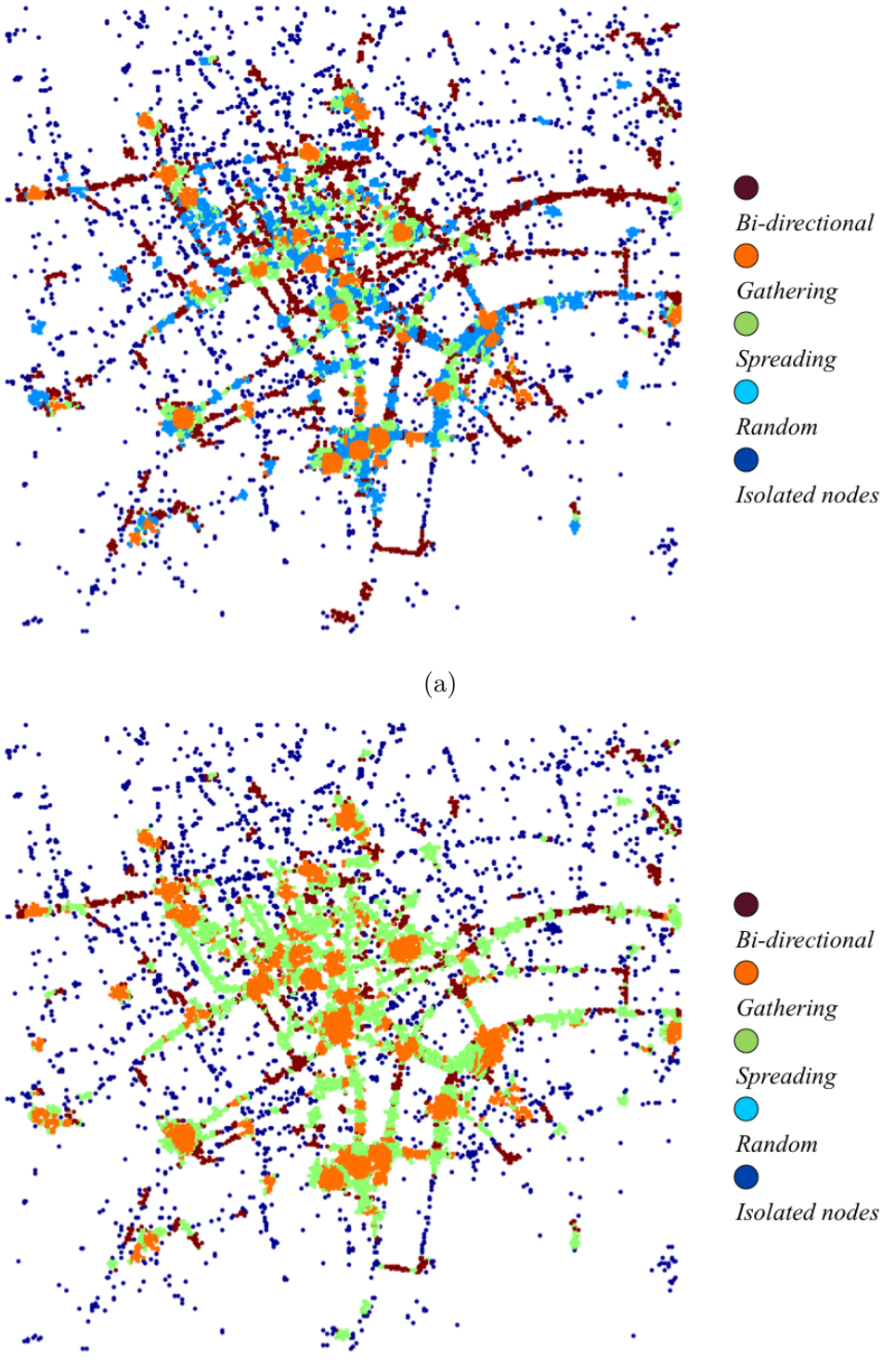
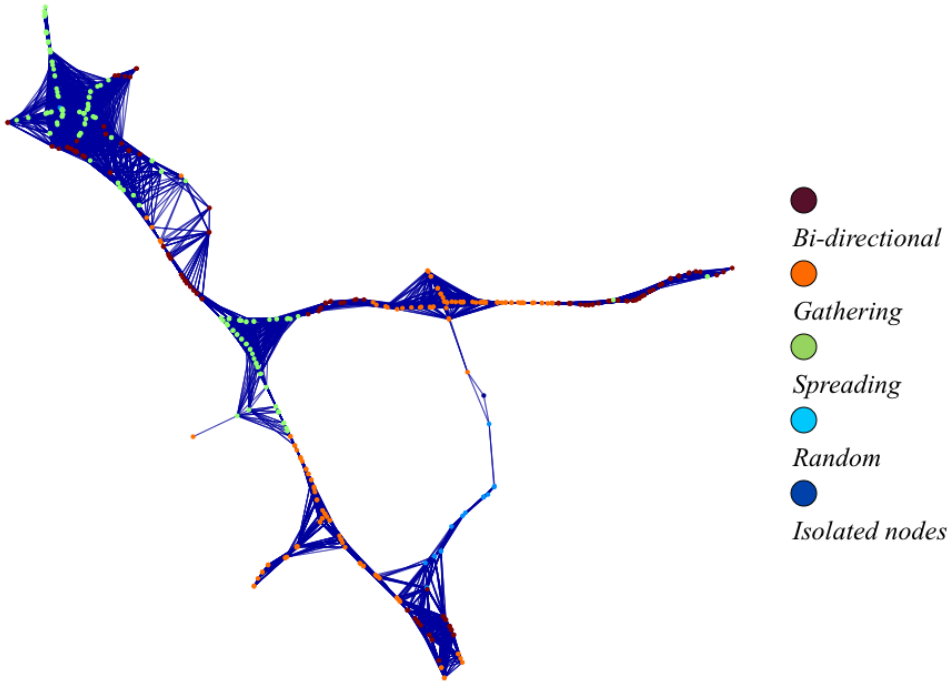
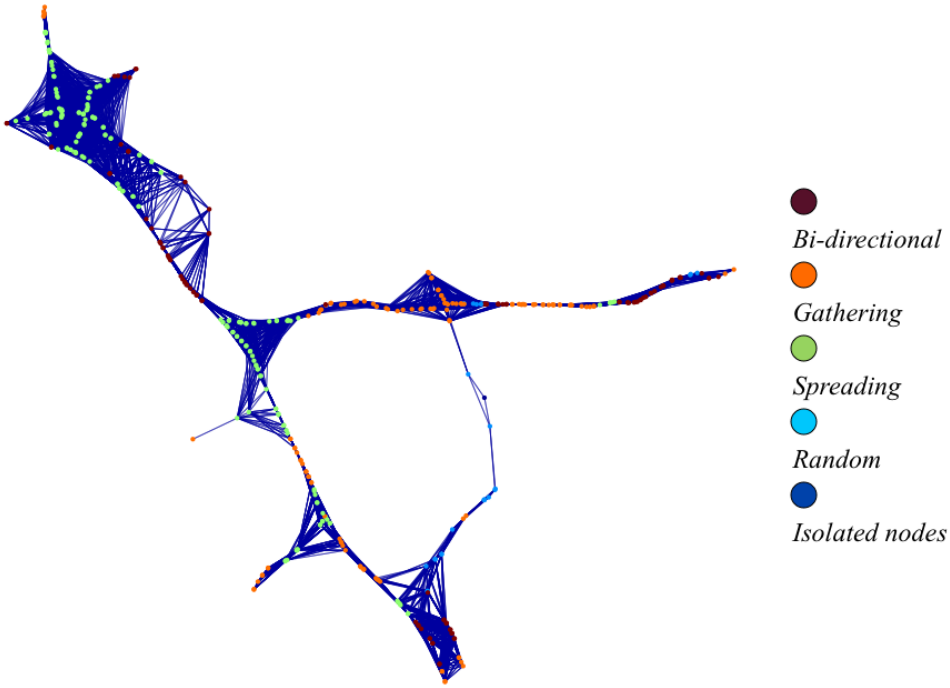


Figure 5.8: The class labels of mobility patterns for the central area of London, obtained by unsupervised clustering using the (a) spectral features ϕ_{spectral} and (b) spatial features ϕ_{spatial}.

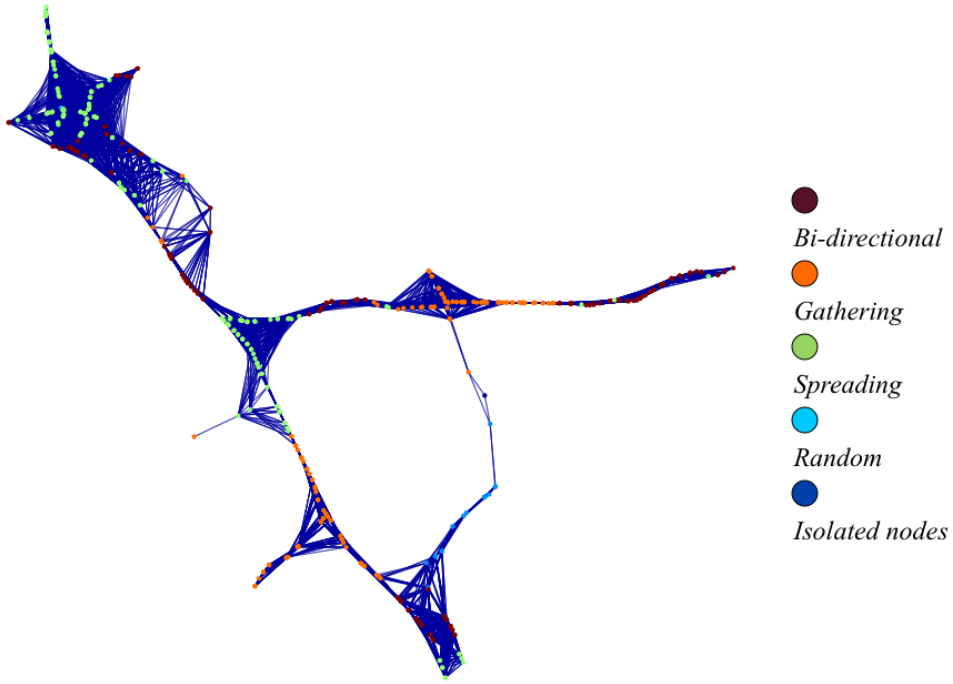


(a)

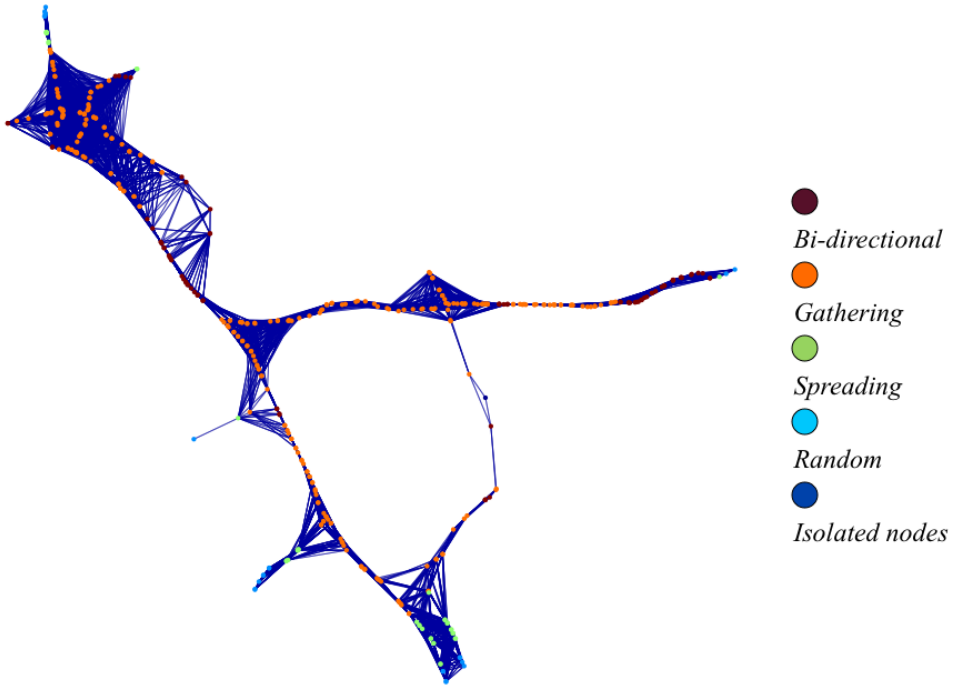


(b)

Figure 5.9: The class labels of mobility patterns for the freeways in Alameda County, obtained by a classifier trained using the (a) spectral features ϕ_{spectral} and (b) spatial features ϕ_{spatial}.



(a)



(b)

Figure 5.10: The class labels of mobility patterns for the freeways in Alameda County, obtained by unsupervised clustering using the (a) spectral features ϕ_{spectral} and (b) spatial features ϕ_{spatial}.

match between the labels obtained by classification and clustering using the spectral features, while this is not the case for the labels obtained using spatial features. In addition, Table 5.4(b) confirms the superiority of the proposed spectral features over the spatial features in separating the graph signals, in terms of cross-validation accuracy and feature consistency.

In summary, the experimental results on the two sets of real world data suggest that classifiers based on spatial features are not robust enough for signals on real world graphs. These can be much more irregular than the grid graphs, where spatial features perform relatively well in synthetic experiments. In the contrary, the spectral features do not seem to suffer from the same problem, since they better capture the intrinsic characteristics of the signals on graphs irrespective of the regularity of the graphs.

5.6 Conclusion

In this chapter, we have studied the classification problem of signals defined on graphs. We have shown that spectral graph models like wavelets defined on graphs are of great potential and lead to pretty robust features in learning tasks associated with signals with noticeable local variations, such as those that follow the random walk model. The present chapter opens new and interesting research questions in the field of signal processing on graphs, for example, (i) the development of learning algorithms for specific models of graph signals, and (ii) the design of graph transforms and dictionaries tailored for learning tasks such as classification and clustering.

We demonstrate the effectiveness of the proposed framework by a concrete classification application where we classify mobility patterns in real world traffic and social media data. It is interesting to see that our mobility inference framework relies only on coarse and aggregate data. Compared to the state-of-the-art approaches that are based on the analysis of empirical individual traces [52, 74, 84, 28, 142], our approach interestingly permits to the understanding of mobility behaviors without sacrificing privacy. Furthermore, the explicit mobility models proposed in this chapter presents an interesting comparison to the probabilistic models proposed in [34] for predicting different human mobility patterns.

It is important to notice that the choices of the graph always play a crucial role in graph-based signal processing and learning. Ideally, the graphs should contain information related to the true causes of the observations. However, in practice, the graphs are usually assumed to be known a priori; or, they come naturally from the domain of application, such as the geographical graphs that we considered in the real world experiments in this chapter. In order to maximize the benefits of the proposed learning frameworks on graphs, we therefore would like to learn a meaningful graph topology that well explains the observed data. This is the focus of the next chapter.

Chapter 6

Graph learning: A signal representation perspective

6.1 Introduction

In graph-based signal processing and learning tasks, the graph is often assumed to be known a priori, or given by the specific properties of the target application, like a geographical graph or a social friendship graph for example. In some applications, however, a natural graph is not easy to define at all. In these scenarios, we need to learn the graph topology from the observed data, such that it matches the prior knowledge on the data and captures the intrinsic relationships between the entities for efficient processing of the signals. This is the main objective of this chapter.

The key challenge in the problem of graph learning is to compute the topology such that the graph satisfies some models of the relationships between the signals and the graph topology. More precisely, we need some meaningful criteria to evaluate the matching between the two, which guide the process of graph learning given the observed signals. In this chapter, we propose to define such criteria by revisiting the representation learning theory [10] used in the traditional signal setting. Specifically, in a factor analysis model considered in the theory of representation learning, the Gaussian prior is a widely adopted probabilistic prior imposed on the latent variables that explain the observed signals. This leads to a Gaussian assumption on the observed signals; furthermore, the signal representation by the factor analysis model provides a probabilistic interpretation of the highly successful representation learned by the Principal Component Analysis (PCA) [10]. We generalize the Gaussian prior in the graph setting and impose it on the latent variables that control the observations (which are now graph signals). The only difference with the traditional case is that the transformation from the latent variables to the observed signals involves information about the topology of the graph. This permits to define joint properties (or joint model) between the signals and the graph, such that the signal representation is consistent with the Gaussian prior. Specifically, we show that this generalization leads to a PCA-like representation for the graph signals, which turns out to be a smooth signal representation on graphs. We then design an algorithm for graph learning that favors signal representations which are smooth and consistent with the statistical prior defined for the data.

We test our graph learning algorithm on several synthetic experiments, where we show that

the proposed algorithm is able to efficiently infer the topology of the groundtruth graphs, and outperforms a learning algorithm for the estimation of a sparse inverse covariance matrix for Gaussian graphical models [18]. We further demonstrate the meaningfulness of the proposed framework through a practical example, where we are able to infer the hidden altitude relationships between 89 measuring stations in Switzerland from the average monthly temperature observations collected at these stations between 1981 and 2010.

The graph learning framework proposed in this chapter provides new insight into the understanding of the interactions between signals and graphs in graph signal processing, and would surely lead to new theoretical development in this field. It will also benefit numerous emerging real world applications, such as the analysis of transportation, biomedical, and social networks, where it is beneficial to infer some hidden relationships between the entities.

The rest of the chapter is organized as follows. In Section 6.2, we formally define the problem of graph learning. In Section 6.3, we establish the link between signal representations in the traditional and graph settings when a Gaussian prior is imposed on the latent variables in a factor analysis model. We then propose our novel graph learning framework in Section 6.4 and present the experimental results in Section 6.5. Finally, we highlight some important connections between the proposed graph learning framework and the problems addressed in the previous chapters, and conclude the chapter in Section 6.6.

6.2 Problem formulation

The graph learning problem can be motivated from the following simple example. Consider a signal given by a set of unordered scalar values, which can potentially live on three different graphs G_1, G_2 and G_3, leading to three graph signals shown in Fig. 6.1. Without any assumption on the properties of the graph signal, the three candidate graphs are all valid choices. However, if we assume that the signal is smooth on the underlying graph, which loosely means that neighboring vertices share similar signal values, then G_1 is obviously a more reasonable choice than G_2 and G_3. Our objective is exactly to learn a graph similar to G_1 when the signal is expected to be smooth on an "unknown" graph topology.

The above example shows that learning a meaningful graph by enforcing desired signal properties on the observed signals, such as smoothness, is surely beneficial for the appropriate representation of the data. This is exactly the objective of our learning framework. Specifically, we consider in this chapter the following problem:

Graph learning for signal representation. *Given a set of observed data $X = \{x_i\}_{i=1}^p$ ($x_i \in \mathbb{R}^n$) that are considered as signals defined on a weighted and undirected graph G of n vertices, we would like to infer an optimal topology of G, namely, its edges and the associated weights, that results in given statistical properties of X, such as smoothness of the graph signals.*

We remark here that our objective is slightly different from the objective of frameworks for learning graphical models proposed in machine learning, where the learned graph usually represents the conditional independence structure between the random variables. We will provide further discussion about this later in the chapter.

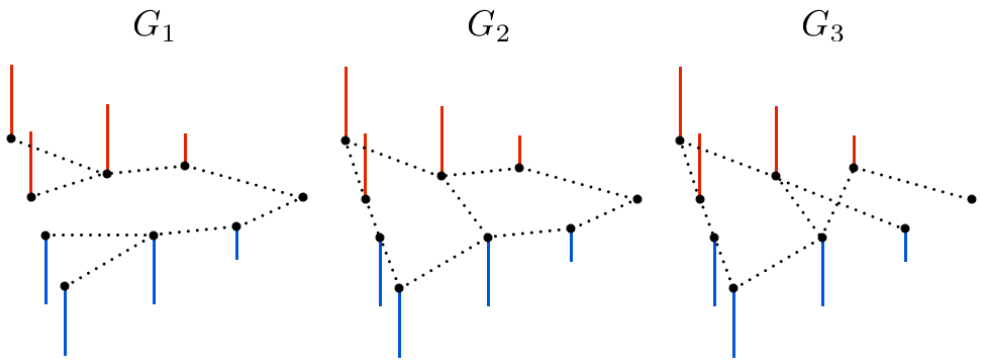


Figure 6.1: The same signal can potentially live on different graphs (this figure is inspired by the one in [119]). In these figures, the red bars pointing upwards and the blue bars pointing downwards represent positive and negative signal values, respectively. The length of the bars reflects the magnitude of the values. In this illustration, while all the choices are valid a priori, we are looking for the graph that offers desirable properties of the resulting graph signal. Specifically, in this example, choosing graph G_1 favors the smoothness property of the graph signal.

6.3 Factor analysis framework

6.3.1 Signal representation in the traditional setting

We present our learning framework by making the connections between the signal representations in the traditional setting and the graph setting. We start with signal representations in the traditional setting, and then generalize it to the graph setting. In our analysis, we consider the factor analysis [8, 9] model as our signal model, which is a generic linear statistical model that tries to explain observations of a given dimension with a potentially smaller number of unobserved latent variables. Such latent variables with given probabilistic prior lead to efficient signal representations in both the traditional and graph settings. Specifically, we consider:

$$x = Wh + u_x + \epsilon, \tag{6.1}$$

where $x \in \mathbb{R}^n$ is the observation, $h \in \mathbb{R}^k$ is the latent variable that controls x, $W \in \mathbb{R}^{n \times k}$ is the full rank representation matrix that linearly relates the two random variables. The parameter $u_x \in \mathbb{R}^n$ is the mean of x, and ϵ is a multivariate Gaussian noise that has mean zero and covariance $\sigma_\epsilon^2 I_n$. Its probability density function is given by:

$$p(\epsilon) \sim \mathcal{N}(0, \sigma_\epsilon^2 I_n), \tag{6.2}$$

where I_n represents the identity matrix of dimension n. It is usually assumed that $k < n$, so that the latent variable h provides a more parsimonious signal representation than x. Notice also that we do not require the columns of W to be orthonormal. The conditional probability of x given h can then be written as:

$$p(x|h) \sim \mathcal{N}(Wh + u_x, \sigma_\epsilon^2 I_n). \tag{6.3}$$

In the traditional setting, a Gaussian prior is conventionally imposed on the latent variable h in Eq. (6.3), which leads to an efficient representation of the observation. Specifically, assume that h follows a zero-mean multivariate Gaussian distribution with covariance matrix $\sigma_h^2 I_n$:

$$p(h) \sim \mathcal{N}(0, \sigma_h^2 I_n). \tag{6.4}$$

Due to the properties of multivariate Gaussian distributions under linear transformations, x also follows a multivariate Gaussian distribution:

$$p(x) \sim \mathcal{N}(u_x, \sigma_h^2 W W^T + \sigma_\epsilon^2 I_n). \tag{6.5}$$

Eq. (6.5) shows that the representation matrix W spans the same subspace as the k leading principal components of the covariance matrix of x. To see this more clearly, consider the following eigen-decomposition:

$$\sigma_h^2 W W^T + \sigma_\epsilon^2 I_n = U \Lambda U^T, \tag{6.6}$$

where U and Λ are the eigenvector and eigenvalue matrices of the covariance matrix. We can rewrite Eq. (6.6) as follows:

$$\sigma_h^2 W W^T = U \Lambda U^T - \sigma_\epsilon^2 I_n = U(\Lambda - \sigma_\epsilon^2 I_n) U^T. \tag{6.7}$$

Since W has rank k, Eq. (6.7) implies that the diagonal matrix $(\Lambda - \sigma_\epsilon^2 I_n)$ has only its first k diagonal entries that are non-zero; their value is σ_h^2. Therefore, Eq. (6.7) can be simplified as:

$$W W^T = U_k I_k U_k^T, \tag{6.8}$$

where $U_k \in \mathbb{R}^{n \times k}$ contains the k leading principal components as columns. As a result, there must exist an orthogonal matrix $R \in \mathbb{R}^{k \times k}$ that satisfies

$$W = U_k R. \tag{6.9}$$

Eq. (6.9) implies that W spans the same subspace as U_k. It has been pointed out in [10] that signal representation by the factor analysis model shown above provides a probabilistic interpretation of the highly successful representation learned by the PCA, which was originally presented in [105, 130]. The key observation here is that, under a Gaussian prior distribution of the latent variable h, the representation matrix W in the factor analysis model spans the same subspace as the k leading principal components of the covariance matrix of x. We will see in the following section that this observation has an analogy in the graph setting; we will use this property in our graph learning framework.

Given the observation x and the Gaussian prior distribution in Eq. (6.4), one is often interested in a maximum a posteriori (MAP) estimate of h that explains the observations x, which is of key importance in our learning framework. In the following section, we will show an analogy of such an estimate in the graph setting, based on which we derive our graph learning framework.

6.3.2 Signal representation in the graph setting

We now consider signal representation in the graph setting, where the signals are defined on the vertex set of the graph. In this setting, the Laplacian matrix L of a graph G provides tools to establish connections between signal representations in the traditional and graph settings. Specifically, since L is a real and symmetric matrix, it has a complete set of orthonormal eigenvectors and associated eigenvalues:

$$L = \chi \Lambda \chi^T, \tag{6.10}$$

where χ is the eigenvector matrix, and Λ is the diagonal eigenvalue matrix where the eigenvalues are sorted in increasing order. We assume that the graph G is connected, which implies that the smallest eigenvalue is 0 and that all the other eigenvalues are strictly positive.

In the graph setting, for the representation of graph signals, we would like to generalize the factor analysis model in Eq. (6.1). The key is therefore to choose a representation matrix W in Eq. (6.1) that relates the graph signals with the latent variable. In our analysis, we propose to define the representation matrix as the eigenvector matrix χ of the graph Laplacian L. The motivation is that, under such a definition, a Gaussian prior imposed on the latent variable leads to a Gaussian assumption and smooth representations of the graph signals, as we will see later. One may have also noticed that, in Eq. (6.1), the representation matrix W is a rank-k tall and thin matrix, while in the graph setting we choose χ which is a rank-n square matrix. The reason behind is that we would like to base our studies on all the eigenvectors of the graph Laplacian L, not just the k leading ones.

Specifically, let us consider the following model:

$$x = \chi h + u_x + \epsilon, \tag{6.11}$$

where $x \in \mathbb{R}^n$ represents the observed graph signal, $h \in \mathbb{R}^n$ represents the latent variable that controls the graph signal x through the representation matrix χ, $u_x \in \mathbb{R}^n$ is the mean of x, and ϵ is a multivariate Gaussian noise with mean zero and covariance $\sigma_\epsilon^2 I_n$. The probability density function of ϵ is thus given by:

$$p(\epsilon) \sim \mathcal{N}(0, \sigma_\epsilon^2 I_n). \tag{6.12}$$

Similarly to the traditional setting, we impose a Gaussian prior on the latent variable h. Specifically, we assume that the latent variable h follows a degenerate zero-mean multivariate Gaussian distribution with covariance matrix $\Lambda^\dagger$:

$$p(h) \sim \mathcal{N}(0, \Lambda^\dagger), \tag{6.13}$$

where $\Lambda^\dagger$ denotes the Moore-Penrose pseudoinverse of Λ since it is not full rank. In particular, the covariance matrix defined in Eq. (6.13) leads to a smooth signal representation that we will see later. The conditional probability of x given h, and the probability of x, are thus given as follows:

$$p(x|h) \sim \mathcal{N}(\chi h + u_x, \sigma_\epsilon^2 I_n), \tag{6.14}$$

$$p(x) \sim \mathcal{N}(u_x, L^\dagger + \sigma_\epsilon^2 I_n), \tag{6.15}$$

where in Eq. (6.15) we have used the fact that, the pseudoinverse of L, $L^\dagger$, admits the following

eigen-decomposition:

$$L^\dagger = \chi \Lambda^\dagger \chi^T. \tag{6.16}$$

It can be seen from Eq. (6.15) that, in a noise-free scenario where $\sigma_\epsilon = 0$, x also follows a degenerate multivariate Gaussian distribution with zero-mean and covariance $L^\dagger$. In this case, x can be interpreted as a Gaussian Markov Random Field (GMRF) with respect to the graph G, where the inverse covariance matrix, or the so-called precision matrix, is chosen to be the graph Laplacian L. Notice that the GMRF is a very generic model such that the precision matrix can be defined with much freedom, as long as its non-zero entries encode the partial correlations between random variables and as long as their locations correspond to the edges in the graph [106]. However, the graph Laplacian L is widely adopted in the literature of image analysis, as the precision matrix of the GMRFs that model images [139]. From Eq. (6.15) we see that, by defining the representation matrix in the factor analysis model as the eigenvector matrix χ and assuming that the latent variable follows a degenerate Gaussian distribution with covariance $\Lambda^\dagger$, we can recover the GMRF with precision matrix L in a noise-free scenario.

In the presence of noise, we see from Eq. (6.15) that, under a Gaussian prior on the latent variable h, the representation matrix χ is the eigenvector matrix of the covariance of x:

$$L^\dagger + \sigma_\epsilon^2 I_n = \chi(\Lambda^\dagger + \sigma_\epsilon^2 I_n)\chi^T. \tag{6.17}$$

This is analogous to the observation we have in the traditional setting, where the representation matrix W spans the same subspace as the k leading principal components of the covariance of x under a Gaussian prior on h. The representation in Eq. (6.11) can then be considered as a PCA-like representation for the graph signal x. More importantly, it leads to a smooth assumption of the signal on the graph, as we show as follows.

As we have seen in Eq. (6.11), the latent variables h explains the graph signal x through the representation matrix, namely, the eigenvector matrix χ of the graph Laplacian. Given the observation x and the multivariate Gaussian prior distribution of h in Eq. (6.13), we are thus interested in a MAP estimate of h. Specifically, by applying Bayes' rule and assuming without loss of generality that $u_x = 0$, the MAP estimate of the latent variable h can be written as follows [58]:

$$\begin{aligned} h_{\text{MAP}}(x) &:= \arg\max_h p(h|x) \\ &= \arg\max_h p(x|h)p(h) \\ &= \arg\min_h \left(-\log\, p_E(x - \chi h) - \log\, p_H(h)\right). \end{aligned} \tag{6.18}$$

Given the probability distributions shown in Eq. (6.12) and Eq. (6.13), the above MAP estimate can then be expressed as:

$$h_{\text{MAP}}(x) = \arg\min_h ||x - \chi h||_2^2 + \alpha\ h^T \Lambda h, \tag{6.19}$$

where α is some constant parameter. In a noise-free scenario where $x = \chi h$, Eq. (6.19) corresponds to minimizing the following quantity:

$$h^T \Lambda h = (\chi^T x)^T \Lambda \chi^T x = x^T \chi \Lambda \chi^T x = x^T L x. \tag{6.20}$$

The Laplacian quadratic term in Eq. (6.20) is usually considered as a measure of smoothness of the signal x on G [144]. Therefore, we see that in a factor analysis model in Eq. (6.11), a Gaussian prior in Eq. (6.13) imposed on the latent variable h leads to a smooth assumption of the graph signal. We are going to make use of this observation in our graph learning algorithm in the following section.

6.4 Learning graphs under signal smoothness prior

Given a Gaussian prior and the smooth assumption of the graph signal as we have shown above, the MAP estimate of h in Eq. (6.19) provide us with the connections between the signal observations and the desired graph topology to be learned. Specifically, notice in Eq. (6.19) that both the representation matrix χ and the inverse covariance matrix Λ of the Gaussian prior distribution imposed on h come from the graph Laplacian L. When the graph is unknown, we can have the following joint optimization problem of χ, Λ and h:

$$\arg\min_{\chi,\Lambda,h} ||x - \chi h||_2^2 + \alpha\ h^T \Lambda h, \tag{6.21}$$

where χ and Λ are the eigenvector and eigenvalue matrices of L, respectively. Eq. (6.21) can be simplified via a change of variable $y = \chi h$ as:

$$\arg\min_{L,y} ||x - y||_2^2 + \alpha\ y^T L y. \tag{6.22}$$

According to the factor analysis model in Eq. (6.11), y can be considered as a "noiseless" version of the zero-mean observation x. Furthermore, as we have explained before, due to the properties of the graph Laplacian L, the quadratic form $y^T L y$ in Eq. (6.22) is usually considered as a measure of smoothness of the signal y on G. In the problem of Eq. (6.22), we are thus interested in finding jointly L and y, such that y is close to the observation x, and at the same time y is smooth on the graph G. Therefore, the learned graph topology enforces the smoothness property of the signals, defined by the Laplacian quadratic smoothness measure.

We propose to solve the optimization problem of Eq. (6.22), with constraints and an additional penalty term on the Frobenius norm of L in the objective, in the following matrix form:

$$\begin{aligned} \arg\min_{L,Y}\ & ||X - Y||_F^2 + \alpha\ tr(Y^T L Y) + \beta ||L||_F^2, \\ \text{s.t.}\quad & tr(L) = n, \\ & L_{ij} = L_{ji} \leq 0,\ i \neq j, \\ & L \cdot \mathbf{1} = \mathbf{0}, \end{aligned} \tag{6.23}$$

where $X \in \mathbb{R}^{n \times p}$ contains the p input data samples $\{x_i\}_{i=1}^p$ as columns, α and β are two regularization parameters, and $\mathbf{1}$ and $\mathbf{0}$ denote the constant one and zero vector. The Frobenius norm of L is added to improve the numerical stability of the solution and, together with the Laplacian quadratic term, will also influence the sparsity of the learned graph (as we will see in the experimental section)[1]. The first constraint (the trace constraint) in Eq. (6.23) helps avoiding trivial solutions, and

[1] We do not impose a L^1-norm penalty on L since we do not specifically assume that the learned graph is sparse.

the second and third constraints guarantee that the learned L is a valid Laplacian matrix.

The optimization problem of Eq. (6.23) is not jointly convex in L and Y. Therefore, we adopt an alternating optimization scheme where at each step we fix one variable and solve the other. Specifically, at the first step, for a given Y, we solve the following optimization problem with respect to L:

$$\begin{aligned} \arg\min_{L} \; & \alpha \; tr(Y^T L Y) + \beta ||L||_F^2, \\ \text{s.t.} \quad & tr(L) = n, \\ & L_{ij} = L_{ji} \leq 0, \; i \neq j, \\ & L \cdot \mathbf{1} = \mathbf{0}. \end{aligned} \tag{6.24}$$

At the second step, L is fixed and we solve the following optimization problem with respect to Y:

$$\arg\min_{Y} \; ||X - Y||_F^2 + \alpha \; tr(Y^T L Y). \tag{6.25}$$

Both of the problems of Eq. (6.24) and Eq. (6.25) can be casted as convex optimization problems with unique solutions. Specifically, the problem of Eq. (6.24) can be written as a quadratic program, which can be efficiently solved via interior point methods [17] or operator splitting methods (e.g., alternating direction method of multipliers (ADMM) [18]). In addition, the problem of Eq. (6.25) has the following closed-form solution:

$$Y = (I_n + \alpha L)^{-1} X. \tag{6.26}$$

In practice, we alternate between these two steps to get the final solution to the problem of Eq. (6.23), and we observe empirical convergence within a few iterations. The complete algorithm is summarized in Algorithm 6.

Algorithm 6 Graph Learning for Smooth Signal Representation (**GL-SigRep**)

1: **Input:** Input signal X, number of iterations $iter$, α, β
2: **Output:** Output signal Y, graph Laplacian L
3: **Initialization:** $Y = X$
4: **for** $t = 1, 2, ..., iter$ **do:**
5: **Step to update Graph Laplacian L:**
6: Solve the optimization problem of Eq. (6.24) to update L.
7: **Step to update Y:**
8: Solve the optimization problem of Eq. (6.25) to update Y.
9: **end for**
10: $L = L^{iter}, Y = Y^{iter}$.

We note that the proposed learning framework has some similarity with the one in [64], where the authors have proposed a similar objective as the one in Eq. (6.23), based on a smoothness or fitness metric of the signals on graphs. However, we rather take a probabilistic approach that is analogous to the one in the traditional setting, which gives us an extra data fitting term $||X - Y||_F^2$

in the objective of the optimization problem of Eq. (6.23). In practice, when the power of Laplacian in [64] is chosen to be 1, it corresponds to assuming $X = Y$ and finding the solution to a single instance of the problem of Eq. (6.24).

Finally, we compare the proposed graph learning framework to a state-of-the-art approach for estimating a sparse inverse covariance matrix for GMRFs. Specifically, the works in [6, 49] propose to solve the following L^1-regularized log-determinant program:

$$\arg\min_{L_{\text{pre}}} \; tr(SL_{\text{pre}}) - \log\,\det(L_{\text{pre}}) + \lambda||L_{\text{pre}}||_1, \tag{6.27}$$

where L_{pre} is the precision (inverse covariance) matrix to estimate, $S = XX^T$ is the sample covariance matrix, λ is a regularization parameter, $\det(\cdot)$ denotes the determinant, and $||\cdot||_1$ denote the L^1-norm. The main motivation behind the optimization of Eq. (6.27) is as follows. When the number of samples is smaller than the sample dimension, the sample covariance S is not full rank. In this case, the optimization of Eq. (6.27) tries to find a full rank precision matrix L_{pre} (this is enforced by the log-determinant term), which is the closest to the sample covariance S (this is enforced by the trace term). Additionally, the L^1 penalty is introduced to enforce the sparsity of L_{pre} in terms of the L^1-norm. The problem of Eq. (6.27) is conceptually similar to the problem of Eq. (6.23), in the sense that both can be interpreted as estimating the precision matrix of a GMRF. An important difference is however that the precision matrix in our framework is a valid graph Laplacian, while the one in Eq. (6.27) is not. Indeed, L_{pre} is a full rank matrix that usually have both positive and negative off-diagonal entries, reflecting both positive and negative correlations. Therefore, L_{pre} cannot be interpreted as a graph topology for defining graph signals; it rather reflects the partial correlations between the random variables that control the observations. This major difference leads to the different performances of the two approaches in our experiments, as we will see in the following section.

6.5 Experimental results

In this section, we evaluate the performance of the proposed graph learning algorithm. We first describe the general experimental setting and the implementation details for the algorithms under comparison, and then we present the experimental results on synthetic and real world data.

6.5.1 Experimental settings

Given a set of synthetic or real world observations, we test the performance of Algorithm 6 by comparing the learned graph to the groundtruth one. Specifically, we compare the edges in the learned graph to the ones of the groundtruth graph. In the quantitative evaluation, we focus on the presence or absence of the edges and do not consider the weights of the edges. We use four evaluation criteria commonly used in information retrieval [88] to test the performance of our algorithm:

- *Precision*: *Precision* is defined as:

$$Precision = \frac{TP}{TP + FP}, \tag{6.28}$$

where TP and FP are the numbers of true positives and false positives. It thus evaluates the percentage of correct edges in the learned graph, that is the edges that are present in the groundtruth graph.

- *Recall*: *Recall* is defined as:
$$Recall = \frac{TP}{TP + FN}, \tag{6.29}$$
where FN is the number of false negatives. It thus evaluates the percentage of the edges in the groundtruth graph that are present in the learned graph.

- *F-measure*: *F-measure* is defined as the harmonic mean of *Precision* and *Recall*:
$$F\text{-}measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}. \tag{6.30}$$
The *F-measure* thus takes into account both *Precision* and *Recall* to measure the accuracy of the obtained results.

- *Normalized Mutual Information* (*NMI*): *NMI* is defined as in Eq. (3.22). The *NMI* measures the mutual dependence between the obtained results and the groundtruth.

In our experiments, we solve the optimization of Eq. (6.24) using the convex optimization package CVX [55, 54]. The algorithm stops when the maximum number of iterations is reached or the absolute change in the objective is smaller than 10^{-4}. In most of the cases, we observed that the algorithm converges within less than five iterations. The experiments are carried out on different sets of parameters, namely, for different values of α and β in Eq. (6.23). Finally, we prune insignificant edges with a weight less than 10^{-4} in the learned graph.

We compare the proposed graph learning algorithm to an algorithm for estimating a sparse inverse covariance matrix for Gaussian graphical models, solving the L^1-regularized log-determinant program of Eq. (6.27) with the ADMM [18]. We denote this algorithm as **GL-LogDet**. This method however aims at estimating a full rank precision matrix which is not a valid graph Laplacian. Even if the comparison with **GL-LogDet** is not completely fair, it is nevertheless interesting since the learned precision matrix also encodes the information about the partial correlations between the variables. In our experiments, following the suggestions in [18], we set the augmented Lagrangian parameter and the over-relaxation parameter in **GL-LogDet** to be 1 and 1.5, respectively. We test **GL-LogDet** based on different choices of the parameter λ in Eq. (6.27). In the evaluation, all the off-diagonal non-zero entries whose absolute values are above the threshold of 10^{-4} are considered as valid correlations[2]. These correlations are then considered as learned "edges" and compared against the edges in the groundtruth graph for performance evaluation.

6.5.2 Results on synthetic data

We first carry out our synthetic experiments on three different graphs of 20 vertices, namely, a graph whose edges are determined based on Euclidean distances between vertices, and two graphs

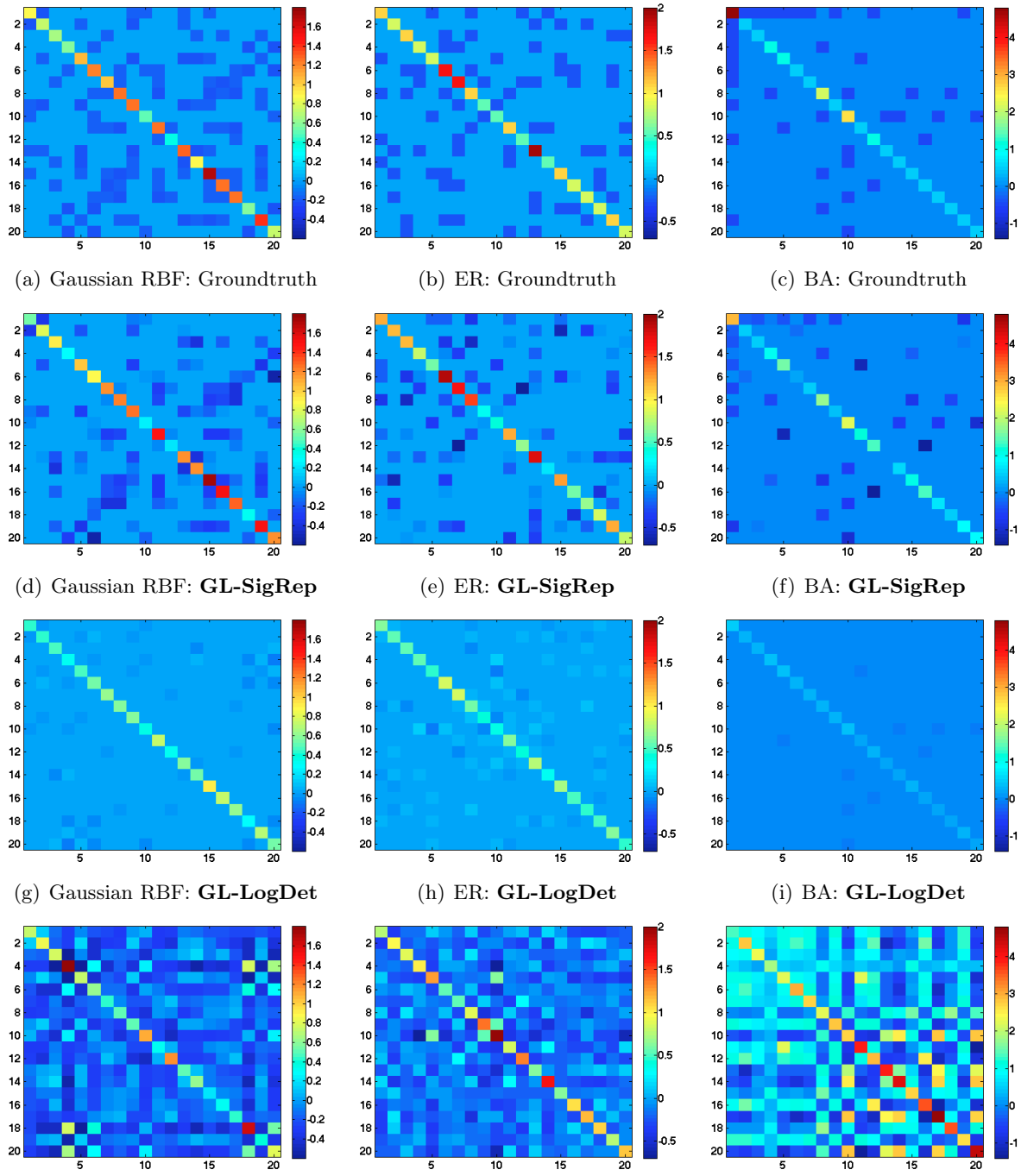
[2]Since the precision matrix learned by **GL-LogDet** seldom contains entries with values smaller than 10^{-4}, the thresholding does not make a difference in the evaluation. Nevertheless, we still implement this step for a fair comparison.

that follow the Erdős-Rényi model [45] and the Barabási-Albert model [7], respectively. For the first graph, we generate the coordinates of the vertices uniformly at random in the unit square, and compute the edge weights between every pair of vertices using the Euclidean distances between them and a Gaussian radial basis function (RBF): $\exp\left(-d(i,j)^2/2\sigma^2\right)$, with the width parameter $\sigma = 0.5$. We then remove all the edges whose weights are smaller than 0.75. Next, we use the Erdős-Rényi (ER) model with edge probability 0.2 to generate a random graph, that is, each possible edge is included in the graph with probability 0.2 independent form every other edge. Finally, we use the Barabási-Albert (BA) model to generate a scale-free random graph. Specifically, the BA graph in our experiments is generated by adding one new vertex to the graph at each time, connecting to one existing vertex in the graph. The probability of the new vertex attaching to a given existing vertex in the graph is proportional to the ratio of the degree of that existing vertex to the sum of degrees of all the existing vertices. The BA and ER graphs are important random graph models studied in network science. Specifically, the former have power-law (or scale-free) degree distributions similarly to many networks observed in real world, while the latter do not. The BA and ER graphs in our experiments have unitary edge weights. Given a synthetic groundtruth graph, we compute the graph Laplacian L and normalize the trace according to Eq. (6.23). Then, for each graph, we generate 100 signals $X = \{x_i\}_{i=1}^{100}$ that follow the distribution shown in Eq. (6.15) with $u_x = 0$ and $\sigma_\epsilon = 0.5$. We then apply **GL-SigRep** and **GL-LogDet** to learn the graph Laplacian or the precision matrix, respectively, given only the signals X.

We first show visually the learned graph Laplacians or precision matrices for the algorithms **GL-SigRep** and **GL-LogDet**. In Fig. 6.2, we show from the top to the bottom rows the Laplacian matrices of the groundtruth graph, the graph Laplacians learned by **GL-SigRep**, the precision matrices learned by **GL-LogDet**, and the sample covariance matrices $S = XX^T$, for one random instance of each of the three graph models[3]. First, we see that, although the positions of the large entries in the sample covariance matrices partially correspond to the positions of the edges in the groundtruth graph, they generally contain many noisy correlations. Next, for all the three types of graphs, the graph Laplacian matrices learned by **GL-SigRep** is visually more consistent with the groundtruth data than the precision matrices learned by **GL-LogDet**. In addition to the position of the edges, the edge weights in the Laplacians learned by **GL-SigRep** is similar to the ones in the groundtruth graph, while the correlations in the precision matrices learned by **GL-LogDet** are relatively small (this is especially the case for the BA graph). Possible explanations of this observation are that: (i) it estimates a full rank precision matrix instead of the groundtruth graph Laplacian, and (ii) the L^1-norm penalty in the optimization of Eq. (6.27) leads to many relatively small correlations.

Next, we evaluate quantitatively the performances of the two graph learning algorithms in recovering the positions of the edges in the groundtruth. In Table 6.1, we show the best *F-measure*, *Precision*, *Recall* and *NMI* scores achieved by the two algorithms averaged over ten random instances of the three graphs with the associated signals X. Our algorithm **GL-SigRep** clearly outperforms **GL-LogDet** in terms of all the evaluation criteria. Especially, for the Gaussian RBF and BA graphs, **GL-SigRep** achieves an average *F-measure* score close to 0.9, which means that

[3]These results are obtained based on the parameters that lead to a similar number of edges as the ones in the groundtruth graph. More discussion about the choices of these parameters are presented later. The values of the sample covariance matrices are scaled before the visualization.



(a) Gaussian RBF: Groundtruth (b) ER: Groundtruth (c) BA: Groundtruth

(d) Gaussian RBF: **GL-SigRep** (e) ER: **GL-SigRep** (f) BA: **GL-SigRep**

(g) Gaussian RBF: **GL-LogDet** (h) ER: **GL-LogDet** (i) BA: **GL-LogDet**

(j) Gaussian RBF: Sample covariance (k) ER: Sample covariance (l) BA: Sample covariance

Figure 6.2: The learned graph Laplacian or precision matrices. From the top to the bottom rows are the groundtruth Laplacians, the Laplacians learned by **GL-SigRep**, the precision matrices learned by **GL-LogDet**, and the sample covariances. From the left to the right columns we show the Gaussian RBF graph, the ER graph, and the BA graph.

Table 6.1: Graph learning performance for **GL-SigRep** and **GL-LogDet**: (a) the Gaussian RBF graph, (b) the ER graph, and (c) the BA graph.

(a) Gaussian RBF graph

	F-measure	Precision	Recall	NMI
GL-SigRep	0.8803	0.8535	0.9108	0.5902
GL-LogDet	0.4379	0.2918	0.8851	0.0220

(b) ER graph

	F-measure	Precision	Recall	NMI
GL-SigRep	0.7495	0.7185	0.7901	0.4126
GL-LogDet	0.3181	0.2235	0.5587	0.0109

(c) BA graph

	F-measure	Precision	Recall	NMI
GL-SigRep	0.8898	0.9398	0.8474	0.7345
GL-LogDet	0.3277	0.2657	0.5000	0.0665

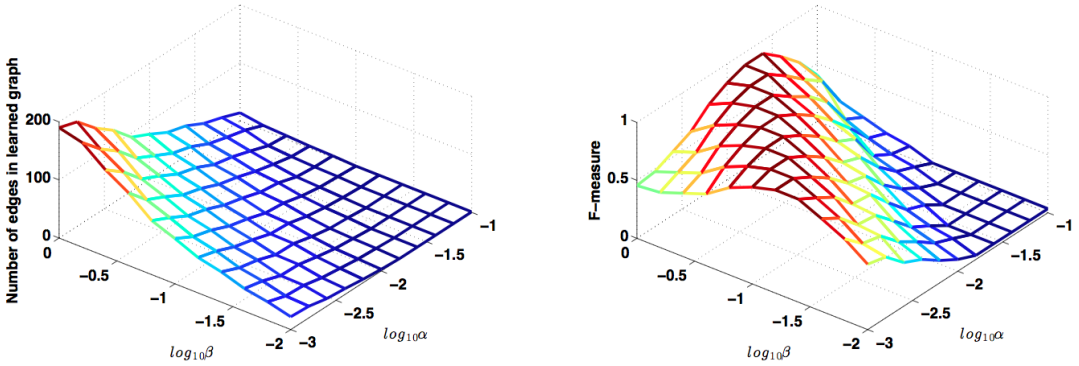
the learned graphs have very similar topologies compared to the groundtruth.

To better understand the behavior of **GL-SigRep** under different sets of parameters, we plot in Fig. 6.3 the numbers of edges in the learned graph, and the *F-measure* scores, under 121 different combinations of the parameters α and β in Eq. (6.23), for a random instance of the Gaussian RBF graph. First, we see that the number of edges in the learned graph decreases as β decreases and α increases. The intuitions behind this behavior are as follows. When β increases, the Frobenius norm of L in the objective function in Eq. (6.23) tends to be small, leading to a more uniform distribution of the entries, thus the number of edges tends to increase. Decreasing β leads to the opposite effect. When α increases, the trace of the quadratic term tends to be small. In this case, since we have:

$$tr(Y^T LY) = -\frac{1}{2}\sum_{m=1}^{p}\sum_{i,j}^{n} L_{ij}\left(y_m(i) - y_m(j)\right)^2, \tag{6.31}$$

the algorithm favors a smaller number of non-zero entries in L, and the number of edges decreases. Therefore, both parameters α and β implicitly affect the sparsity of the learned graph Laplacian. More interestingly, Fig. 6.3(a) and Fig. 6.3(b) show that both the number of edges and the *F-measure* scores are similar for the values of α and β with the same ratio $\frac{\beta}{\alpha}$. This suggests that the trace of the quadratic term and the Frobenius norm are the dominating factors in the optimization of Eq. (6.23), rather than the data fidelity term. This implies that, in practice, we may search for an appropriate ratio $\frac{\beta}{\alpha}$ to maximize the learning performance of the algorithm.

Next, we show in Fig. 6.4(a) and Fig. 6.5(a) the number of edges in the graphs learned by **GL-SigRep** and the learning performance evaluated based on the four criteria, respectively, for

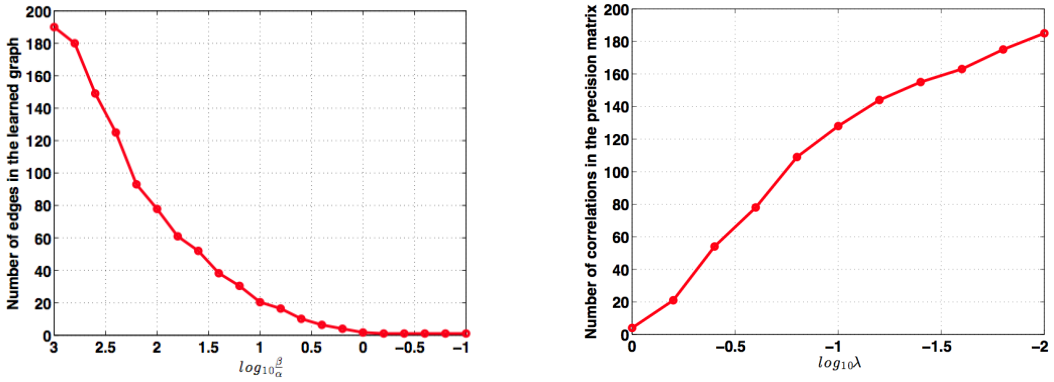


(a) (b)

Figure 6.3: (a) The number of edges in the learned graph, and (b) the *F-measure* score, under 121 different combinations of the parameters α and β for an instance of the Gaussian RBF graph.

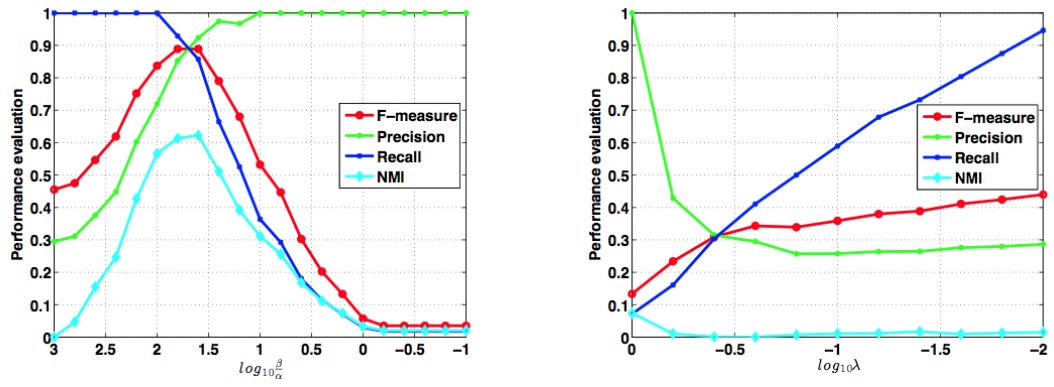
the same Gaussian RBF graph as before under different ratios of β to α. As expected, the number of edges decreases as the ratio of β to α decreases. Looking at Fig. 6.4(a) and Fig. 6.5(a) together, we see that, as the number of edges approaches the number of edges in the groundtruth graph (in this case, 56 edges), the *Recall* stays high and the *Precision* increases rapidly, which makes the *F-measure* increase. When the number of edges in the learned graph is close to the one in the groundtruth graph, the curves for the *Precision* and the *Recall* intersect and the *F-measure* reaches its peak. After that, although the *Precision* keeps increasing towards 1, the *Recall* drops rapidly as fewer and fewer edges are detected, leading to a decreasing trend in the *F-measure*. A similar trend can be observed in the curve for the *NMI* score. These together show that **GL-SigRep** is able to learn a graph that is very close to the groundtruth graph when the number of edges matches the number of edges in the groundtruth graph. For comparison, we plot the same curves in Fig. 6.4(b) and Fig. 6.5(b) for **GL-LogDet**, under different values of the parameter λ in Eq. (6.27). In this case, **GL-LogDet** learns more and more correlations, many with relatively small values, as λ decreases. The *Precision* and *Recall* curves still intersect at a point where the number of correlations matches most closely the number of edges in the groundtruth graph. However, both scores are rather low at this point, which leads to an unsatisfactory *F-measure* score in this case. Similarly, the *NMI* scores are very low. Overall, these comparisons show that **GL-SigRep** achieves better performance than **GL-LogDet** in learning the positions of the edges.

Finally, we investigate the influence of the number of signals available for learning, and the level of noise present in the data. In Fig. 6.6, we show the performance of the two algorithms for different numbers of signals for a given instance of the Gaussian RBF graph. As we can see in Fig. 6.6, the performance of both algorithms increases as more signals are available to learn the graph Laplacian and the precision matrix, respectively. The benefit of having more signals seems slightly larger for **GL-SigRep**, as the *F-measure* increases quickly when more than 20 signals are available. In Fig. 6.7, we show the performance of the algorithms for different values of the standard deviation of the Gaussian noise σ_ϵ, for the same groundtruth graph. We see that the performance of both algorithms is rather stable when the noise intensity increases, although **GL-SigRep** seems to be



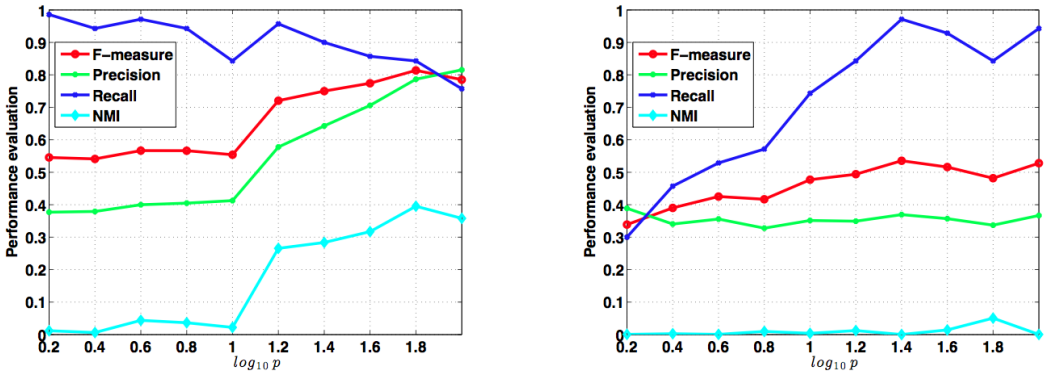
(a) (b)

Figure 6.4: (a) Number of edges in the graphs learned by **GL-SigRep** for different ratios $\frac{\beta}{\alpha}$. (b) Number of correlations in the precision matrices learned by **GL-LogDet** for different values of λ.



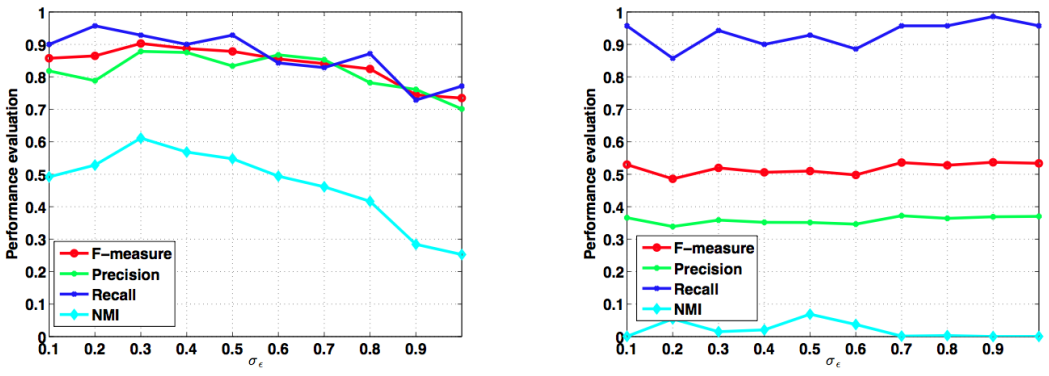
(a) (b)

Figure 6.5: Performance of (a) **GL-SigRep** for different ratios $\frac{\beta}{\alpha}$, and (b) **GL-LogDet** for different values of λ.



(a) (b)

Figure 6.6: Performance of (a) **GL-SigRep** and (b) **GL-LogDet** for different numbers of the signals, p.



(a) (b)

Figure 6.7: Performance of (a) **GL-SigRep** and (b) **GL-LogDet** for different standard deviations of the noise, σ_ϵ.

more sensitive to noise in this experiment where the *F-measure* drops only slightly.

6.5.3 Results on real world data

We now test the proposed graph learning framework on real world data. Specifically, we consider the average monthly temperature data collected at 89 measuring stations in Switzerland (shown in Fig. 6.8) during the period between 1981 and 2010. This leads to 12 signals (i.e., one per month), each of dimension 89, which correspond to the average temperatures at each of the measuring stations. By applying the proposed graph learning algorithm, we would like to infer a graph where stations with similar temperature evolutions across the year are connected. In other words, we aim at learning a graph on which the observed temperature signals are smooth. In this case, the natural choice of a geographical graph based on physical distances between the stations does not seem appropriate for representing the similarity of temperature values between these stations.

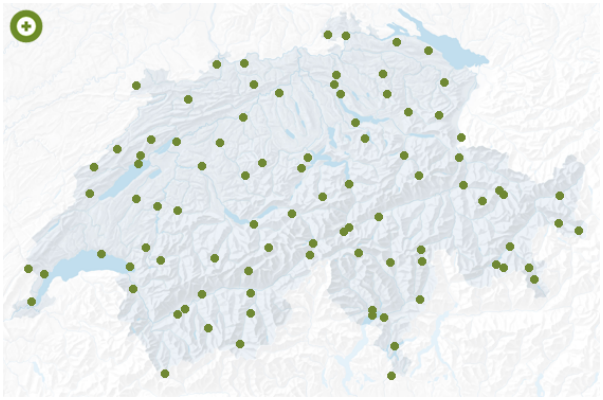
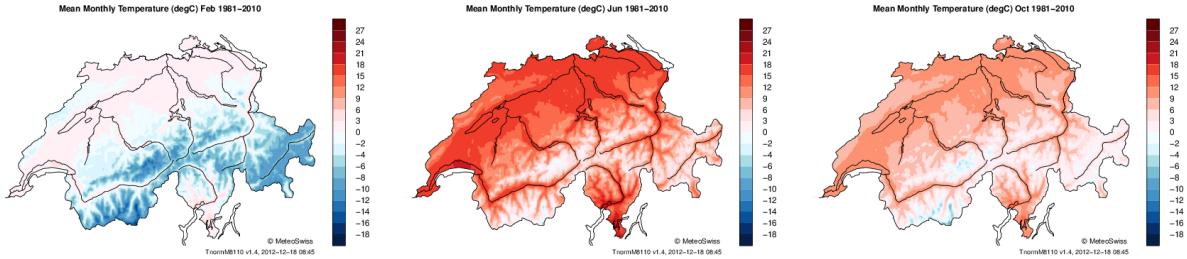


Figure 6.8: The locations of 89 measuring stations in Switzerland (figure from [1]).



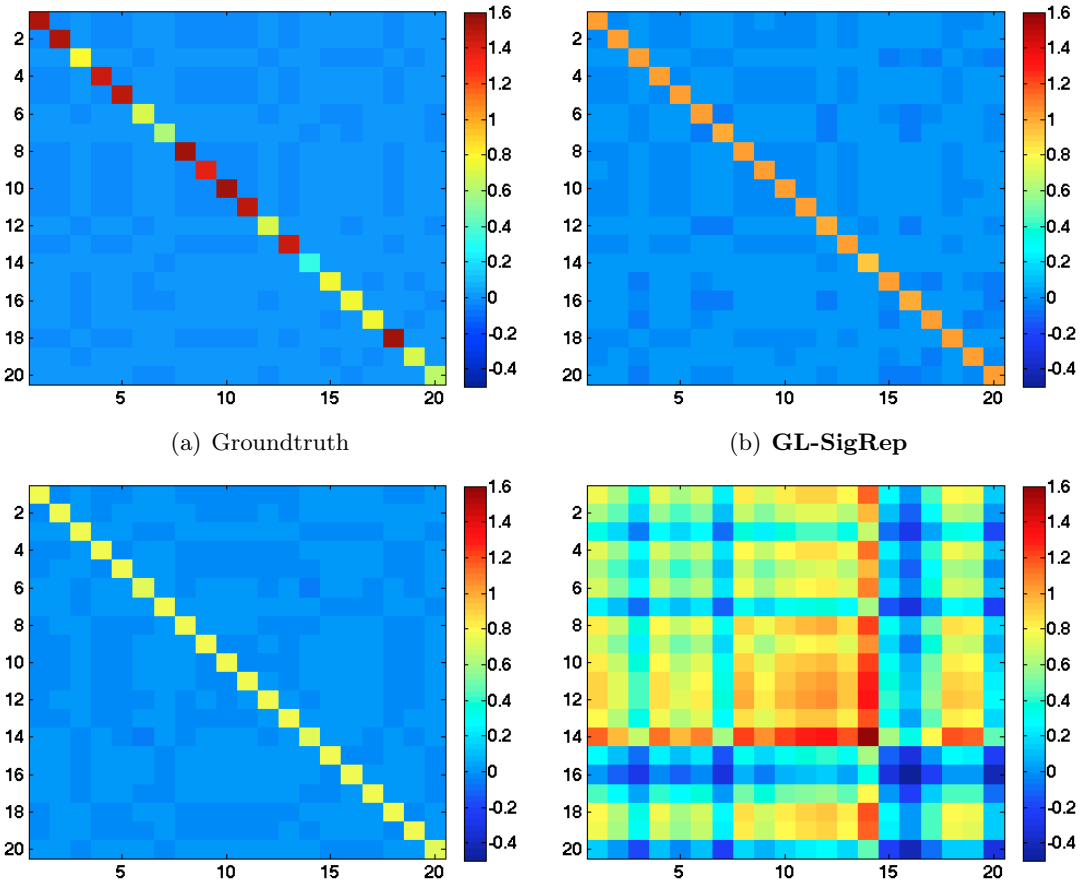
(a) (b) (c)

Figure 6.9: Average monthly temperature in Switzerland during the period from 1981 to 2010 (figure from [2]): (a) February. (b) June. (c) October.

Indeed, Fig. 6.9 shows the average temperatures in Switzerland in February, June and October, and we can see that the evolution of temperatures at most of the stations follow very similar trends and are thus highly correlated, regardless of the geographical distances between them. On the other hand, it turns out that altitude is a more reliable source of information to determine temperature evolutions. For instance, as we observed from the data, the temperature at two stations, Jungfraujoch and Piz Corvatsch, follow similar trends that are clearly different from other stations, possibly due to their similar altitudes (both are more than 3000 metres above sea level). Therefore, we build a groundtruth graph that reflects the similarity between stations in terms of their altitudes. More specifically, we connect two stations with an unitary weight if and only if their altitude difference is smaller than 300 metres. The goal of our learning experiments is then to recover this altitude graph given the 12 observed signals.

We first show visual comparisons between the Laplacian of the groundtruth altitude-based graph, the graph Laplacian learned by **GL-SigRep**, the precision matrix learned by **GL-LogDet**, and the sample covariance matrix. For a more clear visualization, we focus on the top left part of the four matrices and plot them in Fig. 6.10[4]. First, we see that the sample covariance does

[4]The results for **GL-SigRep** and **GL-LogDet** are obtained based on the parameters α, β and λ that lead to the best *F-measure* scores. The values of the sample covariance matrices are scaled before the visualization.



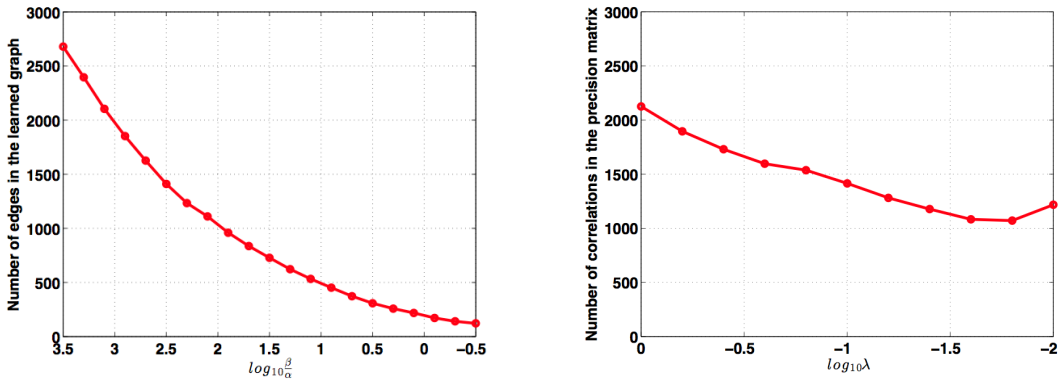
(a) Groundtruth

(b) **GL-SigRep**

(c) **GL-LogDet**

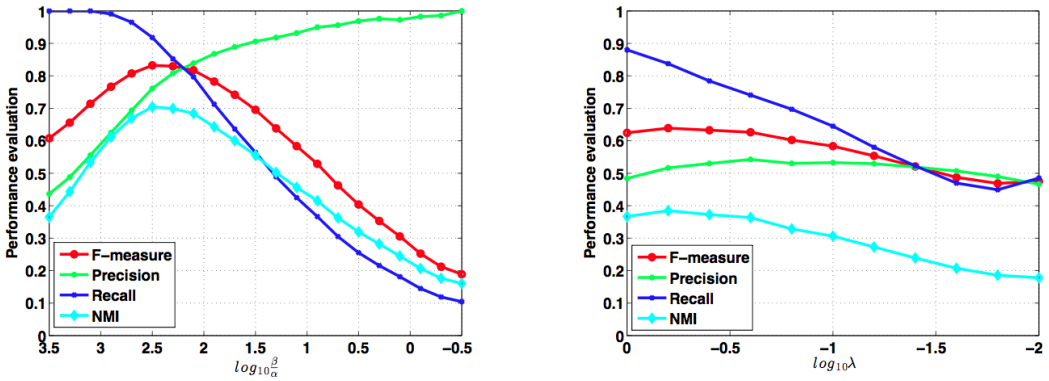
(d) Sample covariance

Figure 6.10: Visual comparisons between (a) The groundtruth graph Laplacian. (b) The graph Laplacian learned by **GL-SigRep**. (c) The precision matrix learned by **GL-LogDet**. (d) The sample covariance matrix.



(a) (b)

Figure 6.11: (a) Number of edges in the graphs learned by **GL-SigRep** for different ratios $\frac{\beta}{\alpha}$. (b) Number of correlations in the precision matrices learned by **GL-LogDet** for different values of λ.



(a) (b)

Figure 6.12: Performance of (a) **GL-SigRep** for different ratios $\frac{\beta}{\alpha}$, and (b) **GL-LogDet** for different values of λ.

not match well the groundtruth, since it contains many correlations that do not appear in the groundtruth graph, as it was the case for the synthetic data. This is especially due to a small number of samples (12) compared to the sample dimension (89) in this case. The comparisons between **GL-SigRep** and **GL-LogDet** show that the edges in the graph learned by our algorithm is again more consistent with the groundtruth data in terms of both positions and weights. Similarly to the synthetic experiments, we investigate the number of edges in the learned graph and the *F-measure* scores for different ratios $\frac{\beta}{\alpha}$ for **GL-SigRep**; the results are presented in Fig. 6.11(a) and Fig. 6.12(a). Clearly, we can observe the same behavior as in the synthetic experiments, namely, when the number of edges matches closely the one in the groundtruth graph (i.e., 1169 edges), the curves for the *Precision* and the *Recall* intersect, and the *F-measure* and *NMI* scores are near their peaks. For the best parameter values, **GL-SigRep** achieves a *F-measure* close to 0.85, which indicates a very good recovery of the topology of the groundtruth graph. In Fig. 6.11(b) and

Fig. 6.12(b), the same results are presented for **GL-LogDet**. We can see that the performance is less convincing in this case. Specifically, both the *Precision* and the *Recall* stay relatively low and do not reach close to 1. As a result, the curve for *F-measure* is less satisfactory and a similar trend is observed for the *NMI* curve. It is worth noting that, in Fig. 6.11(b), the number of correlations in the learned precision matrices decreases as λ decreases, before increasing again in the end. This is not consistent with the monotonically increasing trend that we have observed in Fig. 6.4(b) in the synthetic experiments. By taking a closer look at the precision matrices learned in both cases, we observe that the L^1-norm of the precision matrices always decreases as λ increases. However, the decrease in the L^1-norm does not necessarily translate into a larger sparsity in terms of the number of non-zero entries. In practice, it could happen that more and more non-zero entries with smaller values appear in the precision matrices as the L^1-norm decreases, which is the case in Fig. 6.11(b). Overall, **GL-SigRep** is clearly more reliable than **GL-LogDet** in achieving a desired sparsity level for the learned graph by an appropriate parameter selection.

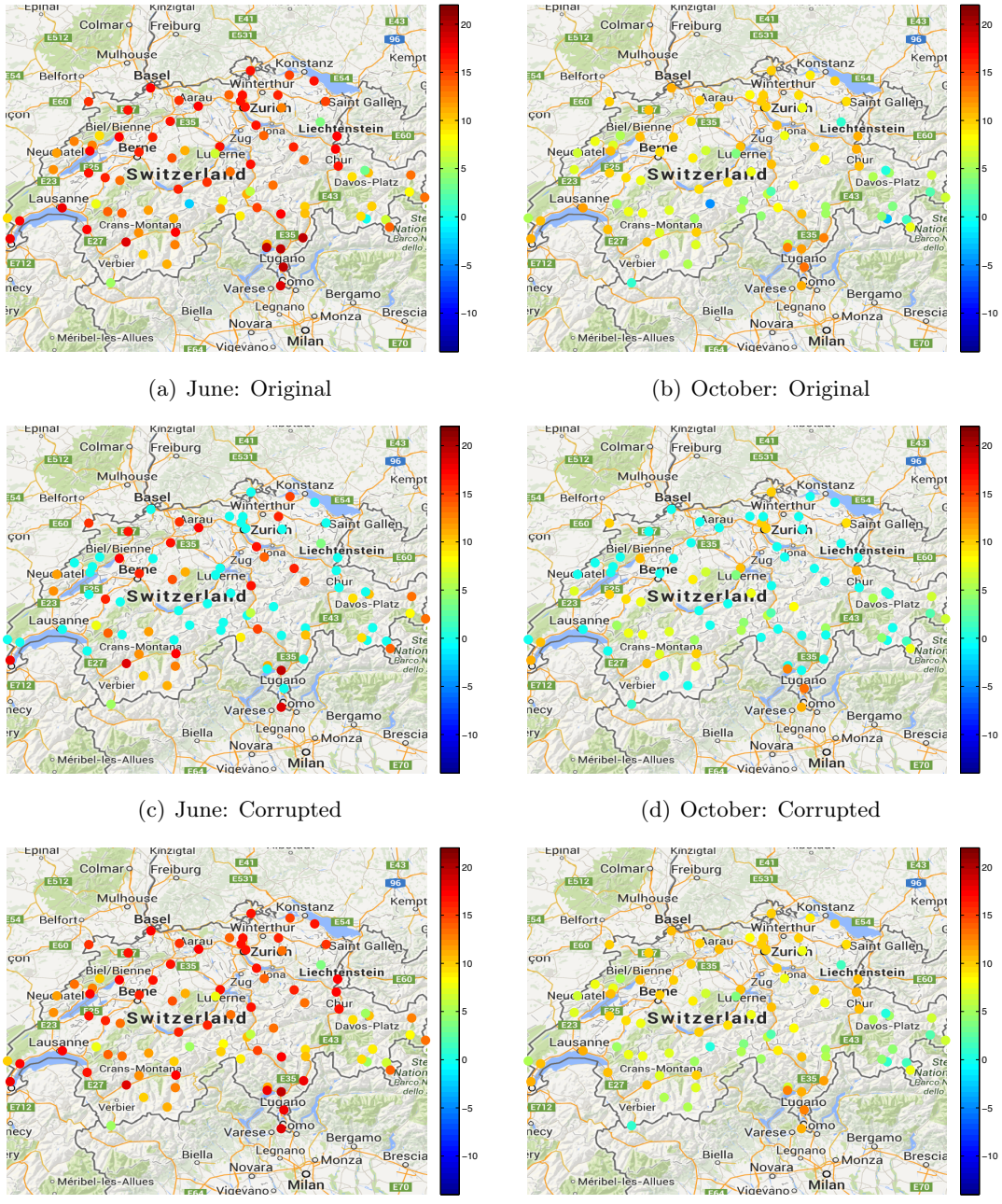
Finally, we propose an example of applying the constructed graph in an interpolation problem via semi-supervised learning. We consider the scenario when the temperature data for some time periods at some measuring stations are corrupted due to problems at these stations. In this case, since the constructed graph contains information about the similarities between the temperature data at different stations across the year, we can try to recover the corrupted observations by assuming that the data is smooth on the learned graph. Specifically, we propose to learn a graph from the complete temperature data for six months of the year, namely, January, March, May, July, September and November. For data from the other six months, we randomly corrupt 50% of the observations by setting the corresponding values to zero. We then recover the corrupted data by solving the following optimization problem:

$$\arg\min_{y} \; ||x - \mathcal{M}y||_2^2 + \gamma \; tr(y^T L y), \tag{6.32}$$

where L is the Laplacian of the learned graph, x and y are the corrupted data and reconstructed observations for each month respectively, and γ is a regularization parameter. The mask operator $\mathcal{M}$ is a diagonal matrix whose diagonal entries are zeros at the locations corresponding to the corrupted observations and are ones otherwise. The problem of Eq. (6.32) has the following closed-form solution:

$$y = (\mathcal{M} + \gamma L)^{-1} \mathcal{M} x. \tag{6.33}$$

We solve the problem of Eq. (6.32) for data for each month with $\gamma = 0.01$, and visualize the reconstruction results for data for two months, namely, June and October, in Fig. 6.13. In this figure, we show from the top to the bottom rows the original observations, corrupted observations and reconstructed data, respectively. We can see that semi-supervised learning on the learned graph indeed helps reconstruct the corrupted observations. We further show in Fig. 6.14 the mean squared error (MSE) between the original signal values in six months, and the corrupted (red) and the reconstructed (blue) signal values, for different percentage of corruption. As expected, the MSE increases as the percentage of corruption increases; however, even with a large percentage of corruption, the algorithm is still able to reconstruct the original signal values reasonably well.



(a) June: Original

(b) October: Original

(c) June: Corrupted

(d) October: Corrupted

(e) June: Reconstructed

(f) October: Reconstructed

Figure 6.13: Signal interpolation using the learned graph. From the top to the bottom rows are the scaled original, corrupted and reconstructed signals, respectively. From the left to the right columns we show results for June and October, respectively.

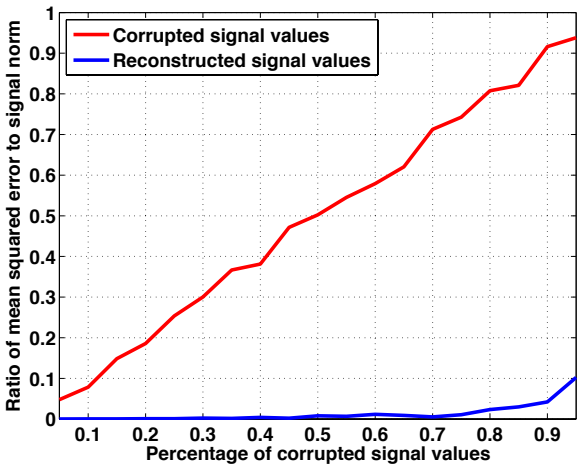


Figure 6.14: The mean squared error between the original signal values and the corrupted (red) and the reconstructed (blue) signal values, for different percentage of data corruption.

6.6 Conclusion

In this chapter, we have presented a framework for learning graph topologies from the observed signals under the assumption that the signals are smooth on the learned graphs. We have developed a method for learning graphs that enforce the smoothness property of the graph signals, under a Gaussian prior distribution imposed on the latent variables in a factor analysis model. In both synthetic and real world experiments, we have shown that the proposed method outperforms an algorithm for estimating a sparse inverse covariance matrix for Gaussian graphical models.

There exist some connections between the framework presented in this chapter and those in the previous chapters, which shows that the proposed graph learning framework can serve as complementary methods to the methods developed earlier in this thesis. Specifically, the proposed graph learning framework in this chapter can be nicely combined with the techniques presented in the previous chapters to solve data processing problems on graphs. Since the learned graph is expected to reveal the intrinsic relationships between the entities than a covariance based graph, it can be used to define a wavelet frame for graph signal classification, as presented in Chapter 5. Graph learning can also help in the problems of learning with multi-layer graphs addressed in Chapter 3 where the proposed graph learning framework can be used to create the individual graphs from each view and then form the multi-layer graph.

Graph learning is a challenging problem from a computational point of view, as the number of variables scales quadratically with the size of the graph. In order to extend the proposed learning framework to graphs of very large scales, computationally efficient methods are certainly worth investigating in further studies. In addition, it would be interesting to explore other prior distributions for the latent variables in the factor analysis model, which would lead to other properties in the representations of the graph signals. Specifically, the generalization of a Laplace prior and the resulting sparse representations in the traditional setting is certainly one of the future research directions in graph learning for efficient signal representations.

Chapter 7

Conclusion

7.1 Main contributions

In this thesis, we have addressed several problems related to the representation, clustering, classification, and learning of complex and multi-view data defined on weighted and undirected graphs. In particular, we have studied the problem of learning the relationships between entities in multi-layer graphs. We have addressed the graph signal classification problem based on spectral graph features. We have proposed a novel framework for learning graphs given observed data, such that they become regular on the constructed graphs.

We have first addressed the problem of analyzing data represented as multi-layer graphs. The key challenge in this problem is to efficiently combine information contained in the individual graph layers. We have first adopted a machine learning approach, where we model the individual layers as subspaces on a Grassmann manifold. The problem of combining multiple layers is then transformed into the problem of merging subspaces on the manifold. To this end, we have studied the distance metrics defined on the Grassmann manifold, and proposed a framework to find a representative subspace whose average distance from the individual subspaces is minimized. We have shown that the resulting subspace can be used for clustering the vertices of the multi-layer graphs. Then, to better take into account the respective importance of individual layers, we have approached the same problem from a signal processing viewpoint. Specifically, we have considered the eigenvectors of the graph Laplacian of one layer as signals defined on the vertex set of another layer. We have then proposed a novel graph regularization framework to find a set of joint eigenvectors, or a joint spectrum, of the two layers, and further extended it to the setting of multiple layers. We have again tested our framework in clustering methods in multi-layer graphs. Our first framework provides a generic and principled way of learning with multi-layer graphs, while the second approach allows us to take into account the respective importance of the individual layers in a unique and convincing way.

Next, we have studied the problem of classification of graph signals. The main difficulty of graph signal classification comes from the irregular support of the signals and the design of discriminative and robust features. To address these challenges, we have adopted signal representations in the graph spectral domain, and studied the benefits of such representations for feature construction in classifying graph signals under a generic model. We have shown that localized spectral rep-

resentations are of great help in classifying signals with noticeable local variations, such as those following the random walk model. We have then proposed discriminative spectral features for graph signal classification and tested their performances in both synthetic and real world experiments. In particular, we have successfully applied our spectral classification framework to the real world problem of the inference of mobility patterns in traffic and social media data. Our classification framework constitutes one of the first complete studies of the graph signal classification problem in the emerging research field of signal processing on graphs.

Finally, we have considered the problem of learning graph topologies given observed signals. From a signal processing point of view, the challenge in the learning of the graphs mainly lies in finding a criteria that matches the signal values and the graph topologies in a meaningful way. Such a criteria can thus guide the learning process. Inspired by the representation learning theory in the traditional signal setting, we have approached this problem from the perspective of signal representations on graphs. Specifically, we have first established links between signal representations in the traditional setting and the graph setting, and then generalized the factor analysis model for the representation of graph signals. We have shown that a Gaussian probabilistic prior imposed on the latent variables in the factor analysis model leads to smooth signal representations on graphs. We have then designed a graph learning framework that learns graph topologies and enforces the smoothness of the observed signals on the resulting graph. We have tested our framework in synthetic and real world graph learning experiments, and demonstrated that its performance is superior to that of a similar approach in the machine learning community. Our graph learning framework is one of the first rigorous frameworks to solve the challenging problem of graph learning in graph signal processing.

To summarize, we have studied in this thesis several emerging research problems related to multi-view data processing associated with various forms of graphs, and provided novel signal processing and machine learning solutions. We believe that the research effort in this thesis provides an example to show the ever-decreasing gap between these two traditionally different research fields, and approaches from both should be leveraged together to meet the challenges of modern information processing with complex data.

7.2 Perspectives

The research problems and proposed approaches presented in this thesis open the following new interesting research directions. First, our model of multi-layer graphs can be further extended to the settings where (i) different graph layers do not necessarily have the same number of vertices, (ii) vertices in different layers only have partially known correspondence, and (iii) individual layers that correspond to temporal observations evolve with time. For example, the works in [101, 69, 20] propose to use functional maps (namely, a given set of corresponding functions) to replace the assumption on the bijective correspondences between vertices in different graphs. These settings pose even bigger challenges in modeling dynamic and incomplete multi-view relationships between the entities in the dataset, where a Grassmann manifold of fixed dimension becomes insufficient for properly representing the data. In such cases, subspace representations of individual layers may also need to take into account the more complex settings. For example, instead of the representation from the spectral embedding of the graph, we could consider subspaces defined by kernels

or dictionaries on graphs. By properly designing the spectral filters behind the kernels and the dictionaries, we could gain more flexibility and adaptivity for representing dynamic and incomplete data.

Second, we have adopted the spectral graph wavelets to construct representations of graph signals for classification. Although wavelets are attractive tools for signal representation and processing, they may not be adaptive enough to handle different classes of graph signals given the specific learning tasks at hand. Dictionaries on graphs provide an appealing alternative, since they are usually learned from the data and can be designed in such a way that they are tailored for the specific learning tasks. In addition, we have based our classification analysis on the random walk model for graph signals, and used classical learning models such as the SVMs as the classifier after feature construction. It would be interesting to design novel classifiers dedicated to the specific signal models of interest. Dictionaries can again show their advantages in the research effort towards this direction due to their flexibility and adaptivity. For example, recent studies in [47] show that properly designed dictionaries can lead to very efficient linear classifiers with competitive performance compared to the state-of-the-art.

Finally, our proposed graph learning framework is based on a Gaussian probabilistic prior on the latent variables in the factor analysis model. It would certainly be interesting to explore other priors that would lead to other desirable properties for the graphs signals, such as a Laplace prior and the resulting sparse representations. Furthermore, instead of a linear representation model such as the factor analysis model we have considered, it is tempting to also consider nonlinear representation models that would explain more complex behaviors of the graph signals and their interactions with the graph topologies. This would not only benefit the specific research problem of graph learning, but also shed new lights in the understanding of the representation and approximation of graph signals in the field of signal processing on graphs.

Multi-view data processing on graphs, as an increasingly popular research topic, presents the typical challenges we need to meet in the analysis and understanding of the "Big Data" flows. First, we need more flexible and robust graph models for the complex, noisy and incomplete data of various forms. Second, we need more efficient graph-based signal processing and machine learning algorithms to handle large-scale data, while still maintaining satisfactory performance. Finally, we need to demonstrate that theoretical methods can indeed be applied to real world scenarios and benefit practical applications. It is our belief that the models, methods, and applications presented in this thesis serve as first attempts in the research towards this direction.

Appendix A

Classification analysis of the Gaussian Markov Random Field model

In this appendix, we analyze the efficiency of adopting spatial and spectral representations of graph signals for feature construction in graph signal classification applications, for graph signals that follow a Gaussian Markov Random Field (GMRF) model. More specifically, for a given graph signal $\{f, G\}$, this model assumes that f is a GMRF with respect to G, with the precision matrix in the GMRF chosen as the graph Laplacian L. Our GMRF model essentially assumes that the graph signal follows a multivariate Gaussian distribution. This is an important class of graph signals to study, since (i) it is closely related to the Gaussian distribution in classical settings, and (ii) it is a smooth model that could explain the behaviors of graph signals in many real world applications, such as temperature changes in a local region or resource allocations in electrical and transportation networks.

A.1 Distributions of the GFT and SGWT coefficients

We first show the following results about the distributions of the GFT and SGWT coefficients[1].

Lemma 2. *If the signal $f \in \mathbb{R}^n$ is a GMRF with respect to a graph G, with a mean u_f and a precision matrix chosen as the graph Laplacian L, then the GFT coefficients of f, namely, $\hat{f}^G$, follows a multivariate Gaussian distribution.*

Proof. By definition, a GMRF f with a mean u_f and a precision matrix L has the following density:

$$p(f) = (2\pi)^{-\frac{n}{2}} |L|^{\frac{1}{2}} \exp\left(-\frac{1}{2}(f - u_f)^T L (f - u_f)\right), \tag{A.1}$$

where $|\cdot|$ denote the operators taking the determinant of a matrix. In other words, f follows a degenerate multivariate Gaussian distribution:

$$f \sim \mathcal{N}(u_f, L^\dagger), \tag{A.2}$$

[1]Similar assumptions about the distributions of the signal and its GFT coefficients have been discussed in [65].

where the covariance matrix $L^\dagger$ is the Moore-Penrose pseudoinverse of L. According to the definition of GFT, we have $\hat{f}^{\mathrm{G}} = \chi^T f$ where χ is the matrix containing the eigenvectors of L as columns. Therefore, following the properties of multivariate Gaussian distributions under linear transformation, and writing the eigendecomposition of $L^\dagger$ as $L^\dagger = \chi\Lambda^\dagger\chi^T$, we have:

$$\hat{f}^{\mathrm{G}} \sim \mathcal{N}(\chi^T u_f, \Lambda^\dagger), \tag{A.3}$$

where $\Lambda^\dagger$ is the pseudoinverse of the eigenvalue matrix Λ. This shows that $\hat{f}^{\mathrm{G}}$ follows a degenerate multivariate Gaussian distribution. □

Lemma 3. *If the signal $f \in \mathbb{R}^n$ is a GMRF with respect to a graph G with a mean u_f and a precision matrix chosen as the graph Laplacian L, then, the SGWT coefficients of f at one particular scale t_j, namely, $\hat{f}^W_{t_j}$, follows a multivariate Gaussian distribution.*

Proof. Given a wavelet kernel function g, by the construction of SGWT we have $\hat{f}^{\mathrm{W}}_{t_j} = \chi g(t_j\Lambda)\chi^T f$ at scale t_j [62]. Similarly to the proof of Lemma 2, by applying the properties of multivariate Gaussian distributions, we have:

$$\hat{f}^{\mathrm{W}}_{t_j} \sim \mathcal{N}\left(\chi g(t_j\Lambda)\chi^T u_f, \chi g(t_j\Lambda)\Lambda^\dagger g(t_j\Lambda)^T\chi^T\right). \tag{A.4}$$

Since $g(t_j\Lambda)$ is a diagonal matrix, we have:

$$\hat{f}^{\mathrm{W}}_{t_j} \sim \mathcal{N}\left(\chi g(t_j\Lambda)\chi^T u_f, \chi g(t_j\Lambda)\Lambda^\dagger g(t_j\Lambda)\chi^T\right). \tag{A.5}$$

This shows that $\hat{f}^{\mathrm{W}}_{t_j}$ follows a degenerate multivariate Gaussian distribution. □

Without loss of generality, in the following analysis we assume that $u_f = 0$, therefore Eq. (A.2), Eq. (A.3) and Eq. (A.5) can be simplified as follows:

$$\begin{aligned} f &\sim \mathcal{N}(0, L^\dagger), \\ \hat{f}^{\mathrm{G}} &\sim \mathcal{N}(0, \Lambda^\dagger), \\ \hat{f}^{\mathrm{W}}_{t_j} &\sim \mathcal{N}(0, \chi g(t_j\Lambda)\Lambda^\dagger g(t_j\Lambda)\chi^T). \end{aligned} \tag{A.6}$$

A.2 Classification analysis

The above results allow us to analyze the efficiency of signal representations in the spatial domain (which is f itself) and in the graph spectral domain (which are its GFT and SGWT coefficients) for feature constructions in classification. Let us define two graph signals $\{f_1, G_1\}$ and $\{f_2, G_2\}$, where both f_1 and f_2 are zero-mean. We denote by L_1, Λ_1, χ_1 the graph Laplacian matrix, its eigenvalue matrix, and its eigenvector matrix for G_1, respectively, and similarly L_2, Λ_2 and χ_2 for G_2. We consider the following problem, where we would like to distinguish the two graph signals to see which one is better explained by a GMRF model, for the purpose of classification. Given features x_1 and x_2 that are respectively constructed based on $\{f_1, G_1\}$ and $\{f_2, G_2\}$, this is equivalent to comparing the posterior probabilities $P(w_{\mathrm{GMRF}}|x_1)$ and $P(w_{\mathrm{GMRF}}|x_2)$, where the

class w_{GMRF} means that the graph signal follows a GMRF model. By applying Bayes' rule, we can use the class-conditional probability instead of the posterior probability and have the following decision function z:

$$z = P(x_1|w_{\text{GMRF}}) - P(x_2|w_{\text{GMRF}}). \tag{A.7}$$

The decision rule is that $\{f_1, G_1\}$ is better explained by a GMRF model if $z > 0$ and $\{f_2, G_2\}$ otherwise.

We now compare the discriminating power of three feature choices, namely, the signal values f, its GFT coefficients $\hat{f}^{\text{G}}$, and its SGWT coefficients $\hat{f}^{\text{W}}_{t_j}$ at one particular scale t_j. Incorporating the statistics of Eq. (A.6) (namely, the class-conditional probabilities) into Eq. (A.7), when the signal values f_1 and f_2 are adopted as features, we have the following decision function z_f for our problem:

$$z_f = (-\frac{1}{2} f_1^T L_1 f_1 + \frac{1}{2}\ln|L_1|_+) - (-\frac{1}{2} f_2^T L_2 f_2 + \frac{1}{2}\ln|L_2|_+). \tag{A.8}$$

Similarly, when $\hat{f}_1^{\text{G}} = \chi_1^T f_1$ and $\hat{f}_2^{\text{G}} = \chi_2^T f_2$ are adopted as features, we have the following decision function z_G:

$$z_G = \left(-\frac{1}{2}(\hat{f}_1^{\text{G}})^T \Lambda_1 (\hat{f}_1^{\text{G}}) + \frac{1}{2}\ln|\Lambda_1|_+\right) - \left(-\frac{1}{2}(\hat{f}_2^{\text{G}})^T \Lambda_2 (\hat{f}_2^{\text{G}}) + \frac{1}{2}\ln|\Lambda_2|_+\right). \tag{A.9}$$

Finally, when $\hat{f}_1^{\text{W}} = \chi_1 g(t_{j_1}\Lambda_1)\chi_1^T f_1$ and $\hat{f}_2^{\text{W}} = \chi_2 g(t_{j_2}\Lambda_2)\chi_2^T f_2$ are adopted as features, we have the following decision function z_W:

$$\begin{aligned} z_W = &\left(-\frac{1}{2}(\hat{f}_1^{\text{W}})^T \chi_1 g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)\chi_1^T (\hat{f}_1^{\text{W}}) + \frac{1}{2}\ln|\chi_1 g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)\chi_1^T|_+\right) \\ &- \left(-\frac{1}{2}(\hat{f}_2^{\text{W}})^T \chi_2 g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)\chi_2^T (\hat{f}_2^{\text{W}}) + \frac{1}{2}\ln|\chi_2 g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)\chi_2^T|_+\right), \end{aligned} \tag{A.10}$$

where t_{j_1} and t_{j_2} determine the j-th scale for the two wavelets, and $|\cdot|_+$ denotes the operator taking the pseudo-determinant of a singular matrix. In Eq. (A.10), we have taken the pseudoinverse of $g^\dagger(t_{j_1}\Lambda_1)$ and $g^\dagger(t_{j_2}\Lambda_2)$ due to their singularity.

We compare the discriminating power of three decision functions as follows. Assuming that the pseudo-determinant terms stay within a similar range[2] for G_1 and G_2, z_f, z_G and z_W are dominated by the difference between the quadratic terms in Eq. (A.8), Eq. (A.9) and Eq. (A.10). First, we see from Eq. (A.8) that, when the signal values are adopted as features, the decision function z_f simply evaluates the sign of the difference between two Laplacian quadratic forms $\frac{1}{2} f^T L_2 f - \frac{1}{2} f^T L_1 f$. That is, we rely on the quantity $\frac{1}{2} f^T L_2 f - \frac{1}{2} f^T L_1 f$ to distinguish which graph signal is better explained by the GMRF model. Furthermore, notice that $\hat{f}_1^{\text{G}}$, $\hat{f}_1^{\text{W}}$, and $\hat{f}_2^{\text{G}}$, $\hat{f}_2^{\text{W}}$, are obtained by linear transformations of f_1 and f_2, respectively. It turns out that, when the GFT coefficients are adopted as features, the difference between the quadratic terms in Eq. (A.9) is of exactly the same

[2]Since L, Λ and $\chi g^\dagger(t_j\Lambda)\Lambda g^\dagger(t_j\Lambda)\chi^T$ are all positive semi-definite matrices, their pseudo-determinants are the products of non-zero eigenvalues. Therefore, we assume that L_1 and L_2 have similar eigenvalue product, which is reasonable especially when we take the normalized versions of L_1 and L_2, whose eigenvalues are bounded between 0 and 2.

form as $\frac{1}{2}f^T L_2 f - \frac{1}{2}f^T L_1 f$. The same statement holds for the SGWT features and the difference between the quadratic terms in Eq. (A.10), as long as the diagonals of $g(t_{j_1}\Lambda_1)$ and $g(t_{j_2}\Lambda_2)$ vanish only at zero, which is the case for the SGWT implementation in [62]. Indeed, using the fact that $\hat{f}_1^{\mathrm{G}} = \chi_1^T f_1$ and $\hat{f}_2^{\mathrm{G}} = \chi_2^T f_2$, we have:

$$\begin{aligned}
&\left(-\frac{1}{2}(\hat{f}_1^{\mathrm{G}})^T\Lambda_1(\hat{f}_1^{\mathrm{G}})\right) - \left(-\frac{1}{2}(\hat{f}_2^{\mathrm{G}})^T\Lambda_2(\hat{f}_2^{\mathrm{G}})\right)\\
=&\frac{1}{2}f_2^T\chi_2\Lambda_2\chi_2^T f_2 - \frac{1}{2}f_1^T\chi_1\Lambda_1\chi_1^T f_1\\
=&\frac{1}{2}f_2^T L_2 f_2 - \frac{1}{2}f_1^T L_1 f_1.
\end{aligned} \tag{A.11}$$

Similarly, using the fact that $\hat{f}_1^{\mathrm{W}} = \chi_1 g(t_{j_1}\Lambda_1)\chi_1^T f_1$ and $\hat{f}_2^{\mathrm{W}} = \chi_2 g(t_{j_2}\Lambda_2)\chi_2^T f_2$, we have:

$$\begin{aligned}
&\left(-\frac{1}{2}(\hat{f}_1^{\mathrm{W}})^T\chi_1 g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)\chi_1^T(\hat{f}_1^{\mathrm{W}})\right)\\
&-\left(-\frac{1}{2}(\hat{f}_2^{\mathrm{W}})^T\chi_2 g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)\chi_2^T(\hat{f}_2^{\mathrm{W}})\right)\\
=&\frac{1}{2}f_2^T\chi_2 g(t_{j_2}\Lambda_2)\chi_2^T\chi_2 g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)\chi_2^T\chi_2 g(t_{j_2}\Lambda_2)\chi_2^T f_2\\
&-\frac{1}{2}f_1^T\chi_1 g(t_{j_1}\Lambda_1)\chi_1^T\chi_1 g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)\chi_1^T\chi_1 g(t_{j_1}\Lambda_1)\chi_1^T f_1\\
=&\frac{1}{2}f_2^T\chi_2 g(t_{j_2}\Lambda_2)g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)g(t_{j_2}\Lambda_2)\chi_2^T f_2\\
&-\frac{1}{2}f_1^T\chi_1 g(t_{j_1}\Lambda_1)g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)g(t_{j_1}\Lambda_1)\chi_1^T f_1.
\end{aligned} \tag{A.12}$$

From Eq. (A.12) we see that, when the diagonals of $g(t_{j_1}\Lambda_1)$ and $g(t_{j_2}\Lambda_2)$ vanish only at zero, we have:

$$\begin{aligned}
g(t_{j_1}\Lambda_1)g^\dagger(t_{j_1}\Lambda_1) = g^\dagger(t_{j_1}\Lambda_1)g(t_{j_1}\Lambda_1) = \mathrm{diag}(0,1,...,1),\\
g(t_{j_2}\Lambda_2)g^\dagger(t_{j_2}\Lambda_2) = g^\dagger(t_{j_2}\Lambda_2)g(t_{j_2}\Lambda_2) = \mathrm{diag}(0,1,...,1).
\end{aligned} \tag{A.13}$$

In this case, the right hand side of Eq. (A.12) can be further simplified as:

$$\begin{aligned}
&\frac{1}{2}f_2^T\chi_2 g(t_{j_2}\Lambda_2)g^\dagger(t_{j_2}\Lambda_2)\Lambda_2 g^\dagger(t_{j_2}\Lambda_2)g(t_{j_2}\Lambda_2)\chi_2^T f_2\\
&-\frac{1}{2}f_1^T\chi_1 g(t_{j_1}\Lambda_1)g^\dagger(t_{j_1}\Lambda_1)\Lambda_1 g^\dagger(t_{j_1}\Lambda_1)g(t_{j_1}\Lambda_1)\chi_1^T f_1\\
=&\frac{1}{2}f_2^T\chi_2\Lambda_2\chi_2^T f_2 - \frac{1}{2}f_1^T\chi_1\Lambda_1\chi_1^T f_1\\
=&\frac{1}{2}f_2^T L_2 f_2 - \frac{1}{2}f_1^T L_1 f_1.
\end{aligned} \tag{A.14}$$

The above analysis implies that, from a Bayesian point of view, the three feature choices, namely, the signal values, its GFT coefficients, or its SGWT coefficients at one particular scale t_j, do not

have a big difference in terms of discriminating power in our problem.

A.3 Further discussions

Let us take a closer look at the probabilistic assumption that we have on the data in this model. In Eq. (A.1), the choice of the precision matrix being the graph Laplacian L has been discussed in [139] and is widely used as an image prior model. On the one hand, since the eigenvectors of L are also the eigenvectors of the covariance $L^{\dagger}$, namely,

$$L = \chi \Lambda \chi^T, \tag{A.15}$$

$$L^{\dagger} = \chi \Lambda^{\dagger} \chi^T, \tag{A.16}$$

the GFT can be considered as the Karhunen-Loève transform (KLT) for graph signals, which decorrelates completely the random variables in f and compacts the energy along the directions of the principal components. However, although such properties are desirable for signal representation, the KLT, or its discrete version, the Principal Component Analysis (PCA), might not be efficient for discrimination [39]. Therefore, the GFT coefficients may not be beneficial as features in classification. On the other hand, the probability distribution in Eq. (A.1) implies that a zero-mean signal f with a smaller quantity $f^T L f$ is more likely to be observed. Since the Laplacian quadratic form $f^T L f$ is usually considered as a smoothness measure of signals on graphs [144], we can essentially think of GMRF as a smooth signal model on graphs. It is important to point out that, in this case, the signal is considered globally smooth on the graph, without significant variations in a particular local region of the graph. As a result, most of the energy of the signal tends to be contained in the low frequency components, and a transform that is aimed at capturing the local variations of the signal (or its high frequency components), such as the SGWT, usually does not differ much from the GFT.

Nevertheless, in practical classification tasks, one usually does not work directly with the representations themselves, namely the signal values in the spatial domain or the coefficients in the graph spectral domain, but rather with features built on these representations. Since the SGWT is a redundant transform that is able to capture different spectral components of the graph signal, the SGWT coefficients are potentially more powerful for feature construction even when most of the signal energy is contained in its low frequency components. If it is constructed properly, it is at least as good as other global representations such as the signal values or the GFT coefficients.

Bibliography

[1] http://www.meteosuisse.admin.ch/web/en/climate/swiss_climate/climate_diagrams_from_swiss_measuring_stations.html.

[2] http://www.meteosuisse.admin.ch/web/en/climate/swiss_climate/Klimakarten_schweiz_1961-90.html.

[3] Y. Aflalo, A. M. Bronstein, M. M. Bronstein, and R. Kimmel, "Deformable shape retrieval by learning diffusion kernels," in *Proceedings of the Third international conference on Scale Space and Variational Methods in Computer Vision (SSVM)*, 2011, pp. 689–700.

[4] A. Agaskar and Y. M. Lu, "A spectral graph uncertainty principle," *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4338–4356, Jul 2013.

[5] A. Argyriou, M. Herbster, and M. Pontil, "Combining graph Laplacians for semi-supervised learning," in *Advances in Neural Information Processing Systems 18 (NIPS)*, 2005.

[6] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *Journal of Machine Learning Research*, vol. 9, pp. 485–516, Jun 2008.

[7] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct 1999.

[8] D. J. Bartholomew, M. Knott, and I. Moustaki, "Latent variable models and factor analysis: A unified approach (3rd edition)," *Wiley*, Jul 2011.

[9] A. Basilevsky, "Statistical factor analysis and related methods," *Wiley*, Jun 1994.

[10] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *arXiv:1206.5538*, 2012.

[11] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi, "Foundations of multidimensional network analysis," in *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2011, pp. 485–489.

[12] R. Bhatia, "Matrix analysis," *Springer, New York*, 1997.

[13] S. Bickel and T. Scheffer, "Multi-view clustering," in *Proceedings of IEEE International Conference on Data Mining (ICDM)*, 2004.

[14] M. B. Blaschko and C. H. Lampert, "Correlational spectral clustering," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[15] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the 18th Annual Conference on Computational Learning Theory*, 1998.

[16] O. Bousquet, O. Chapelle, and M. Hein, "Measure based regularization," in *Advances in Neural Information Processing Systems 16 (NIPS)*, 2004, pp. 1221–1228.

[17] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*, 2004.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[19] J. C. Bremer, R. R. Coifman, M. Maggioni, and A. D. Szlam, "Diffusion wavelet packets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 95–112, Jul 2006.

[20] M. M. Bronstein and K. Glashoff, "Heat kernel coupling for multiple graph analysis," in *arXiv:1312.3035*, 2013.

[21] M. M. Bronstein, K. Glashoff, and T. A. Loring, "Making Laplacians commute," in *arXiv:1307.6549*, 2013.

[22] E. Bruno and S. Marchand-Maillet, "Multiview clustering: A late fusion approach using latent models," in *Proceedings of ACM SIGIR Conference on Research and Development on Information Retrieval*, 2009.

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, Apr 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[24] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *Proceedings of International Conference on Machine Learning (ICML)*, 2009.

[25] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Adaptive graph filtering: Multiresolution classification on graphs," in *Proceedings of IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013, pp. 427–430.

[26] Y. Cheng and R. Zhao, "Multiview spectral clustering via ensemble," in *Proceedings of IEEE International Conference on Granular Computing*, 2009.

[27] Y. Chikuse, "Statistics on special manifolds," *Lecture Notes in Statistics, Springer, New York*, vol. 174, 2003.

[28] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu, "Automatic construction of travel itineraries using social breadcrumbs," in *Proceedings of ACM Conference on Hypertext and Hypermedia*, 2010, pp. 35–44.

[29] F. R. K. Chung, "Spectral graph theory," *CBMS Regional Conference Series in Mathematics, American Mathematical Society*, 1997.

[30] R. R. Coifman and M. Maggioni, "Diffusion wavelets," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 53–94, Jul 2006.

[31] M. Crovella and E. Kolaczyk, "Graph wavelets for spatial traffic analysis," in *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, vol. 3, 2003, pp. 1848–1857.

[32] J. de J. G. Leandro, R. M. Cesar, and R. S. Feris, "Shape analysis using the spectral graph wavelet transform," in *Proceedings of IEEE International Conference on eScience (eScience)*, 2013, pp. 307–316.

[33] V. R. de Sa, "Spectral clustering with two views," in *ICML Workshop on Learning with Multiple Views*, 2005.

[34] T. M. T. Do and D. Gatica-Perez, "Contextual conditional models for smartphone-based human mobility prediction," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp)*, 2012, pp. 163–172.

[35] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering with multi-layer graphs: A spectral perspective," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5820–5831, Nov 2012.

[36] ——, "Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds," in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013.

[37] X. Dong, A. Ortega, P. Frossard, and P. Vandergheynst, "Inference of mobility patterns via spectral graph wavelets," in *Proceedings of the 38th International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 3118–3122.

[38] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering on multi-layer graphs via subspace analysis on Grassmann manifolds," *IEEE Transactions on Signal Processing*, vol. 62, no. 4, pp. 905–918, Feb 2014.

[39] R. O. Duda, P. E. Hart, and D. G. Stork, "Pattern classification (2nd edition)," *Wiley*, 2000.

[40] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, 2006.

[41] N. Eagle, A. Pentland, and D. Lazer, "Inferring social network structure using mobile phone data," in *Proceedings of the National Academy of Sciences*, vol. 106, no. 36, 2009, pp. 15 274–15 278.

[42] N. Eagle, A. Clauset, A. Pentland, and D. Lazer, "Multi-dimensional edge inference," *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, vol. 107, no. 9, p. E31, 2010.

[43] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.

[44] V. Ekambaram, G. Fanti, B. Ayazifar, and K. Ramchandran, "Wavelet regularized graph semi supervised learning," in *Proceedings of IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013, pp. 423–426.

[45] P. Erdős and A. Rényi, "On the evolution of random graphs," *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, vol. 5, pp. 17–61, 1960.

[46] D. Eynard, K. Glashoff, M. M. Bronstein, and A. M. Bronstein, "Multimodal diffusion geometry by joint diagonalization of Laplacians," *arXiv:1209.2295*, 2012.

[47] A. Fawzi, M. Davies, and P. Frossard, "Dictionary learning for fast classification based on soft-thresholding," *arXiv:1402.1973*, 2014.

[48] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75–174, Feb 2010.

[49] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, Jul 2008.

[50] M. Gavish, B. Nadler, and R. R. Coifman, "Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning," in *Proceedings of International Conference on Machine Learning (ICML)*, 2010, pp. 367–374.

[51] Z. Ghahramani and G. E. Hinton, "The EM algorithm for mixtures of factor analyzers," *Technical Report CRG-TR-96-1, University of Toronto*, May 1996.

[52] F. Girardin, F. D. Fiore, C. Ratti, and J. Blat, "Leveraging explicitly disclosed location information to understand tourist dynamics: A case study," *Journal of Location Based Services*, vol. 2, no. 1, pp. 41–56, 2008.

[53] G. H. Golub and C. F. V. Loan, "Matrix computations (3rd edition)," *Johns Hopkins University Press*, 1996.

[54] M. Grant and S. Boyd, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.

[55] ——, "CVX: Matlab software for disciplined convex programming, version 2.0 beta," http://cvxr.com/cvx, Sep. 2013.

[56] D. Greene and P. Cunningham, "A matrix factorization approach for integrating multiple data views," in *Proceedings of European Conference on Machine Learning and Knowledge Discovery in Databases*, 2009.

[57] A. Gretton, O. Bousquet, A. Smola, and B. Scholkopf, "Measuring statistical dependence with Hilbert-Schmidt norms," in *Proceedings of International Conference on Algorithmic Learning Theory*, 2005.

[58] R. Gribonval, "Should penalized least squares regression be interpreted as maximum a posteriori estimation?" *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2405–2410, May 2011.

[59] J. Hamm and D. Lee, "Grassmann discriminant analysis: A unifying view on subspace-based learning," in *Proceedings of International Conference on Machine Learning (ICML)*, 2008.

[60] ——, "Extended Grassmann kernels for subspace-based learning," in *Advances in Neural Information Processing Systems 21 (NIPS)*, 2008.

[61] J. Hamm, D. Lee, and L. Saul, "Semisupervised alignment of manifolds," in *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2004.

[62] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, Mar 2011.

[63] R. P. Horaud, "A short tutorial on graph Laplacians, Laplacian embedding, and spectral clustering." [Online]. Available: http://csustan.csustan.edu/~tom/Lecture-Notes/Clustering/GraphLaplacian-tutorial.pdf

[64] C. Hu, L. Cheng, J. Sepulcre, G. E. Fakhri, Y. M. Lu, and Q. Li, "A graph theoretical regression model for brain connectivity learning of Alzheimer's disease," in *Proceedings of International Symposium on Biomedical Imaging (ISBI)*, 2013.

[65] ——, "Matched signal detection on graphs: Theory and application to brain network classification," in *Proceedings of International Conference on Information Processing in Medical Imaging (IPMI)*, 2013, pp. 1–12.

[66] P. Jain, B. Kulis, J. V. Davis, and I. S. Dhillon, "Metric and kernel learning using a linear transformation," *Journal of Machine Learning Research*, vol. 13, pp. 519–547, Mar 2012.

[67] N. Kiukkonen, J. Blom, O. Dousse, D. Gatica-Perez, and J. Laurila, "Towards rich mobile phone datasets: Lausanne data collection campaign," in *Proceedings of International Conference on Pervasive Services*, 2010.

[68] D. Knossow, A. Sharma, D. Mateus, and R. P. Horaud, "Inexact matching of large and sparse graphs using laplacian eigenvectors," in *Proceedings 7th Workshop on Graph-based Representations in Pattern Recognition*, ser. LNCS 5534. Venice, Italy: Springer, May 2009. [Online]. Available: http://perception.inrialpes.fr/Publications/2009/KSMH09

[69] A. Kovnatsky, M. M. Bronstein, A. M. Bronstein, K. Glashoff, and R. Kimmel, "Coupled quasi-harmonic bases," *Computer Graphics Forum (EUROGRAPHICS)*, vol. 32, no. 2pt4, pp. 439–448, May 2013.

[70] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, Mar 1951.

[71] A. Kumar and H. Daumé III, "A co-training approach for multi-view spectral clustering," in *Proceedings of International Conference on Machine Learning (ICML)*, 2011.

[72] A. Kumar, P. Rai, and H. Daumé III, "Co-regularized spectral clustering with multiple kernels," *NIPS 2010 Workshop: New Directions in Multiple Kernel Learning*, 2010.

[73] ——, "Co-regularized multi-view spectral clustering," in *Advances in Neural Information Processing Systems 24 (NIPS)*, 2011.

[74] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura, "Travel route recommendation using geotags in photo sharing sites," in *Proceedings of ACM International Conference on Information and Knowledge Management*, 2010, pp. 579–588.

[75] R. B. Lehoucq and D. C. Sorensen, "Deflation techniques for an implicitly restarted Arnoldi iteration," *SIAM Journal on Matrix Analysis and Applications*, vol. 17, no. 4, pp. 789–821, Oct 1996.

[76] N. Leonardi and D. V. D. Ville, "Wavelet frames on graphs defined by fMRI functional connectivity," in *Proceedings of 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 2011, pp. 2136–2139.

[77] ——, "Tight wavelet frames on multislice graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3357–3367, Jul 2013.

[78] N. Leonardi, J. Richiardi, M. Gschwind, S. Simioni, J.-M. Annoni, M. Schluep, P. Vuilleumier, and D. V. D. Ville, "Principal components of functional connectivity: A new approach to study dynamic brain connectivity during rest," *NeuroImage*, vol. 83, pp. 937–950, Dec 2013.

[79] M. Levorato, S. K. Narang, U. Mitra, and A. Ortega, "Optimization of wireless networks via graph interpolation," in *Proceedings of the 1st IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013, pp. 483–486.

[80] Z. Li and J. Liu, "Constrained clustering by spectral kernel learning," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2009.

[81] Z. Li, J. Liu, and X. Tang, "Constrained clustering via spectral regularization," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[82] P. Loh and M. J. Wainwright, "Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses," *Annals of Statistics*, vol. 41, no. 6, pp. 3022–3049, 2013.

[83] L. Lovász, "Random walks on graphs: A survey," *Combinatorics, Paul Erdös is Eighty*, vol. 2, pp. 353–398, János Bolyai Mathematical Society, Budapest, 1996.

[84] X. Lu, C. Wang, J.-M. Yang, Y. Pang, and L. Zhang, "Photo2Trip: Generating travel routes from geo-tagged photos for trip planning," in *Proceedings of ACM International Conference on Multimedia*, 2010, pp. 143–152.

[85] Z. Lu and M. Á. Carreira-Perpiñán, "Constrained spectral clustering through affinity propagation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[86] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[87] M. Maggioni, J. C. B. Jr, R. R. Coifman, and A. D. Szlam, "Biorthogonal diffusion wavelets for multiscale representations on manifolds and graphs," in *Proceedings of SPIE Wavelet XI*, vol. 5914, 2005.

[88] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval," *Cambridge University Press*, 2008.

[89] T. Maugey, A. Ortega, and P. Frossard, "Graph-based representation for multiview image coding," in *arXiv:1312.6090*, 2013.

[90] B. McFee and G. Lanckriet, "Learning multimodal similarity," *Journal of Machine Learning Research*, vol. 12, pp. 491–523, 2011.

[91] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Science*, vol. 328, no. 5980, pp. 876–878, May 2010.

[92] P. Muthukrishnan, D. Radev, and Q. Mei, "Edge weight regularization over multiple graphs for similarity learning," in *Proceedings of IEEE International Conference on Data Mining (ICDM)*, 2010.

[93] S. K. Narang and A. Ortega, "Lifting based wavelet transforms on graphs," in *Proceedings of APSIPA Annual Summit and Conference*, 2009, pp. 441–444.

[94] ——, "Perfect reconstruction two-channel wavelet filter-banks for graph structured data," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2786–2799, Jun 2012.

[95] ——, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *arXiv:1210.8129*, 2012.

[96] S. K. Narang, A. Gadde, E. Sanou, and A. Ortega, "Localized iterative methods for interpolation in graph structured data," in *arXiv:1310.2646*, 2013.

[97] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, Sep 2006.

[98] ——, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, vol. 103, no. 23, pp. 8577–8582, Jun 2006.

[99] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14 (NIPS)*, 2001, pp. 849–856.

[100] D. Niu, J. G. Dy, and M. I. Jordan, "Dimensionality reduction for spectral clustering," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011, pp. 552–560.

[101] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas, "Functional maps: A flexible representation of maps between shapes," *ACM Transactions on Graphics (TOG) - SIGGRAPH 2012 Conference Proceedings*, vol. 31, no. 4, Jul 2012.

[102] M. Rabbat, "On the interplay between topology and signals supported on graphs," in *Proceedings of SPIE 8858, Wavelets and Sparsity XV, 88581K*, 2013.

[103] I. Ram, M. Elad, and I. Cohen, "Generalized tree-based wavelet transform," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4199–4209, Sep 2011.

[104] ——, "Redundant wavelets on graphs and high dimensional data clouds," *IEEE Signal Processing Letters*, vol. 19, no. 5, pp. 291–294, May 2012.

[105] S. Roweis, "EM algorithms for PCA and sensible PCA," *CNS Technical Report CNS-TR-97-02, Caltech*, 1997.

[106] H. Rue and L. Held, "Gaussian Markov random fields: Theory and applications," *Chapman and Hall/CRC*, 2005.

[107] R. Rustamov and L. J. Guibas, "Wavelets on graphs via deep learning," in *Advances in Neural Information Processing Systems 26 (NIPS)*, 2013, pp. 998–1006.

[108] M. Saerens, F. Fouss, L. Yen, and P. Dupont, "The principal components analysis of a graph, and its relationships to spectral clustering," in *Proceedings of European Conference on Machine Learning (ECML)*, 2004, pp. 371–383.

[109] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, Apr 2013.

[110] ——, "Classification via regularization on graphs," in *Proceedings of IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013, pp. 495–498.

[111] B. Savas, W. Tang, Z. Lu, and I. S. Dhillon, "Supervised link prediction using multiple sources," in *Proceedings of IEEE International Conference on Data Mining*, 2010.

[112] E. Schaeffer, "Survey: Graph clustering," *Computer Science Review*, vol. 1, no. 1, pp. 27–64, Aug 2007.

[113] A. Sharma, R. P. Horaud, D. Knossow, and E. von Lavante, "Mesh segmentation using laplacian eigenvectors and gaussian mixtures," in *Proceedings of AAAI Fall Symposium on Manifold Learning and its Applications*, ser. Fall Symposium Series Technical Reports. Arlington, VA: AAAI Press, November 2009. [Online]. Available: http://perception.inrialpes.fr/Publications/2009/SHKV09a

[114] G. Shen, W.-S. Kim, S. K. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *Proceedings of Picture Coding Symposium (PCS)*, 2010, pp. 566–569.

[115] G. Shen, W.-S. Kim, A. Ortega, J. Lee, and H. Wey, "Edge-aware intra prediction for depth-map coding," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, 2010, pp. 3393–3396.

[116] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug 2000.

[117] D. I. Shuman, M. J. Faraji, and P. Vandergheynst, "Semi-supervised learning with spectral graph wavelets," in *Proceedings of International Conference on Sampling Theory and Applications (SampTA)*, 2011.

[118] ——, "A framework for multiscale transforms on graphs," in *arXiv:1308.4942*, 2013.

[119] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[120] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," in *arXiv:1307.5708*, 2013.

[121] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, "Spectrum-adapted tight graph wavelet and vertex-frequency frames," in *arXiv:1311.0897*, 2013.

[122] V. Sindhwani and P. Niyogi, "A co-regularization approach to semi-supervised learning with multiple views," in *ICML Workshop on Learning with Multiple Views*, 2005.

[123] A. Smola and R. Kondor, "Kernels and regularization on graphs," in *Proceedings of 16th Annual Conference on Computational Learning Theory*, 2003.

[124] D. C. Sorensen, "Implicit application of polynomial filters in a k-step Arnoldi method," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 1, pp. 357–385, Jan 1992.

[125] G. W. Stewart and J. Sun, "Matrix perturbation theory," *Academic Press, New York*, 1990.

[126] A. Strehl and J. Ghosh, "Cluster ensembles - A knowledge reuse framework for combining multiple partitions," *Journal of Machine Learning Research*, vol. 3, pp. 583–617, Dec 2002.

[127] L. Tang, X. Wang, and H. Liu, "Community detection via heterogeneous interaction analysis," *Data Mining and Knowledge Discovery*, vol. 25, no. 1, pp. 1–33, Jul 2012.

[128] W. Tang, Z. Lu, and I. Dhillon, "Clustering with multiple graphs," in *Proceedings of International Conference on Data Mining (ICDM)*, 2009.

[129] D. Thanou, D. I. Shuman, and P. Frossard, "Learning parametric dictionaries for signals on graphs," in *arXiv:1401.0887*, 2014.

[130] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999.

[131] I. Tosic and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, Mar 2011.

[132] N. Tremblay and P. Borgnat, "Multiscale community mining in networks using spectral graph wavelets," *arXiv:1212.0689*, 2012.

[133] V. Vasuki, N. Natarajan, Z. Lu, and I. S. Dhillon, "Affiliation recommendation using auxiliary networks," in *Proceedings of the 4th ACM Conference on Recommender Systems(RecSys)*, Sep 2010.

[134] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec 2007.

[135] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2010, pp. 563–572.

[136] X. Wang, W. Bian, and D. Tao, "Grassmannian regularized structured multi-view embedding for image classification," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2646–60, Jul 2013.

[137] T. Xia, D. Tao, T. Mei, and Y. Zhang, "Multiview spectral embedding," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 40, no. 6, pp. 1438–1446, Dec 2010.

[138] Q. Xu, M. desJardins, and K. Wagstaff, "Constrained spectral clustering under a local proximity structure," in *Proceedings of the 18th International Florida Artificial Intelligence Research Society (FLAIRS) Conference*, May 2005.

[139] C. Zhang and D. Florencio, "Analyzing the optimality of predictive transform coding using graph-based models," *IEEE Signal Processing Letters*, vol. 20, no. 1, pp. 106–109, Jan 2013.

[140] X. Zhang, X. Dong, and P. Frossard, "Learning of structured graph dictionaries," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[141] Z. Zhang and M. Jordan, "Multiway spectral clustering: A margin based perspective," *Statistical Science*, vol. 23, no. 3, pp. 383–403, 2008.

[142] Y.-T. Zheng, Z.-J. Zha, and T.-S. Chua, "Mining travel patterns from geotagged photos," *ACM Transactions on Intelligent Systems and Technology*, vol. 3, no. 3, May 2012.

[143] D. Zhou and C. Burges, "Spectral clustering and transductive learning with multiple views," in *Proceedings of International Conference on Machine Learning (ICML)*, 2007.

[144] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields*, 2004.

[145] ——, "Regularization on discrete spaces," in *Proceedings of the 27th DAGM conference on Pattern Recognition*, 2003, pp. 361–368.

[146] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems 16 (NIPS)*, 2003, pp. 321–328.

[147] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf, "Ranking on data manifolds," in *Advances in Neural Information Processing Systems 16 (NIPS)*, 2003, pp. 169–176.

[148] X. Zhu, "Semi-supervised learning with graphs," *PhD thesis, Carnegie Mellon University, CMU-LTI-05-192*, 2005.

[149] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 3921–3924.

[150] ——, "Graph spectral compressed sensing for sensor networks," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 2865–2868.

[151] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*, 2003.

Xiaowen Dong

Curriculum Vitae

EPFL-STI-IEL-LTS4, ELD 241, Station 11
CH-1015 Lausanne, Switzerland
☎ *+41 21 693 26 05*
✉ *xiaowen.dong@epfl.ch*
http://lts4.epfl.ch/xdong/

Research Interests

- Signal processing on graphs: Dictionary designs and learning algorithms for graph signals
- Learning theory and algorithms: Supervised and unsupervised learning, kernel methods
- Applications: Social network, social media and urban dynamics analysis

Education

2009.10–2014.05 **Ph.D.**, *École Polytechnique Fédérale de Lausanne (EPFL)*, Signal Processing.
Advisors: Pascal Frossard and Pierre Vandergheynst

2007.09–2008.08 **M.Sc.(Distinction)**, *The University of Edinburgh*, Signal Processing and Communications.
Advisor: Pei-Jung Chung

2000.09–2004.06 **B.Eng.**, *Zhejiang University*, Mixed Class, Information Engineering.

Work Experiences

2009.10–Present **Research and Teaching Assistant**, *Signal Processing Laboratories (LTS4/LTS2), EPFL.*
Research novel graph-based signal processing and learning algorithms, with applications in mobile and social networks and social media analysis

- Developed two frameworks for clustering vertices in multi-layer graphs
- Developed a framework for classification and clustering of graph signals
- Developed a dictionary learning method for approximation of graph signals

Help prepare teaching materials for courses and supervise thesis works

- TA for Digital Signal Processing (EE-351): Spring 2010/2011/2012
- Thesis supervision: one Master's thesis project and three semester projects

2013.05–2013.08 **Research Intern**, *Smarter Urban Dynamics Group, IBM Research - Ireland.*
Manager: Francesco Calabrese, Mentor: Michele Berlingerio

Research novel data mining methods in the context of smarter cities applications at IBM's Smarter Cities Technology Centre

- Developed theoretical frameworks for multiscale event detection in social media platforms
- Contributed to "SaferCity", a system to identify public safety related incidents from social media

2009.03–2009.09 **Engineering Assistant**, *Edinburgh Project Office, Siemens UK.*

2005.05–2007.08 **Engineer and Project Team Leader**, *Hangzhou Branch, Motorola (China).*

Awards

2014 Chinese Government Award for Outstanding Self-Financed Students Abroad
(around 500 worldwide and 10 in Switzerland per annum)

2008 Signal Processing Class Medal, The University of Edinburgh
(for the best overall performance in the M.Sc. class)

Grants and Fellowships

2012 and 2014 Contributed to two grant proposals for projects on signal processing on graphs
(awarded CHF 100,000 by Hasler Foundation, Switzerland)

2009 Trinity Overseas Bursaries, Trinity College, Cambridge
(11 worldwide per annum, declined for studies at EPFL)

Patents

- Michele Berlingerio, Xiaowen Dong, Aris Gkoulalas-Divanis and Dimitrios Mavroeidis, "A System for Identifying, Monitoring and Ranking Incidents from Social Media", US Patent: US 14/143949, filed.

Publications

Journal Papers

- **Xiaowen Dong**, Dimitrios Mavroeidis, Francesco Calabrese and Pascal Frossard, "Multiscale Event Detection in Social Media," submitted to Data Mining and Knowledge Discovery, April 2014.
- **Xiaowen Dong**, Pascal Frossard, Pierre Vandergheynst and Nikolai Nefedov, "Clustering on Multi-Layer Graphs via Subspace Analysis on Grassmann Manifold," *IEEE Transactions on Signal Processing*, vol. 62, no. 4, pp. 905-918, February 2014.
- **Xiaowen Dong**, Pascal Frossard, Pierre Vandergheynst and Nikolai Nefedov, "Clustering with Multi-Layer Graphs: A Spectral Perspective," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5820-5831, November 2012.
- Zhe Wang, Kai Hu, Ke Xu, Baolin Yin and **Xiaowen Dong**, "Structural Analysis of Network Traffic Matrix via Relaxed Principal Component Pursuit," *Computer Networks (Elsevier)*, vol. 56, no. 7, pp. 2049-2067, May 2012.

In Preparation

- **Xiaowen Dong**, Antonio Ortega, Pascal Frossard and Pierre Vandergheynst, "Wavelet-Based Classification of Graph Signals and Applications in Mobility Inference."

Conference Papers

- Michele Berlingerio, Francesco Calabrese, Giusy Di Lorenzo, **Xiaowen Dong**, Yiannis Gkoufas and Dimitrios Mavroeidis (alphabetical order), "SaferCity: a System for Detecting Incidents from Social Media," *Demo paper, IEEE International Conference on Data Mining (ICDM)*, Dallas, TX, USA, December 2013.
- **Xiaowen Dong**, Pascal Frossard, Pierre Vandergheynst and Nikolai Nefedov, "Clustering on Multi-Layer Graphs via Subspace Analysis on Grassmann Manifolds," *Invited paper, IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Austin, TX, USA, December 2013.
- **Xiaowen Dong**, Antonio Ortega, Pascal Frossard and Pierre Vandergheynst, "Inference of Mobility Patterns via Spectral Graph Wavelets," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Vancouver, Canada, May 2013.
- Xuan Zhang, **Xiaowen Dong** and Pascal Frossard, "Learning of Structured Graph Dictionaries," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, March 2012.
- **Xiaowen Dong**, Pascal Frossard, Pierre Vandergheynst and Nikolai Nefedov, "Methods for Clustering Multi-Layer Graphs in Mobile Networks," *Interdisciplinary Workshop on Information and Decision in Social Networks (WIDS)*, MIT, Cambridge, MA, USA, May-June 2011.
- **Xiaowen Dong**, Pascal Frossard, Pierre Vandergheynst and Nikolai Nefedov, "A Regularization Framework for Mobile Social Network Analysis," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, Czech Republic, May 2011.

Academic Services

Reviewer IEEE Transactions on Signal Processing

Skills

Computer

- Programming: MATLAB, Python
- Document Markup: HTML, LaTeX
- Operating System: Mac OS X, Microsoft Windows, UNIX

Language Chinese (native), English (fluent), French (beginner), Spanish (beginner)