UNIVERSITÄT
DES
SAARLANDES

Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science

Graph-based methods for unsupervised and semi-supervised data analysis

Syama Sundar Rangapuram

Dissertation

zur Erlangung des Grades
des Doktors der Naturwissenschaften (Dr. rer. nat.)
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes

Saarbrücken, March 2016

Tag des Kolloquiums:
10.10.2016

Dekan:
Prof. Dr. Frank-Olaf Schreyer

Vorsitzender des Prüfungsausschusses:
Prof. Dr. Joachim Weickert
Universität des Saarlandes

1. Gutachter:
Prof. Dr. Matthias Hein
Universität des Saarlandes

2. Gutachter:
Prof. Dr. Le Thi Hoai An
Université de Lorraine

Akademischer Mitarbeiter:
Dr. Karteek Sreenivasaiah
Max-Planck-Institut für Informatik

Abstract

Clustering and community detection are two important problems in data analysis with applications in various disciplines. Often in practice, there exists prior knowledge that helps the process of data analysis. In this thesis we develop generic graph-based methods for these data analysis problems both in unsupervised and semi-supervised settings. The main advantage of our methods is that they provide a common framework for integrating soft as well as hard prior knowledge. In the latter case, ours is the first method to have provable guarantees on the satisfaction of the given prior knowledge. The foundation of our methods is the exact continuous relaxation result that we derive for a class of combinatorial optimization problems. More specifically, we show that the (constrained) minimization of a ratio of set functions can be equivalently rewritten as a continuous optimization problem. We also present efficient algorithms for solving the continuous relaxations. While the global optimality is not guaranteed, in practice our methods consistently outperform the corresponding convex or spectral relaxations by a large margin. Moreover, our method has an additional guarantee that the solution respects the prior knowledge.

Zusammenfassung

Clustering und Community Detection sind zwei bedeutende Probleme in der Datenanalyse, mit vielfältigen Anwendungen in unterschiedlichen Bereichen. In der Praxis existiert häufig Vorwissen das in den Prozess der Datenanalyse einfließen kann. In dieser Arbeit entwickeln wir generische Graph-basierte Methoden für diese Problemstellungen der Datenanalyse, sowohl für den unüberwachten als auch den teilüberwachten Fall. Der Hauptvorteil unserer Verfahren ist dass sie ein allgemeines Framework zur Integration von weichen und harten Nebenbedingungen bereitstellen. In letzterem Fall ist unsere Methode die erste die beweisbare Garantien zur Erfüllung des gegebenen Vorwissen liefern kann. Die Grundlage unserer Methoden ist ein Resultat über exakte kontinuierliche Relaxierungen das wir für eine Klasse von kombinatorischen Optimierungsproblemen herleiten. Konkret zeigen wir dass die (beschränkte) Minimierung eines Bruches von Mengenfunktionen in ein äquivalentes kontinuierliches Optimierungsproblem umgeformt werden kann. Des Weiteren präsentieren wir effiziente Algorithmen zur Lösung der kontinuierlichen Relaxierungen. Während die globale Optimalität nicht garantiert werden kann, werden die entsprechenden konvexen oder spektralen Relaxierungen in der Praxis mit einem deutlichen Vorsprung übertroffen. Darüber hinaus hat unsere Methode eine zusätzliche Garantie dass die berechnete Lösung das Vorwissen stets berücksichtigt.

Acknowledgments

First and foremost I would like to thank my advisor Matthias Hein for giving me the opportunity to do my doctoral thesis with him. It has been a great learning experience for me which I thoroughly enjoyed. I really appreciate the valuable insights and time he has provided in all the projects. I am also thankful to him for allowing me to attend summer school and conferences.

Next, I would like to thank Prof. Le Thi Hoai An for agreeing to review my thesis. I also would like to thank International Max Planck Research School for Computer Science for funding part of my Ph.D. work.

Special thanks to Thomas Bühler for being a wonderful officemate and co-author. I thank him for all the valuable discussions we had and for being always available for a quick help in professional as well as personal matters.

I thank Simon Setzer for the inspiring discussions we had during our collaboration and I especially appreciate his help in solving optimization problems. I also thank Pramod Kaushik Mudrakarta for the pleasant time we had during our collaboration.

I thank Martin Slawski for being an excellent colleague and Pratik Kumar Jawanpuria for proofreading parts of my thesis. I also thank former and present members of the Machine Learning group for a nice working atmosphere. Special thanks go to Irmtraud Stein and Dagmar Glaser for their help in administrative tasks.

I thank my friends Shiva Lingam, Sairam Gurajada, Srikanth Duddela, Satish Pammi, Harish Bokkasam for the fun and support they provided. I also thank Indian community in Saarbrücken for all the wonderful parties and the nice time we had here.

Finally I would like to thank my family for always allowing me to pursue my goals and my wife Meena for her understanding and constant support during the work on this thesis.

Contents

Chapter 1

Introduction

Cluster analysis, an instance of unsupervised learning, is the problem of grouping a given set of objects into *clusters* based on *similarity* (or dissimilarity). Here cluster refers to a set of points that are more closely related to each other than the rest of the points. Clustering has applications in a variety of disciplines including bioinformatics [106], computer vision [62] and consequently different clustering methods have been developed over the years [83, 52, 108, 46, 34]. These clustering algorithms differ by their choice of the objective that measures the quality of a clustering and the similarity function. Most of the algorithms can directly work with the given pairwise similarities thus avoiding the explicit feature representation of the data. Some algorithms assume that the number of required clusters k is given as input while others like hierarchical clustering outputs nested clusterings for all possible values of k. In this thesis, we assume that the number of clusters k is fixed and is provided as an input. Moreover, we assume that the clustering is a partitioning of the given objects; i.e., each object is assigned to exactly to one cluster.

It has been shown that incorporating the available prior knowledge improves the performance of clustering methods [116, 117]. The prior knowledge for clustering varies from instance-level (e.g., must-link or cannot-link constraints) to the group level (e.g., bound on the size of each cluster) [14]. Since acquiring constraint information is relatively cheaper, constrained clustering has been an active area of research [14].

Community detection is a related problem where the goal is to find a single well-connected component in a given graph. The problem of finding dense regions in graphs arises in several applications e.g., in bioinformatics [106, 111, 12, 60, 102] and network analysis [72, 40, 47]. In real world applications one often needs to integrate priors based on the size or the locality of the subset being extracted [102, 47].

We now describe in some detail the graph-based setting for unsupervised learning, in particular highlighting the options available in terms of graph construction from the given pairwise similarities and the criteria used thereafter for the data analysis.

1.1 Graph-based setting

Given a set of objects $x_1, \ldots, x_n$ along with pairwise similarities $\kappa(x_i, x_j) \geq 0, \forall i, j = 1, \ldots, n$, where κ is symmetric, a natural way to represent the relationships (or the

global structure) in the data is via graphs. A graph $G(V, E)$ is defined by a set of vertices $V = \{1, \ldots, n\}$ and a set of edges $E = \{(i, j) : i, j \in V\}$ encoding the pairwise relationship between the vertices. In this thesis we assume that all the graphs are undirected; i.e., $(i, j) \in E \implies (j, i) \in E$. From the given pairwise similarities of objects, one can build a *similarity* graph by forming a vertex i for each data point x_i and connecting two vertices i and j by an edge (i, j) depending on the similarity between x_i and x_j. Moreover the edges can be given weights w_{ij} indicating the amount of similarity. For simplicity, we often use the terms objects and vertices interchangeably.

1.1.1 Graph construction

A straightforward way to build a similarity graph is to connect each pair of vertices i and j by an edge if $\kappa(x_i, x_j) > 0$. This leads, depending on the similarity κ, to a densely connected graph making it computationally expensive to work with. Moreover, unless κ models the local neighborhoods well, such a construction can obscure the global structure inherent in the data. On the other hand, *neighborhood* graphs, which are built using the local neighborhood information, are more likely to uncover the overall structure in the data. Moreover, neighborhood graphs are usually sparse. The most widely used neighborhood graphs are ϵ-neighborhood graphs and K-nearest neighbor graphs.

K-nearest neighbor graphs: Here a vertex i is connected to a vertex j if j is one of the K-nearest neighbors of i, where the nearest neighbor is defined based on the similarity κ. Note the adaptive nature of this procedure to the local neighborhood: high similarity between x_i and x_j does not imply an edge (i, j) (e.g., x_i belongs to a high density region and has more similar objects in its neighborhood) and an edge (i, j) does not necessarily mean x_i and x_j are highly similar (e.g., outliers, even though highly dissimilar from the rest of the objects, are still connected to K vertices). Thus, it makes sense here to assign weights to the edges according to the similarity.

Note that the neighborhood relation defined above is not symmetric although the original similarities are symmetric; a vertex j can be among the K nearest neighbors of i while j may have K points closer (or more similar) to it than i. Thus, this construction leads to a directed graph. In practice, one of the following two approaches are followed to maintain the symmetric relation and thus making the graph undirected. The first one, where the direction of the edges is ignored, corresponds to connecting two vertices if either one is among the K-nearest neighbors of the other. Graph constructed in this way is known as symmetric KNN graph. The secoondnd one imposes a stronger restriction and often leads to disconnected graph; here two vertices are connected if each of them is among the K-nearest neighbors of the other resulting in the so-called mutual KNN graph. Since outliers are typically not among the K nearest neighbors of other vertices, they are likely to be disconnected in a mutual KNN graph.

ϵ-neighborhood graphs: These graphs are typically constructed using pairwise *distances*. All pairs of vertices whose distances are below a given threshold ϵ are

connected by an edge. Note that this construction leads to an undirected graph. Both unweighted as well as weighted versions of ϵ-neighborhood graphs are used in machine learning. In the latter case, the weights are constructed from the pairwise similarities of the points, which are derived from the given pairwise distances. We note here that ϵ-neighborhood graphs can also be constructed from pairwise similarities.

1.1.2 Clustering based on balanced cuts

Once the similarity graph is constructed, the clustering problem can be transformed to a graph partitioning problem: divide the graph into k components such that the total edge weight is large within each component and small across the components. A natural way of partitioning the graph is by minimizing the cut, the total weight of the edges that separate different components. However, since the cut value increases with the number of edges, minimizing the cut often results in components containing single vertices that are isolated from the rest of the graph. To avoid this unwanted bias towards clusters of small size, different measures were proposed based on the cut as well as on the "size" of clusters [52, 108]. A widely used "balanced" cut criteria has the form

$$\min_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C}_l)}{\hat{S}(C_l)} \tag{1.1}$$

where P_k is the set of all k-partitions of V, $\overline{C} = V \backslash C$, $\mathrm{cut}(C, \overline{C}) = \sum_{i\in C,\ j\in\overline{C}} w_{ij}$ and $\hat{S}(C) : 2^V \to \mathbb{R}_+$ is a balancing function. Two most popular criteria are ratio cut where $\hat{S}(C) = |C|$, the size of the component C and normalized cut where $\hat{S}(C) = \mathrm{vol}(C)$, the volume of the component C. The volume of C is defined as sum of the degrees of the vertices in C, $\mathrm{vol}(C) = \sum_{i\in C} d_i$, and $d_i = \sum_{j=1}^{n} w_{ij}$ is the degree of vertex i. Ratio cut is a good model if final clusters have to be of the same size, e.g., applications in parallel computing [52]. Normalized cuts are most popular in image processing and computer vision applications [108, 125, 86] as they indirectly maximize the similarity of points within each cluster apart from minimizing the similarity across the clusters. This can be seen by rewriting the normalized cut as

$$\sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C}_l)}{\mathrm{vol}(C_l)} = \sum_{l=1}^{k} \frac{\mathrm{vol}(C_l) - \mathrm{assoc}(C_l)}{\mathrm{vol}(C_l)} = k - \sum_{l=1}^{k} \frac{\mathrm{assoc}(C_l)}{\mathrm{vol}(C_l)},$$

where assoc is the association of a set defined as $\mathrm{assoc}(C) = \sum_{i,j\in C} w_{ij}$, which measures the similarity with in the set C. Thus minimizing the normalized cut across the clusters can be seen as maximizing the normalized association of each cluster. Two variants that enforce balance more strongly are the so-called ratio Cheeger cut and normalized Cheeger cut. The balancing functions in these cases are given respectively by $\min\{|C|, \left|\overline{C}\right|\}$ and $\min\{\mathrm{vol}(C), \mathrm{vol}(\overline{C})\}$.

It has been shown that the normalized cut and ratio cut problems are NP-hard [108, 16] and consequently one has to use approximation or relaxation algorithms in practice. Spectral clustering [108, 92, 115] is based on solving a continuous relaxation of the ratio/normalized cut that leads to an eigenvalue problem, which

can be optimally solved. However, one has to rely on heuristic procedures to obtain a clustering from the continuous solution of the spectral relaxation. Moreover, the spectral relaxation is very loose [51] and often greedy approaches [31] outperform spectral clustering. Furthermore, it is very hard to enforce any prior knowledge in the spectral clustering framework. In this thesis, we develop methods for solving the balanced cut problem (1.1) for a generic balancing function. Moreover, we show how prior information can be efficiently incorporated in clustering. In fact, our constrained clustering method is the first to guarantee that the solution respects the given prior knowledge in a hard-enforcement setting.

1.1.3 Community detection based on densest subgraphs

In the case of community detection, a natural graph-theoretic criteria is the density of a subset, which is defined as the ratio of total edge weight and the "size" of the subset. Then the community detection problem can be formulated as finding a densest subgraph

$$\max_{C \subseteq V} \frac{\text{assoc}(C)}{|C|}, \tag{1.2}$$

where $\text{assoc}(C) = \text{vol}(C) - \text{cut}(C, V \backslash C)$, and cut, vol are the volume and cut functions as defined previously. Although, the maximum densest subgraph problem can be optimally solved in polynomial time [49], incorporating intuitive prior information such as restricting the size of the subset makes the problem NP-hard [45, 4, 68]. It has been shown in [68] that for the densest subgraph problem with a lower bound constraint, i.e., $|C| \geq k$, there exists a polynomial time algorithm achieving an approximation guarantee of 2. For equality constraint, the best known approximation algorithm has an approximation ratio of $O(V^{\delta})$, for some $\delta < \frac{1}{3}$ [45]. It is also shown that the densest subgraph problem with upper bound and equality constraints are equally hard [68]. Up to our knowledge there is no method that can handle both lower and upper bound constraints simultaneously which for example arise in some applications [95]. In this thesis we develop a method for a very generic version of the densest subgraph problem allowing any type of size constraints as well as application specific priors.

1.2 Incorporating prior knowledge

An important form of prior information in clustering comes via must-link and cannot-link constraints. A must-link constraint (p, q) indicates that the objects p and q should be assigned to the same cluster, while a cannot-link constraint (p, q) indicates that p and q should belong to different clusters. Depending on the application, there are other kinds of supervision available. For example, in image segmentation application, the supervision can be partial labeling of the pixels (background or object of a particular type) [18]. In the one-class setting (i.e., densest subgraph problem) a more relevant prior is a bound on the size of the cluster [86, 85, 47].

Depending on the application one may have to provide a solution that satisfies all constraints (hard-enforcement) or allow for the noise or inconsistency in the

prior knowledge (soft-enforcement). In the case of labels in image segmentation or size constraints in the case of densest subgraph problem, one can argue in favor of satisfying all the constraints. In contrast, it is not clear if one should always enforce pairwise constraints in clustering as they might be inconsistent in the worst case, unlike label or size constraints. On the other hand preliminary work in constrained clustering shows that satisfying pairwise constraints actually improves the performance [116, 117]. Moreover, there exist problem settings such as alternative clustering [13] where it maybe necessary to satisfy the constraints in order to find a clustering different from the given clustering. Thus, ideally one would prefer a method that can handle both situations in a common framework and this is precisely the approach we take in this thesis.

Fundamentally there are two different approaches in incorporating the constraints in clustering. One line of work attempts to directly modify [64] or learn [119] the similarity metric between the objects using the given constraints and then apply a standard (constrained or unconstrained) clustering method on the learned metric. The other approach is to modify the clustering problem by explicitly encoding the prior information as side constraints [116, 121, 117]. In the latter case, there are methods that encode priors as hard constraints [121] while others require the solution to satisfy a certain amount of prior information [117]. In this thesis we take the latter approach and address both soft and hard enforcement settings in a single framework.

1.3 Overview of the thesis

In Chapter 2 we first present the necessary mathematical background and then discuss the existing methods for clustering highlighting the connections between different approaches. Chapter 3 presents our work in the area of constrained clustering for the two-class setting. We address the multi-class setting in Chapter 4 where we present both unconstrained and constrained clustering methods for the generic case ($k > 2$). We consider the one-class setting in Chapter 5 where we discuss our new method for solving a generic version of the densest subgraph problem.

1.3.1 Contributions of the thesis

Here is the summary of the contributions of the thesis.

1. **Two-class setting:** In Chapter 3, we derive a novel formulation for the constrained clustering problem in the two-class setting, that allows minimization of a trade-off between constraint violation and the clustering objective (balanced cut in our case). Unlike the existing work, we show that the proposed framework can handle both soft and hard constraints. We derive exact continuous relaxation result for the new formulation and show that the resulting non-smooth problem can be efficiently solved. We also present a new preconditioning technique for a generic graph-based problem that is of independent interest. In contrast to the existing work, our method finds a solution that is guaranteed to satisfy all the constraints in the hard-enforcement setting.

2. **Multi-class setting:** In Chapter 4, we present a new cnvontinuous relaxation to directly address the multi-class clustering. It turns out that enforcing the partitioning constraint is extremely difficult for the multi-class problem and as a result many methods [34, 124, 123, 19] even fail to produce a k-way clustering. In contrast, we develop a generic method that allows to minimize any application specific balancing criteria apart from the standard balanced cuts such as ratio and normalized cuts. Another contribution of this chapter is a monotonic descent method for solving a very difficult sum-of-ratios minimization problem.

   Although exact continuous relaxations similar to the two-class case could not be obtained here, we show that our method makes the integration of hard-prior information possible for the first time in the multi-class setting. Similarly to the two-class case, we develop a constrained clustering method that can guarantee a solution satisfying all the constraints (as long as *any* consistent partition can be obtained efficiently) while still allowing for soft enforcement of noisy priors. Note that none of the existing methods in the multi-class setting have such a guarantee even for simple label constraints nor do they have any control on the number of constraints violated.

3. **One-class setting:** In Chapter 5, we consider a generic version of the densest subgraph problem that models a variety of applications in bioinformatics [106, 111, 12, 60, 102] and social network analysis [72, 47]. We show that there exists an exact continuous relaxation for the generic version of the densest subgraph problem considered and present an efficient method for solving the resulting continuous problem. As an application, we discuss the team formation problem in the context of social networks. The main feature of our method is the guarantee that our solution satisfies all the given constraints which is a necessary condition for the team formation problem.

Chapter 2

Background

In the first part of this chapter we present the necessary mathematical background required for understanding the material presented later in the thesis. In the second part we describe various methods for clustering including graph-based approaches and the recent developments based on non-negative matrix factorization.

2.1 Mathematical background

In this section we describe set functions and the so-called Lovász extensions which play an important role in deriving the exact continuous relaxations of a class of combinatorial problems considered in this thesis. We present known exact continuous relaxations and recent methods for solving them. As we will see later, these exact continuous relaxations result in optimization problems where one needs to minimize a ratio of a certain class of difference of convex functions. So, we also discuss related optimization methods from fractional programming as well as difference of convex programming. We begin with a review of basic concepts from analysis.

2.1.1 Basics from analysis

Convex functions play an important role in mathematical optimization especially because of their property that every local minimum is a global minimum. Many algorithms exist [17] for solving convex optimization problems where the goal is to minimize a convex function over a convex set. Moreover, convex optimization is a key ingredient in some of the methods designed for solving non-convex problems. Here we first gather some basic definitions from convex analysis [98].

Definition 2.1 (Convex set) *A set $D \subseteq \mathbb{R}^n$ is a convex set if for all $f, g \in D$ and $\alpha \in [0, 1]$, it holds that $\alpha f + (1 - \alpha)g \in D$.*

Definition 2.2 (Convex function) *A function $R : D \to \mathbb{R}$ is a convex function if its domain $D \subseteq \mathbb{R}^n$ is a convex set and for all $f, g \in D$, and $\alpha \in [0, 1]$ it holds that*

$$R(\alpha f + (1 - \alpha)g) \leq \alpha R(f) + (1 - \alpha)R(g).$$

Definition 2.3 (ρ-convexity) *A function $R : \mathbb{R}^n \to \mathbb{R}$ is called ρ-convex for some $\rho \geq 0$ on a convex set D if for all $f, g \in D$ and all $\alpha \in [0, 1]$, it holds that*

$$R(\alpha f + (1 - \alpha)g) \leq \alpha R(f) + (1 - \alpha)R(g) - \frac{\rho}{2}\alpha(1 - \alpha)\left\|f - g\right\|_2^2.$$

We denote the supremum of all $\rho \geq 0$ satisfying the above inequality by $\rho_D(R)$. We use the simplified notation $\rho(R)$ if $D = \mathbb{R}^n$. R is said to be strongly convex on D with parameter $\rho_D(R)$ if $\rho_D(R) > 0$.

The continuous extensions of set functions that we later consider are in general not differentiable. We now define the subdifferential of a convex function which is a generalization of the gradient to the non-differentiable case.

Definition 2.4 (Subdifferential) *Let $R : \mathbb{R}^n \to \mathbb{R}$ be a convex function. The subdifferential of R at a given point $f \in \mathbb{R}^n$ is defined as*

$$\partial R(f) = \{r \in \mathbb{R}^n : R(g) \geq R(f) + \langle r, g - f \rangle, \forall g \in \mathbb{R}^n\}.$$

An element $r \in \partial R(f)$ is called a subgradient of R at f.

Definition 2.5 (Normal cone) *The normal cone of a non-empty convex set $D \subseteq \mathbb{R}^n$ at a given point $f \in D$ is defined as*

$$N_D(f) = \{r \mid \langle r, g - f \rangle \leq 0, \forall g \in D\}.$$

Let $D \subseteq \mathbb{R}^n$ be a non-empty convex set and ι_D denote the indicator function of D defined as $\iota_D(f) = 0$, if $f \in D$, and ∞ otherwise. It is easy to deduce from the definition that the subdifferential of ι_D at $f \in D$ is given by $N_D(f)$.

Another class of functions considered in this thesis are positively p-homogeneous functions which are defined as follows.

Definition 2.6 (p-homogeneity) *A function $R : \mathbb{R}^n \to \mathbb{R}$ is called positively p-homogeneous if $R(\alpha f) = \alpha^p R(f)$, $\forall \alpha \geq 0$.*

The relation between convex p-homogeneous functions ($p > 0$) and their subdifferentials is given by the generalized Euler's identity [122].

Lemma 2.1 *Let $R : \mathbb{R}^n \to \mathbb{R}$ be a continuous, convex and positively p-homogeneous function. Then for each $f \in \mathbb{R}^n$, $r \in \partial R(f)$ and $p > 0$, it holds that $\langle r, f \rangle = pR(f)$.*

We close this section by introducing the notion of subdifferential for non-smooth, non-convex functions. Recall that Definition 2.4 is valid only for convex functions. Clarke's subdifferential or the generalized gradient is a further generalization of subdifferentials to non-convex functions which are locally Lipschitz continuous [27].

Definition 2.7 (Lipschitz continuity) *A function $R : \mathbb{R}^n \to \mathbb{R}$ is L-Lipschitz continuous if $\exists L > 0$ such that for all $f, g \in \mathbb{R}^n$ the following holds*

$$\|R(f) - R(g)\|_2 \leq L \|f - g\|_2 .$$

A function is called locally Lipschitz continuous if for every $f \in \mathbb{R}^n$ there exists a neighborhood U of f such that R restricted to U is Lipschitz continuous.

Definition 2.8 (Clarke's directional derivative, Clarke's subdifferential) *Let $R : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$ be a locally Lipschitz continuous function at $f \in \mathbb{R}^n$. The Clarke's directional derivative and the Clarke's subdifferential [27] of R at f are defined as*

$$R^{\uparrow}(f, v) = \lim_{g \to f,\ t \to 0} \sup \frac{R(g + tv) - R(g)}{t},$$

$$\partial_C R(f) = \{\xi \in \mathbb{R}^n \mid R^{\uparrow}(f, v) \geq \langle \xi, v \rangle ,\ \forall v \in \mathbb{R}^n\}.$$

If R is convex then the Clarke's subdifferential is same as the subdifferential defined above for a convex function. Moreover, the Clarke's subdifferential reduces to the usual gradient if R is differentiable.

Recall that a critical point or stationary point of a differentiable function is any point in its domain where all the partial derivatives are zero. One can extends this notion for general non-smooth, non-convex functions using the Clarke's subdifferential [25].

Definition 2.9 (Critical point) *A point $f \in \mathbb{R}^n$ is called critical point of the function $R : \mathbb{R}^n \to \mathbb{R}$ if $0 \in \partial_C R(f)$.*

2.1.2 Set functions and Lovász extensions

Given an n-element ground set $V = \{1, \ldots, n\}$, a set function $\hat{R} : 2^V \to \mathbb{R}$ is a function defined on the subsets of V. One can view the set functions as functions on the vertices (or the extreme points) of the unit hypercube in $\mathbb{R}^n$, since each subset $C \subseteq V$ can be canonically identified with vertices of $[0,1]^n$. More precisely, any set $C \subseteq V$ can be represented by an indicator vector $\mathbf{1}_C \in \{0,1\}^n$, whose i^{th} entry is 1 if $i \in C$ and 0 otherwise. Lovász extension [82], which allows the extension of set functions from $\{0,1\}^n$ to the entire space $\mathbb{R}^n$, plays a key role in deriving the exact continuous relaxations of combinatorial optimization problems ([56, 22]).

Definition 2.10 (Lovász extension) *Let $\hat{R}$ be a set function with $\hat{R}(\emptyset) = 0$. Given $f \in \mathbb{R}^n$, let $(j_1, \ldots, j_n)$ be a permutation of V satisfying $f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_n}$. Then the function $R : \mathbb{R}^n \to \mathbb{R}$ defined in the following equivalent ways is called the Lovász extension of $\hat{R}$*

$$R(f) = \sum_{i=1}^{n} f_{j_i} \left(\hat{R}\left(\{j_i, \ldots, j_n\}\right) - \hat{R}\left(\{j_{i+1}, \ldots, j_n\}\right) \right) \tag{2.1}$$

$$= \sum_{i=1}^{n-1} \hat{R}(\{j_{i+1}, \ldots, j_n\}) \left(f_{j_{i+1}} - f_{j_i}\right) + f_{j_1} \hat{R}(V). \tag{2.2}$$

First note that the set function and its Lovász extension agree on the vertices of the hypercube, i.e., $\hat{R}(C) = R(\mathbf{1}_C), \forall C \subseteq V$. Next, on the unit hypercube $[0,1]^n$, one can interpret the Lovász extension as the linear interpolation of the values of the set function at the vertices of $n!$ simplices that make up the set $[0,1]^n$ [10]. This can be seen as follows. For an arbitrary permutation $\{j_1, \ldots, j_n\}$ of V, one can deduce that the polytope $\{f \in [0,1]^n, f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_n}\}$ is a convex hull of the $n+1$ vertices $\mathbf{1}_\emptyset, \mathbf{1}_{\{j_1\}}, \ldots, \mathbf{1}_{\{j_1, \ldots, j_n\}}$ (and hence a simplex). In fact, any f in this polytope can be written as the convex combination of these vertices

$$f = \sum_{i=1}^{n-1} (f_{j_{i+1}} - f_{j_i}) \mathbf{1}_{\{j_{i+1}, \ldots j_n\}} + f_{j_1} \mathbf{1}_V + (1 - f_{j_n}) \mathbf{1}_\emptyset.$$

Comparing this with the definition (2.2), one sees that the Lovász extension is in fact a linear interpolation of the values of the set function at these vertices. Note that there are $n!$ possible orderings of V and hence any f in the hypercube belongs to at least one of these simplices (exactly one simplex, if the elements of f are

unique) and can be similarly written as a convex combination of corresponding vertices.

There are several other equivalent definitions of the Lovász extension [10]. In this thesis, we use the following definition expressed in terms of the sets obtained by thresholding. We define the sets

$$C_0 = V,\ C_i = \{j \in V | f_j > f_i\},\ i = 1, \ldots, n. \tag{2.3}$$

Note that these sets are not same as the sets used in the definitions of (2.1) and (2.2) if the components of f are not unique. To make the notation simpler in the following definition, we assume without loss of generality that components of f are ordered in increasing order $f_1 \leq f_2 \leq \ldots \leq f_n$.

Definition 2.11 (Lovász extension) *Let $\hat{R}$ be a set function with $\hat{R}(\emptyset) = 0$ and the sets C_i be defined as in (2.3). Let $f \in \mathbb{R}^V$ be ordered in increasing order $f_1 \leq f_2 \leq \ldots \leq f_n$. Then the Lovász extension $R : \mathbb{R}^n \to \mathbb{R}$ of $\hat{R}$ is defined as*

$$R(f) = \sum_{i=1}^{n} f_i \left(\hat{R}(C_{i-1}) - \hat{R}(C_i) \right) \tag{2.4}$$

$$= \sum_{i=1}^{n-1} \hat{R}(C_i) \left(f_{i+1} - f_i \right) + f_1 \hat{R}(V) \tag{2.5}$$

It is easy to check that these definitions are equivalent to (2.1) and (2.2). Again, on the unit hypercube $[0, 1]^n$, one can interpret the Lovász extension (2.5) as the linear interpolation of the values of the set function evaluated at the sets obtained via thresholding of the continuous f (2.3). From computational perspective, an important feature of the Lovász extension is that it can be evaluated efficiently; one needs to sort the given input f apart from evaluating the set function at the sets $C_i,\ i = 1, \ldots, n$.

Here we present some of the properties of the Lovász extension. All the results presented here can be found in [10].

Proposition 2.1 *Let $\hat{R}$ be a set function with $\hat{R}(\emptyset) = 0$ and $R : \mathbb{R}^n \to \mathbb{R}$ be its Lovász extension. Then the following hold:*

- *Let $\hat{S}$ be any set function with $\hat{S}(\emptyset) = 0$ and its Lovász extension be S, then the Lovász extension of $\hat{R} + \hat{S}$ is given by $R + S$. The Lovász extension of $\gamma \hat{R}$ for any $\gamma \in \mathbb{R}$ is given by γR.*
- *R is positively 1-homogeneous function.*
- *$R(f + \alpha \mathbf{1}_V) = R(f) + \alpha \hat{R}(V)$, for all $f \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$,*

2.1.3 Submodular set functions

An important class of set functions are the so-called submodular set functions, which, to a certain extent, play a similar role in combinatorial optimization as that of convex functions in continuous optimization [82]. Because of this, there has been a growing interest in modeling problems in terms of submodular set functions in machine learning and related areas [88, 71, 70, 80].

Definition 2.12 (Submodular set function) *A set function $\hat{R} : 2^V \to \mathbb{R}$ is submodular if and only if, for all subsets $A, B \subseteq V$, it holds*

$$\hat{R}(A) + \hat{R}(B) \geq \hat{R}(A \cup B) + \hat{R}(A \cap B).$$

It is clear from the definition that submodular set functions are closed under non-negative linear combinations. That is if $\hat{R}_1, \ldots, \hat{R}_k$ are submodular functions then for any non-negative scalars $\alpha_i \in \mathbb{R}_+$, $\sum_{i=1}^{k} \alpha_i \hat{R}_i$ is submodular. We similarly define a function $\hat{R}$ supermodular if $-\hat{R}$ is submodular. A function is called modular if it is both submodular and supermodular. Modular functions play a similar role in the set domain as that of linear functions in vector spaces.

We now state the result connecting submodularity and convexity [82].

Proposition 2.2 (Convexity and submodularity) *A set function is submodular if and only if its Lovász extension is convex.*

It turns out that every set function can be written as a difference of submodular set functions as shown in [56].

Proposition 2.3 *Every set function $\widehat{S}$ with $\widehat{S}(\emptyset) = 0$ can be written as $\widehat{S} = \widehat{S}_1 - \widehat{S}_2$, where S_1 and S_2 are submodular and $\widehat{S}_1(\emptyset) = \widehat{S}_2(\emptyset) = 0$. The Lovász extension S can be written as difference of convex functions.*

Let us derive the Lovász extensions of some submodular functions that we are going to use in this thesis.

Lemma 2.2 (Constant functions I) *Let $\hat{P}_0 : 2^V \to \mathbb{R}$ be defined as*

$$\hat{P}_0(C) := \begin{cases} 0 & \text{if } C = \emptyset, \\ a & \text{otherwise,} \end{cases}$$

for some $a \in \mathbb{R}$. The Lovász extension of $\hat{P}_0$ is given by $a \max_i\{f_i\}$ and $\hat{P}_0$ is submodular.

Proof: Let $f \in \mathbb{R}^n$, where $n = |V|$, be ordered in increasing order $f_1 \leq f_2 \leq \ldots, f_n$. Using the definition (2.4), the Lovász extension can be derived as

$$P_0(f) = \sum_{i=1}^{n-1} f_i(a - a) + f_n(a - 0) = a \max_i\{f_i\}.$$

The submodularity of $\hat{P}_0$ follows from the fact that $\max_i\{f_i\}$ is convex [17] and by Proposition 2.2. □

Lemma 2.3 (Constant functions II) *Let $\hat{P}_0 : 2^V \to \mathbb{R}$ be defined as*

$$\hat{P}_0(C) := \begin{cases} 0 & \text{if } C = \emptyset \text{ or } C = V \\ a & \text{otherwise,} \end{cases}$$

for some $a \in \mathbb{R}$. The Lovász extension of $\hat{P}_0$ is given by $a \max_i\{f_i\} - a \min_i\{f_i\}$ and $\hat{P}_0$ is submodular.

Proof: Again let $f \in \mathbb{R}^n$, be ordered in increasing order $f_1 \leq f_2 \leq \ldots, f_n$. Using the definition (2.4), the Lovász extension can be derived as

$$P_0(f) = f_1(0-a) + \sum_{i=2}^{n-1} f_i(a-a) + f_n(a-0) = a(\max_i\{f_i\} - \min_i\{f_i\}).$$

The submodularity of $\hat{P}_0$ follows from the fact that $\max_i\{f_i\}$ and $-\min_i\{f_i\}$ are convex [17] and by Proposition 2.2. □

Lemma 2.4 (Modular functions) *Let $\hat{R} : 2^V \to \mathbb{R}$ be defined as $\hat{R}(C) = \sum_{i\in C} d_i$ for some fixed $d \in \mathbb{R}^n$. Then $\hat{R}$ is modular and its Lovász extension is given by $\langle d, f\rangle$.*

Proof: Again let $f \in \mathbb{R}^n$, be ordered in increasing order $f_1 \leq f_2 \leq \ldots, f_n$. Using the definition (2.4), the Lovász extension can be derived as

$$R(f) = \sum_{i=1}^{n} f_i \left(\sum_{j\in C_{i-1}} d_j - \sum_{j\in C_i} d_j \right) = \sum_{i=1}^{n} d_i f_i.$$

Note that the Lovász extension of $\hat{R}$ is a convex (in fact linear) function which implies by Proposition 2.2 that $\hat{R}$ is submodular. One can immediately check that the Lovász extension of $-\hat{R}$ is also convex which combined with the previous fact implies that $\hat{R}$ is modular. □

Lemma 2.5 (Cut function) *Let $G(V, W)$ be a graph with symmetric weight matrix W. The cut function defined as*

$$\text{cut}(C, \overline{C}) = \sum_{i\in C,\ j\in\overline{C}} w_{ij},$$

is submodular. Its Lovász extension is given by the total variation function $\text{TV}(f) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}\,|f_i - f_j|$.

Proof: We derive the Lovász extension of cut using the Definition 2.2. First note that $\hat{R}(V) = \text{cut}(V, \emptyset) = 0$. Given $f \in \mathbb{R}^V$, let $(j_1, \ldots, j_n)$ be a permutation of V satisfying $f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_n}$. Then we have

$$\hat{R}(\{j_{i+1}, \ldots, j_n\}) = \text{cut}(\{j_{i+1}, \ldots, j_n\}, \{j_1, \ldots, j_i\}) = \sum_{l,m=1}^{n} w_{j_l j_m} \delta_{l\geq i+1} \delta_{m\leq i},$$

where $\delta_c = 1$ if the condition c is true and zero otherwise. Plugging this into the definition 2.2, we get

$$R(f) = \sum_{i=1}^{n-1} \sum_{l,m=1}^{n} w_{j_l j_m} \delta_{l\geq i+1} \delta_{m\leq i} (f_{j_{i+1}} - f_{j_i})$$

By exchanging the sums we get

$$R(f) = \sum_{l,m=1}^{n} w_{j_l j_m} \sum_{i=1}^{n-1} \delta_{l \geq i+1} \delta_{m \leq i} (f_{j_{i+1}} - f_{j_i}) = \sum_{l,m=1}^{n} w_{j_l j_m} \delta_{l>m} \sum_{i=m}^{l-1} (f_{j_{i+1}} - f_{j_i})$$
$$= \sum_{l,m=1}^{n} w_{j_l j_m} \delta_{l>m} (f_{j_l} - f_{j_m}) = \frac{1}{2} \sum_{l,m=1}^{n} w_{j_l j_m} \left| f_{j_l} - f_{j_m} \right|,$$

where the last step follows from the symmetry of W. Since the last summation is over all the possible indices, we can rewrite it as

$$\frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \left| f_i - f_j \right|.$$

Since this is a convex function, by Proposition 2.2 the cut function is submodular. □

The total variation $\mathrm{TV}(f) = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} \left| f_i - f_j \right|$ is a widely used regularizer in computer vision [101] as it prefers piecewise constant solutions. We now derive the subdifferential of the total variation function which will be useful in later chapters.

Lemma 2.6 *Let $G(V, W)$ be a graph with symmetric weight matrix W. The subdifferential of the total variation function at a point f is given by*

$$\partial(\mathrm{TV}(f))_r = \left\{ \sum_{j=1}^{m} w_{rj} u_{rj} \,|\, u_{rj} = -u_{jr}, u_{rj} \in \mathrm{sign}(f_r - f_j) \right\},$$

where

$$\mathrm{sign}(x) := \begin{cases} 1, & \text{if } x > 0 \\ [-1, 1], & \text{if } x = 0 \\ -1 & \text{otherwise} \end{cases}$$

Proof: The subdifferential of the function $g(x, y) = |x - y| = \left| \begin{pmatrix} -1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \right|$ is given by (see Theorem 23.9 in [98])

$$\partial g(x, y) = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \partial(|x - y|) = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \mathrm{sign}(x - y).$$

Now for the total variation, we have

$$\partial \mathrm{TV}(f) = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij} (e_i - e_j) \,\mathrm{sign}(f_i - f_j),$$

where e_i is the i^{th} unit vector of the standard basis of $\mathbb{R}^n$. Thus we have for $r = 1, \ldots, n$,

$$\partial \left(\mathrm{TV}(f) \right)_r = \frac{1}{2} \sum_{j=1}^{n} w_{rj} \,\mathrm{sign}(f_r - f_j) - \frac{1}{2} \sum_{i=1}^{n} w_{ir} \,\mathrm{sign}(f_i - f_r)$$
$$= \frac{1}{2} \sum_{j=1}^{n} w_{rj} \,\mathrm{sign}(f_r - f_j) - \frac{1}{2} \sum_{j=1}^{n} w_{jr} \,\mathrm{sign}(f_j - f_r)$$
$$= \frac{1}{2} \sum_{j=1}^{n} w_{rj} \left(\mathrm{sign}(f_r - f_j) - \mathrm{sign}(f_j - f_r) \right),$$

where the last equality follows because of the symmetry of W. Let $v_{rj} \in \text{sign}(f_r - f_j),\ r, j = 1, \ldots, n$. Note that $v_{rj} \neq -v_{jr}$ if $f_r = f_j$. Let $u_{rj} = \frac{1}{2}(v_{rj} - v_{jr}),\ r, j = 1, \ldots, n$. Then it holds that $u_{rj} \in \text{sign}(f_r - f_j)$ as well as $u_{rj} = -u_{jr}$. Thus the subdifferential can be written as

$$\partial \left(\text{TV}(f)\right)_r = \sum_{j=1}^{n} w_{rj} u_{rj},$$

with $u_{rj} = -u_{jr}$ and $u_{rj} \in \text{sign}(f_r - f_j)$. □

It may not be always possible to derive a closed-form expression for the Lovász extension for a given submodular set function making it difficult to compute the subdifferential. We now present a result that shows how to compute an element of the subdifferential of the Lovász extension using only the evaluations of the set function.

Proposition 2.4 *([10]) Let $\hat{R}$ be a submodular set function satisfying $\hat{R}(\emptyset) = 0$ and R be its Lovász extension. Let $\mathcal{B}(\hat{R})$ be the base polyhedron associated with $\hat{R}$ defined as*

$$\mathcal{B}(\hat{R}) = \{r \in \mathbb{R}^n, \langle r, \mathbf{1}_V \rangle = \hat{R}(V), \forall A \subseteq V, \langle r, \mathbf{1}_A \rangle \leq \hat{R}(A)\} \tag{2.6}$$

Then we have

$$\max_{r \in \mathcal{B}(\hat{R})} \langle r, f \rangle = R(f) \tag{2.7}$$

and a maximizer is given by

$$r^*_{j_i}(f) = \hat{R}\left(\{j_i, \ldots, j_n\}\right) - \hat{R}\left(\{j_{i+1}, \ldots, j_n\}\right),\ i = 1, \ldots, n,$$

where $(j_1, \ldots, j_n)$ is a permutation of V satisfying $f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_n}$.

Now it is straightforward to determine an element of the subdifferential of a submodular set function as shown in the following lemma.

Lemma 2.7 *Let $\hat{R}$ be a submodular set function satisfying $\hat{R}(\emptyset) = 0$ and R be its Lovász extension. Then an element $r \in \partial R(f)$ of the subdifferential of R at f is given by*

$$r_{j_i}(f) = \hat{R}\left(\{j_i, \ldots, j_n\}\right) - \hat{R}\left(\{j_{i+1}, \ldots, j_n\}\right),\ i = 1, \ldots, n,$$

where $(j_1, \ldots, j_n)$ is a permutation of V satisfying $f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_n}$.

Proof: For any given f, by Proposition 2.4, it follows that $\langle r(f), f \rangle = R(f)$, where $r(f)$ is as defined in the theorem statement. Thus, we have for any $g \in \mathbb{R}^n$

$$R(f) + \langle r(f), g - f \rangle = \langle r(f), g \rangle \leq \max_{r \in \mathcal{B}(\hat{R})} \langle r, g \rangle = R(g),$$

where $\mathcal{B}(\hat{R})$ is the base polyhedron defined in (2.6). The inequality above follows from the fact that $r(f) \in \mathcal{B}(\hat{R})$ and the last equality follows from Proposition 2.4. Thus we have for any $g \in \mathbb{R}^n$,

$$R(g) \geq R(f) + \langle r(f), g - f \rangle .$$

Thus $r(f)$ is and element of $\partial R(f)$. □

2.1.4 Exact continuous relaxations

We now present the known exact continuous relaxation results for a class of combinatorial optimization problems. By exact continuous relaxation, we mean that the minimum values of the combinatorial and the continuous problems are equal and a minimizer of the combinatorial problem can be obtained from a minimizer of the continuous problem (and vice-versa).

Theorem 2.1 *[10] Let $\hat{R} : 2^V \to \mathbb{R}$ be a submodular set function with $\hat{R}(\emptyset) = 0$. Then minimizing $\hat{R}$ is equivalent to minimizing its Lovász extension R on $[0,1]^n$, i.e.,*

$$\min_{C \subseteq V} \hat{R}(C) = \min_{f \in [0,1]^n} R(f).$$

Moreover, the set of minimizers of R on $[0,1]^n$ is the convex hull of minimizers of R on $\{0,1\}^n$.

This result has been generalized recently in [56] which established exact continuous relaxation results for combinatorial problems arising in graph-based clustering. In the following, we call a set function symmetric if $\hat{S}(C) = \hat{S}(\overline{C})$.

Theorem 2.2 *[56] Let $G(V,W)$ be a weighted undirected graph and $\hat{S}$ be a non-negative symmetric set function with $\hat{S}(\emptyset) = 0$, then*

$$\min_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C})}{\hat{S}(C)} = \min_{f \in \mathbb{R}^n} \frac{\mathrm{TV}(f)}{S(f)},$$

where TV *and S are the Lovász extensions of the cut function and the balancing function $\hat{S}$ respectively. Moreover, it holds for all $f \in \mathbb{R}^n$,*

$$\min_{i=1,\ldots,n-1} \frac{\mathrm{cut}(C_i, \overline{C_i})}{\hat{S}(C_i)} \leq \frac{\mathrm{TV}(f)}{S(f)},$$

where C_i are the sets as defined in (2.3)*. This implies that a minimizer C^* of the combinatorial problem can be obtained by optimal thresholding of any minimizer f^* of the continuous problem.*

The result also holds if S is any positively 1-homogeneous even convex function that extends a non-negative symmetric set function $\hat{S}(C)$ satisfying $\hat{S}(\emptyset) = 0$. We refer the reader to [56] for more details.

In our work [96], we have extended the exact relaxation result to the minimization of balanced cut under constraints arising in the constrained clustering setting. We will present this result as well as our constrained clustering method for the two-class setting in Chapter 3. Later in a joint work [22], we have further derived exact continuous relaxation result for the minimization of a general ratio of non-negative set functions under arbitrary constraints. Based on this result, we developed a new method for solving a generic version of the densest subgraph problem, which we discuss in Chapter 5.

Before presenting a recent algorithm for solving the optimization problem appearing in the continuous relaxation, we briefly review related methods in DC (difference of convex) programming and fractional programming.

2.1.5 DC Programming

A function $R : \mathbb{R}^n \to \mathbb{R}$ is called a DC (difference of convex) function if there exists a pair of convex functions R_1 and R_2 such that

$$R(f) = R_1(f) - R_2(f), \ \forall f \in \mathbb{R}^n.$$

The functions R_1 and R_2 are called DC components of R. A DC function has infinitely many DC decompositions [35]. A large class of functions, for example, every $F \in C^2(\mathbb{R}^n)$ can be written as a difference of convex functions [58]. Difference of convex (DC) programming addresses the problem of minimizing a DC function. A standard DC program is of the form (with the convention $\infty - \infty = \infty$)

$$\inf_{f \in \mathbb{R}^n} R_1(f) - R_2(f), \tag{2.8}$$

where $R_1 : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ and $R_2 : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ are lower semi-continuous proper convex functions [37]. DC programs provide a greater flexibility in modeling practical applications beyond convex programming. Several problems arising in machine learning can be formulated as DC programs [8, 75].

Algorithms for DC programming (DCA, for short) based on duality were first introduced in [38] and later extended by [35]. Originally DCA was developed by deriving a dual DC program associated with the (primal) problem (2.8) and designing a scheme for approximately solving both primal and dual problems simultaneously [35]. Algorithm 2.1 presents a simplified version of DCA [35] for solving the primal problem (2.8). The main idea here is to replace in every iteration the second component R_2 of the objective by its affine minorization at the current iterate and solve the resulting sequence of convex problems until convergence. Note that the affine minorization of a convex function R_2 at f^t is given by $R_2(f^t) + \langle r_2^t, f - f^t \rangle$, where r_2^t is an element of the subdifferential of R_2 at f^t. We now present some basic results concerning the convergence of DCA to a critical point of R. We say that $f \in \mathbb{R}^n$ is a critical point of the DC function $R = R_1 - R_2$ if $\partial R_1(f) \cap \partial R_2(f) \neq \emptyset$, where ∂ is the subdifferential of a convex function. Note that this is consistent with the general notion of critical point given in Definition 2.9 for non-smooth, non-convex functions.

Algorithm 2.1 DCA [35]: Minimization of a difference of convex functions

1: **Initialization:** $f^0 \in \mathbb{R}^n$
2: **repeat**
3: $\quad r_2^t \in \partial R_2(f^t)$
4: $\quad f^{t+1} = \arg\min_{u \in \mathbb{R}^n} \{R_1(u) - \langle u, r_2^t \rangle\}$
5: **until** $f^{t+1} = f^t$

Theorem 2.3 *[35] The sequence $\{f^t\}$ produced by DCA satisfies $F(f^t) \geq F(f^{t+1})$ and the equality holds if and only if both f^t and f^{t+1} are the critical points of $R_1 - R_2$ and $(\rho(R_1) + \rho(R_2)) \|f^{t+1} - f^t\| = 0$. That is, if $\rho(R_1) + \rho(R_2) > 0$ and $F(f^t) = F(f^{t+1})$, then $f^{t+1} = f^t$. Moreover, if the optimal value of problem (2.8) is finite and the sequences $\{f^t\}$ and $\{r_2^t\}$ are bounded, then every limit point f^* of the sequence $\{f^t\}$ produced by DCA is a critical point of $R_1 - R_2$.*

We refer the reader to [35, 36, 73] for further results on duality as well as a discussion on local and global optimality conditions for DC programming.

A more general constrained DC program [74, 37] is of the form

$$\begin{aligned} &\min_{f\in\mathbb{R}^n} R_1(f) - R_2(f), \\ &\text{subject to} : S(f) \leq 0, \\ &\qquad f \in D, \end{aligned} \tag{2.9}$$

where S is a DC function and $D \subseteq \mathbb{R}^n$ is a non-empty closed convex set. For simplicity of the presentation, we restrict ourselves to only one DC constraint; however, the ideas presented here can be extended to problems with multiple DC constraints. Let $S = S_1 - S_2$ be a DC decomposition of S. Similar to the idea presented in DCA, one can replace the second component S_2 of the constraint function S by its affine minorization in every iteration and solve the resulting sequence of constrained convex problems. This leads to inner approximation of the constraint set and consequently, the solution of each of the convex problems is feasible for the original problem (2.9). This way one generates a sequence of feasible points that achieve monotonic descent in the objective of problem (2.9) until convergence. A slightly different approach was developed in [74, 37] which is especially useful when the affine minorization yields an infeasible subproblem. The main idea here is to solve the following relaxed version of the subproblem in iteration t,

$$\min_{f\in\mathbb{R}^n,\ \gamma\in\mathbb{R}} R_1(f) - \left\langle r_2^t, f\right\rangle + \beta^t\gamma \tag{2.10}$$

$$\text{subject to} : S_1(f) - S_2(f^t) - \left\langle s_2^t, f - f^t\right\rangle \leq \gamma, \tag{2.11}$$

$$\gamma \geq 0, \tag{2.12}$$

$$f \in D,$$

where $r_2^t \in \partial R_2(f^t), s_2^t \in \partial S_2(f^t)$ and $\beta^t > 0$ is a penalty parameter for iteration t. Note that the case $\gamma = 0$ corresponds to the case where the constraint set is approximated by convex inner approximation as described above. The complete details are given in Algorithm 2.2. In Theorem 2.4, we present the convergence result of Algorithm 2.2. We need the following definition for presenting this result.

Definition 2.13 (KKT point) *Any $\bar{f} \in \mathbb{R}^n$ that is feasible for the problem* (2.9) *is a Karush-Kuhn-Tucker point for* (2.9) *if there exists non-negative scalar λ such that*

$$\begin{aligned} &0 \in \partial_C R(\bar{f}) + \lambda\partial_C S(\bar{f}) + N_D(\bar{f}) \\ &0 = \lambda S(\bar{f}), \end{aligned}$$

where $N_D(\bar{f})$ is the normal cone of D at the point $\bar{f}$.

We say that (extended) Mangasarian-Fromovitz constraint qualification is satisfied at a feasible point $\bar{f}$ of problem (2.9) if there is a vector $d \in \text{cone}(D - \{\bar{f}\})$ such that $S^{\uparrow}(\bar{f}, d) < 0$, where $\text{cone}(D)$ is the conic hull of the set D defined as

$$\left\{ \sum_{i=1}^{m} \alpha_i f_i \mid f_i \in D, \alpha_i \in \mathbb{R}, \alpha_i \geq 0,\ m = 1, 2, \dots \right\}.$$

Algorithm 2.2 Constrained DCA [74]: Minimization of constrained DC program (2.9)

Initialization: $f^0 \in D, \delta_1, \delta_2 > 0$, initial penalty parameter $\beta^0 > 0$
repeat
  $r_2^t \in \partial R_2(f^t)$, $s_2^t \in \partial S_2(f^t)$
  Let (f^{t+1}, γ^{t+1}) be an optimal solution of problem (2.10) and $(\lambda^{t+1}, \mu^{t+1})$ be the associated Lagrange multipliers corresponding to the constraints (2.11) and (2.12) respectively.
  $d^t = \min\{\|f^{t+1} - f^t\|^{-1}, \lambda^{t+1} + \delta_1\}$

$$\beta^{t+1} = \begin{cases} \beta^t & \text{if } \beta^t \geq d^t \\ \beta^t + \delta_2 & \text{otherwise} \end{cases}$$

until $f^{t+1} = f^t$ and $\gamma^{t+1} = 0$
Output: f^{t+1}.

Theorem 2.4 *[74] Suppose $D \in \mathbb{R}^n$ is a nonempty closed convex set and the objective function $R = R_1 - R_2$ and the constraint function $S = S_1 - S_2$ are DC functions on D such that the following assumptions are satisfied.*

1. *R and S are locally Lipschitz continuous functions at every point of D.*
2. *The extended Mangasarian-Fromovitz constraint qualification is satisfied at any $f \in \mathbb{R}^n$ satisfying $S(f) \geq 0$.*
3. *Either R_1 or R_2 is differentiable on D, either S_1 or S_2 is differentiable on D and*

$$\rho(R_1) + \rho(R_2) + \rho(S_1) > 0.$$

Let $\{f^t\}$ be the sequence generated by Algorithm 2.2. Then Algorithm 2.2 either stops after finitely many iterations at a KKT point of problem (2.9) or generates an infinite sequence $\{f^t\}$ of iterates such that $\lim_{t\to\infty} \|f^{t+1} - f^t\| = 0$ and every limit point of the sequence $\{f^t\}$ is a KKT point of problem (2.9).

We note here that an alternative approach for solving the constrained DC program (2.9) is to enforce the DC constraint itself via a penalty term [74]. For example, one can define the following penalty term directly for the DC constraint without approximating it

$$P(f) = \max\{0, S(f)\} = \max\{S_1(f), S_2(f)\} - S_2(f).$$

Thus, P is a DC function and one can use ideas similar to that of Algorithm 2.2 to solve the following penalized version iteratively,

$$\min_{f \in \mathbb{R}^n} \big(R_1(f) + \beta^t \max\{S_1(f), S_2(f)\}\big) - \big(R_2(f) + \beta^t S_2(f)\big) \tag{2.13}$$

$$\text{subject to} : f \in D. \tag{2.14}$$

We refer the reader to [74, 37] for more details.

2.1.6 Fractional programming

Fractional (i.e., ratio) programming is concerned with the minimization (or maximization) of a ratio of functions. Several real world problems such as resource allocation (minimize cost over return), portfolio selection (maximize return on investment) can be formulated as fractional programming problems [103].

Consider the following fractional programming problem,

$$\min_{f \in D} \frac{R(f)}{S(f)}, \tag{2.15}$$

where $D \subset \mathbb{R}^n$ is a compact and connected subset of $\mathbb{R}^n$ and $R : \mathbb{R}^n \to \mathbb{R}$, $S : \mathbb{R}^n \to \mathbb{R}$ are continuous functions on D. Let us further assume that $S(f) > 0, \forall f \in D$. Since R and S are continuous, D is compact and $S(f) > 0, \forall f \in D$, problem (2.15) has a solution.

It is shown in [39][1] that the following parametric problem

$$\min_{f \in D} R(f) - \lambda^* S(f), \tag{2.16}$$

where λ^* is the optimal value of problem (2.15), has the same minimizers as that of (2.15). Moreover the optimal value of (2.16) is zero. This parametric subproblem, assuming λ^* is known, can be optimally solved if R is convex, S is concave and $\lambda^* \geq 0$. Since λ^* is not known, one can instead use an upper bound $\bar{\lambda} \geq \lambda^*$ obtained from any feasible $\bar{f} \in D$ and solve the problem,

$$\min_{f \in D} R(f) - \bar{\lambda} S(f).$$

Since $\bar{f}$ is feasible for this problem and $\bar{\lambda} = \frac{R(\bar{f})}{S(\bar{f})}$, the optimal value of this problem is non-positive. If the optimal value is zero, then $\bar{\lambda} = \lambda^*$. Otherwise, the optimal solution leads to an improved upper bound. Sequentially solving the above problem using improved upper bounds leads to the Dinkelbach's method [39] (Algorithm 2.3) for solving a ratio of non-negative convex and concave functions.[2]

The following convergence result holds for Algorithm 2.3.

Theorem 2.5 *[39, 20] Let R and S be non-negative convex and concave functions respectively. The sequence $\{f^t\}$ produced by the Dinkelbach's method (Algorithm 2.3) satisfies $F(f^t) > F(f^{t+1})$ for all $t \geq 0$ or the sequence terminates. Moreover the sequence $\{f^t\}$ satisfies $\lim_{t \to \infty} \frac{R(f^t)}{S(f^t)} = \lambda^*$, where λ^* is the global minimum of the problem* (2.15).

[1] Originally, this result is shown for maximization problem; however as noted in [39], the statement is still valid for minimization problem (2.15).

[2] Originally, [39] considered maximization of a ratio of concave and convex functions with the assumption that $\max\{R(f) | f \in D\} \geq 0$. To maintain consistency with the following section, we presented this idea for minimization problems where the objective is a ratio of convex and concave functions and R is further assumed to be non-negative. The results presented in [39] can be directly transferred to this case [20].

Algorithm 2.3 Dinkelbach's Method [39]: Minimization of a ratio of non-negative convex and concave functions

1: **Initialization:** $f^0 \in D$, $\lambda^0 = \frac{R(f^0)}{S(f^0)}$
2: **repeat**
3: $\quad f^{t+1} = \arg\min_{u \in D} \{R(u) - \lambda^t S(u)\}$
4: $\quad \lambda^{t+1} = \frac{R(f^{t+1})}{S(f^{t+1})}$
5: **until** $R(f^{t+1}) - \lambda^t S(f^{t+1}) < \epsilon$
6: **Output:** f^{t+1}.

2.1.7 Nonlinear eigenproblems and RatioDCA

The exact continuous relaxation presented in Theorem 2.2 resulted in a continuous optimization problem where one needs to minimize a non-negative ratio of Lovász extensions. Note that the Lovász extensions of a general set function can be written as positively 1-homogeneous, difference of convex functions (by Propositions 2.1, 2.2 and 2.3). Now, we discuss recent advances in algorithms for solving such problems.

Let R_1, R_2, S_1, S_2 be convex, Lipschitz continuous, even and positively p- homogeneous ($p \geq 1$) functions and let $R = R_1 - R_2$, $S = S_1 - S_2$. Further assume that R and S are non-negative and $S(f) = 0$ if and only if $f = 0$. By considering a nonlinear ratio $F(f) = \frac{R(f)}{S(f)}$, the authors of [55, 20] motivate the following *nonlinear eigenproblem*,

$$0 \in \partial R_1(f) - \partial R_2(f) - \lambda \left(\partial S_1(f) - \partial S_2(f)\right), \tag{2.17}$$

where ∂ denotes the subdifferential of a convex function and $\lambda = F(f)$. Any f satisfying the above condition is referred to as a nonlinear eigenvector and λ as the corresponding nonlinear eigenvalue. Note that for any symmetric matrix A, one recovers with $R_1(f) = \langle f, Af \rangle$, $S_1(f) = \langle f, f \rangle$ and $R_2 = S_2 = 0$, the standard eigenproblem,

$$Af - \lambda f = 0.$$

The following result from [55] characterizes the relation between nonlinear eigenvectors and critical points of F.

Theorem 2.6 *[55, 20] Suppose that the functions $R = R_1 - R_2$ and $S = S_1 - S_2$ satisfy the conditions stated above. Then a necessary condition for f^* to be a critical point of F is*

$$0 \in \partial R_1(f) - \partial R_2(f) - \lambda \left(\partial S_1(f) - \partial S_2(f)\right),$$

where $\lambda^* = F(f^*)$. If S is continuously differentiable at f^*, then this condition is also sufficient.

A nonlinear inverse power method is proposed in [55] for computing a nonlinear eigenvector for the case where $R_2 = S_2 = 0$ and $p \geq 1$. A more general scheme called RatioDCA was proposed in [56] to minimize a non-negative ratio of positively 1-homogeneous, difference of convex functions

$$\min_{f \in \mathbb{R}^n} \frac{R_1(f) - R_2(f)}{S_1(f) - S_2(f)} =: F(f). \tag{2.18}$$

Algorithm 2.4 RatioDCA [56]: Minimization of F, a non-negative ratio of positively one-homogeneous DC functions

Initialization: $f^0 \in \mathbb{R}^n$, $\lambda^0 = F(f^0)$
repeat
  $f^{t+1} = \underset{u\in\mathbb{R}^n,\, \|u\|_2\leq 1}{\arg\min} \{R_1(u) - \langle u, r_2(f^t)\rangle \ + \lambda^t\big(S_2(u) - \langle u, s_1(f^t)\rangle\big)\}$
  where $r_2(f^t) \in \partial R_2(f^t)$, $s_1(f^t) \in \partial S_1(f^t)$
  $\lambda^{t+1} = F(f^{t+1})$
until $\frac{|\lambda^{t+1}-\lambda^t|}{\lambda^t} < \epsilon$
Output: eigenvalue λ^{t+1} and eigenvector f^{t+1}.

It is shown that RatioDCA (Algorithm 2.4) is a monotonic descent method and converges to an eigenvector associated with the eigenproblem (2.17).

Proposition 2.5 *[56] The sequence $\{f^t\}$ produced by RatioDCA satisfies $F(f^t) > F(f^{t+1})$ for all $t \geq 0$ or the sequence terminates.*

Theorem 2.7 *[56] Each cluster point $\{f^*\}$ of the sequence f^k produced by RatioDCA is a nonlinear eigenvector with eigenvalue $\lambda = \frac{R(f^*))}{S(f^*)} \in [0, F(f^0)]$ in the sense that if fulfills*

$$0 \in \partial R_1(f^*) - \partial R_2(f^*) - \lambda^* \left(\partial S_1(f^*) - \partial S_2(f^*)\right).$$

If $S_1 - S_2$ is continuously differentiable at f^*, then F has a critical point at f^*.

Since R_1, R_2, S_1 and S_2 are positively 1-homogeneous functions, an equivalent problem to (2.18) (with the convention $\frac{0}{0} := \infty$) is given by

$$\min_{f\in\mathbb{R}^n,\ \|f\|_2\leq 1} \frac{R_1(f) - R_2(f)}{S_1(f) - S_2(f)}. \tag{2.19}$$

Since the feasible set of this problem is compact, one can apply Dinkelbach's idea (see Section 2.1.6) and form the following parametric subproblem

$$\min_{u\in\mathbb{R}^n,\, \|u\|_2\leq 1} R_1(u) - R_2(u) - \lambda^t \left(S_1(u) - S_2(u)\right). \tag{2.20}$$

This is a DC programming problem. One can interpret each iteration of RatioDCA as applying only one step of DCA (Algorithm 2.1) to this DC problem. We note here that RatioDCA is not same as applying DCA to the Dinkelbach's parametric subproblem (2.20). This is because in the latter approach, λ^t is fixed until the subproblem (2.20) is fully solved whereas in RatioDCA, λ^t is updated after each step.

2.2 Clustering methods

In this section we review various state-of-the-art clustering techniques and highlight the connections between different methods.

2.2.1 k-means clustering

Let $\{x_1, \ldots, x_n\}$ be the given set of points where $x_i \in \mathbb{R}^n,\ i = 1, \ldots, n$. A possible formulation for the k-way clustering of the given points is

$$\min_{(C_1,\ldots,C_k)\in P_k} \sum_{i,j\in C_l} \|x_i - x_j\|^2 ,$$

where P_k is the set of all k-partitions of $\{1, \ldots, n\}$ and $\|\cdot\|$ is the standard Euclidean norm. The objective function here minimizes the within cluster dissimilarity. One can show that this problem is equivalent to the standard k-means clustering problem (see Theorem 8.18 of [107]) formulated as

$$\min_{(C_1,\ldots C_k)\in P_k,\ \mu_1,\ldots,\mu_k\in\mathbb{R}^n} \sum_{l=1}^{k} \sum_{j\in C_l} \|x_j - \mu_l\|^2 . \tag{2.21}$$

The following holds at an optimal solution $(C_1, \mu_1), \ldots (C_k, \mu_k)$ of this problem

$$\mu_l = \frac{1}{|C_l|} \sum_{i\in C_l} x_i,\ l = 1, \ldots, k.$$

Thus, k-means clustering aims to find k *prototypes* $\mu_l, l = 1, \ldots, k$ one for each cluster and minimize the distance of points in a cluster to its prototype.

Lloyd's algorithm

The k-means clustering problem has been shown to be NP-hard [2, 84] and hence one has to rely on approximate or heuristic methods in practice. Lloyd's algorithm [81] is a very simple and efficient method to solve the k-means problem locally optimally. The idea of this algorithm is to alternate between finding the prototypes by fixing the clusters and finding the clusters by fixing the prototypes until there is no change in the value of the objective (2.21). If one fixes the clustering then one can minimize the above objective over prototypes; in fact the solution is the means of current clusters. Similarly, if one fixes the prototypes, then the minimization over k-partitions simply results in assigning each point to its closest prototype. Note that the Lloyd's algorithm is a monotonic descent method for solving (2.21).

Drawbacks: A main drawback of the formulation of k-means clustering is that the objective enforces spherical clusters thus limiting its usefulness. Moreover, the Euclidean distance is not robust to outliers; other measures such as l_1 norm can be used leading to the so-called k-medians clustering.

Kernel k-means

A nonlinear generalization of the k-means problem is presented in [48] using symmetric positive definite kernels [105]. By choosing a symmetric positive-definite kernel $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, one transfers the k-means problem from the Euclidean space to the reproducing kernel Hilbert space $\mathcal{H}$ associated with the kernel K:

$$\min_{(C_1,\ldots C_k)\in P_k,\ \mu_1,\ldots,\mu_k\in\mathcal{H}} \sum_{l=1}^{k} \sum_{j\in C_l} \|\phi(x_j) - \mu_l\|_{\mathcal{H}}^2 ,$$

where $\phi : \mathbb{R}^n \to \mathcal{H}, x \mapsto K(x, \cdot)$. For the positive definite kernel K, it holds that

$$K(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}} .$$

Hence one can avoid the explicit feature representation in the kernel k-means objective by rewriting it in terms of inner products in $\mathcal{H}$

$$\begin{aligned} \|\phi(x_j) - \mu_l\|_{\mathcal{H}}^2 &= \langle \phi(x_j), \phi(x_j) \rangle_{\mathcal{H}} + \left\langle \frac{1}{|C_l|} \sum_{i \in C_l} \phi(x_i), \frac{1}{|C_l|} \sum_{i \in C_l} \phi(x_i) \right\rangle_{\mathcal{H}} \\ &\quad - 2 \left\langle \phi(x_j), \frac{1}{|C_l|} \sum_{i \in C_l} \phi(x_i) \right\rangle_{\mathcal{H}} \\ &= \langle \phi(x_j), \phi(x_j) \rangle_{\mathcal{H}} + \frac{1}{|C_l|^2} \sum_{i,r \in C_l} \langle \phi(x_i), \phi(x_r) \rangle_{\mathcal{H}} - \frac{2}{|C_l|} \sum_{i \in C_l} \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}} . \end{aligned}$$

Note that even in Lloyd's algorithm we never need to compute the prototypes (i.e., means) if we are interested only in obtaining the final clustering. We just need a way to compute the distance of a point to every mean. The distance computation to the mean can be done using only the kernel function as shown above.

2.2.2 Spectral clustering

One of the widely used criteria in graph-based clustering is the normalized cut. Given a similarity graph $G(V, W)$ where $V = \{1, \ldots, n\}$ is the vertex set and W is the symmetric weight matrix, the normalized cut problem is formulated as

$$\min_{(C_1, \ldots, C_k) \in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C}_l)}{\mathrm{vol}(C_l)} =: \mathrm{NCut}(C_1, \ldots, C_k), \tag{2.22}$$

where P_k is the set of all k-partitions of the vertex set V and $\mathrm{vol}(C) = \sum_{i \in C} d_i$, d_i is the degree of vertex i. Spectral clustering is based on solving the following continuous relaxation of this NP-hard combinatorial optimization problem [115],

$$\begin{aligned} &\min_{F \in \mathbb{R}^{n \times k}} \mathrm{tr}(F^T L F) \\ &\text{subject to} : F^T D F = I, \end{aligned} \tag{2.23}$$

where tr denotes the trace of a matrix, L is the symmetric graph Laplacian matrix defined as $L = D - W$, D is the diagonal matrix containing the degrees of vertices on the diagonal and I is the identity matrix. Note that the rows of the optimization variable F represent the vertices and the columns represent the clusters.

In fact the problems (2.22) and (2.23) are equivalent if F is restricted to have the following form, for $i = 1, \ldots, n$, $j = 1, \ldots, k$,

$$F_{ij} = \begin{cases} \frac{1}{\sqrt{\mathrm{vol}(C_j)}}, & i \in C_j \\ 0 & \text{otherwise} \end{cases}$$

where $(C_1, \ldots, C_k)$ is a k-partition of the vertex set V. Here by equivalence, we mean that the optimal values of both problems are equal and one can deduce an optimal solution of one problem from an optimal solution of the other problem.

Using the variable substitution $F = D^{-\frac{1}{2}}G$, we can rewrite the problem (2.23) as

$$\begin{aligned} &\min_{G \in \mathbb{R}^{n \times k}} \operatorname{tr}(G^T \bar{L} G) \\ &\text{subject to} : G^T G = I, \end{aligned}$$

where $\bar{L}$ is the normalized graph Laplacian defined as $\bar{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$. Since $\bar{L}$ is symmetric, the solution of this problem is given by the matrix G whose columns are the eigenvectors of $\bar{L}$ corresponding to the k smallest eigenvalues (see Theorem 1 of [44]). One obtains the solution of (2.23) from G as $F = D^{-\frac{1}{2}}G$.

When the number of clusters $k = 2$, then the spectral relaxation reduces to

$$\begin{aligned} \min_{f \in \mathbb{R}^n} \ & \langle f, Lf \rangle \\ \text{subject to} : \ & \langle f, Df \rangle = \operatorname{vol}(V), \\ & \langle Df, \mathbf{1}_n \rangle = 0, \end{aligned} \tag{2.24}$$

where $\mathbf{1}_n$ denote the vector of all ones. Similar to the general case, one can show that if f has the following special form, then this relaxation is in fact equivalent to the normalized cut problem (2.22),

$$f_i = \begin{cases} \sqrt{\frac{\operatorname{vol}(\overline{C})}{\operatorname{vol}(C)}}, & i \in C \\ -\sqrt{\frac{\operatorname{vol}(C)}{\operatorname{vol}(\overline{C})}} & \text{otherwise} \end{cases} \tag{2.25}$$

where $(C, \overline{C})$ is the 2-way partitioning of V and $i = 1, \ldots, n$.

Again, we can rewrite the problem (2.24) in terms of the normalized graph Laplacian $\bar{L}$ as

$$\begin{aligned} \min_{g \in \mathbb{R}^n} \ & \left\langle g, \bar{L} g \right\rangle \\ \text{subject to} : \ & \langle g, g \rangle = \operatorname{vol}(V), \\ & \left\langle g, D^{\frac{1}{2}} \mathbf{1}_n \right\rangle = 0. \end{aligned}$$

Note that $\bar{L}$ is symmetric and $D^{\frac{1}{2}}\mathbf{1}_n$ is the eigenvector of $\bar{L}$ corresponding to the smallest eigenvalue. Hence by Rayleigh-Ritz principle or Rayleigh theorem [59], the solution of the above problem is given by the eigenvector of $\bar{L}$ corresponding to the second smallest eigenvalue.

Ratio cuts, where the balancing function is the size of clusters, $\hat{S}(C) = |C|$, is another widely used objective in graph-based clustering. The relaxation of the ratio cut problem solved by spectral clustering is given by

$$\begin{aligned} &\min_{F \in \mathbb{R}^{n \times k}} \operatorname{tr}(F^T L F) \\ &\text{subject to} : F^T F = I. \end{aligned} \tag{2.26}$$

One can similarly show that this problem is equivalent to the ratio cut problem if F is restricted to have a certain special form.

Rounding the continuous solution: Although the spectral relaxation (2.23) can be optimally solved, it is not straightforward to obtain a clustering from the optimal solution. In practice, the optimal solution F is used to construct a new representation of the data, known as spectral embedding. One then applies k-means in this new representation to get the clustering. Precisely, spectral embedding $\phi : V \to \mathbb{R}^k$ is defined as $i \mapsto \phi(i) = (F_{i1}, \dots F_{ik})$ and k-means is applied on the points $\{\phi(i)\}_{i=1}^n$ in $\mathbb{R}^k$. We note here that applying k-means on the spectral embedding is only a heuristic without explicit connection to the normalized cut objective that we are optimizing; however perturbation theory of eigenvalues and eigenvectors of matrices, when applied to the graph Laplacian, provides some justification for such a heuristic [115].

When $k = 2$, apart from applying k-means on the spectral embedding, several heuristics have been proposed [53] such as optimal thresholding of the continuous solution f, rounding the continuous solution f based on the sign or the median of the components of f. One possible explanation for the rounding based on the sign is the structure of f (2.25) that yields the equivalence between (2.22) and (2.24). In optimal thresholding, one defines the following sets, for a given $f \in \mathbb{R}^n$,

$$C_0 = V,\ C_i = \{j \in V | f_j > f_i\},\ i = 1, \dots, n.$$

and chooses the set having the smallest balanced cut

$$C^* = \underset{C_i,\ i=1,\dots,n-1}{\arg\min} \frac{\mathrm{cut}(C_i, \overline{C_i})}{\hat{S}(C_i)}.$$

Clearly, optimal thresholding yields better results (in terms of obtaining a better normalized cut) than the other heuristics in the case of $k = 2$. Moreover, for ratio and normalized Cheeger cuts, it is shown in [21] that the cut value h^*_{NCC} obtained by optimal thresholding of the second eigenvector has the approximation guarantee

$$h_{\mathrm{NCC}} \leq h^*_{\mathrm{NCC}} \leq 2(h_{\mathrm{NCC}})^{\frac{1}{2}},$$

where h_{NCC} is the optimal normalized Cheeger cut value.

2.2.3 p-Spectral clustering

Using the graph p-Laplacian [3], a nonlinear generalization of the standard graph Laplacian, the so-called p-spectral clustering method was proposed in [21]. The method is motivated by first showing, for a connected graph, that the second eigenvector of the normalized graph p-Laplacian, for $1 < p < 2$, interpolates between a relaxation of the 2-way normalized cut and Cheeger cut. Note that the standard graph Laplacian introduced in the previous section corresponds to the case $p = 2$. They further show that in the limit $p \to 1$, the cut obtained by thresholding the second eigenvector of the normalized graph p-Laplacian converges to the optimal 2-way normalized Cheeger cut. More precisely, if $h_{\mathrm{NCC}}, h^*_{\mathrm{NCC}}$ denote respectively the optimal normalized Cheeger cut value and the cut value obtained by thresholding the second eigenvector of normalized graph p-Laplacian, then it holds for $p > 1$ [21] that

$$h_{\mathrm{NCC}} \leq h^*_{\mathrm{NCC}} \leq p(h_{\mathrm{NCC}})^{\frac{1}{p}}.$$

Notice that the bounds on the cut h^*_{NCC} obtained by thresholding the second eigenvector become tighter as $p \to 1$.

Given a graph $G(V, W)$ with the degrees of the vertices given by $d \in \mathbb{R}^n$, the problem of finding the second eigenvalue $\lambda_p^{(2)}$ of the normalized graph p-Laplacian is formulated in [21] as

$$\lambda_p^{(2)} = \min_{f \in \mathbb{R}^n} \frac{\frac{1}{2}\sum_{i,j=1}^n w_{ij} \left|f_i - f_j\right|^p}{\min_{c \in \mathbb{R}} \sum_{i=1}^n d_i \left|f_i - c\right|^p}, \tag{2.27}$$

from whose solution f^* the second eigenvector is derived as $v_p^2 = f^* - c^*\mathbf{1}$, where $c^* = \arg\min_{c \in \mathbb{R}} \sum_{i=1}^n d_i \left|f_i^* - c\right|^p$. A method is proposed for solving (2.27) in [21]; however since the problem (2.27) is non-convex, it is not guaranteed to obtain the global optimum. For two-class clustering, the clusters are obtained by thresholding the solution v_p^2 returned by their method. For multi-class clustering, they suggest recursive two-way splitting until the desired number of clusters is obtained. We note here that similar results for the ratio Cheeger cut were also established in [21] using the unnormalized version of the graph p-Laplacian.

2.2.4 1-Spectral clustering

Inspired by the tighter relaxation result of p-spectral clustering, the authors of [112] showed that there exists an exact continuous relaxation for the ratio Cheeger cut problem,

$$h_{\text{RCC}} = \min_{f \in \mathbb{R}^n} \frac{\frac{1}{2}\sum_{i,j=1}^n w_{ij} \left|f_i - f_j\right|}{\left\|f - \text{median}(f)\right\|_1}, \tag{2.28}$$

where h_{RCC} is the optimal ratio Cheeger cut value. It is further shown that there exists a minimizer f^* for the above problem taking only two distinct values in which case obtaining a clustering is trivial. Moreover, for any minimizer f^*, there exists a threshold γ such that the following two-valued function is also a minimizer

$$f_\gamma(x) = \begin{cases} 1 & x > \gamma \\ 0 & \text{otherwise} \end{cases}$$

A scheme based on Split Bregman method [50] is proposed in [112] for solving 2.28; however, they could not prove any convergence guarantees for the method.

Later it is established in [55] that the problem of finding the optimal Cheeger cut can be formulated as a nonlinear eigenproblem associated with the graph 1-Laplacian and hence the name 1-spectral clustering; refer to (2.17) in Section 2.1.7 for the definition of nonlinear eigenproblem. They showed that, for a connected graph, the optimal ratio Cheeger cut is in fact equal to the second eigenvalue of the graph 1-Laplacian and the second eigenvector is equal to the indicator function of the optimal partition. They also proposed a nonlinear inverse power method that is guaranteed to converge to a nonlinear eigenvector. These results have been generalized in [56] which shows that exact continuous relaxation exists for any balanced cut problem whenever the balanced cut is expressed as a ratio of cut and a balancing function; see Theorem 2.2 in Section 2.1.7. Moreover, a generic algorithm for solving a ratio of non-negative 1-homogeneous DC functions has been

proposed in [56]; see the RatioDCA Algorithm in Section 2.1.7. Similar to p-spectral clustering, one has to rely on the recursive two-way splitting procedure to obtain multiple clusters. Thus, it is difficult to incorporate prior information in 1-spectral clustering whenever $k > 2$.

In Chapter 3, we extend the exact continuous relaxation result to balanced cut problem subject to constraints arising in the constrained clustering setting. Then in Chapter 4, we develop a direct method for multi-class clustering that allows easy integration of prior information.

2.2.5 Relation between kernel k-means and normalized cut

The equivalence between weighted k-means and normalized cuts for positive semi-definite weight matrices W is first pointed in [11]. The authors of [28, 30, 31] discuss this relation for general weight matrices W. The weighted kernel k-means problem is given by

$$\min_{(C_1,\dots C_k)\in P_k,\ \mu_1,\dots,\mu_k\in\mathbb{R}^n} \sum_{l=1}^{k}\sum_{j\in C_l} b_j \left\|\phi(x_j)-\mu_l\right\|^2. \tag{2.29}$$

where $b_j \geq 0,\ j = 1,\dots,n$ are the weights of the data points and ϕ is a feature map to the reproducing kernel Hilbert space induced by a symmetric positive definite kernel K. Define the class proportions $p_l = \sum_{j\in C_l} b_j,\ l = 1,\dots,k$ and the (scaled) cluster assignment matrix $H \in \mathbb{R}^{n\times k}$ for a k-partition $(C_1,\dots,C_k)$ as

$$H_{il} = \begin{cases} \frac{1}{\sqrt{p_l}} & \text{if } i \in C_l \\ 0 & \text{otherwise} \end{cases} \tag{2.30}$$

Further let the matrix B be the diagonal matrix containing the weights $b_i,\ i = 1,\dots,n$ on the diagonal. Then the weighted kernel k-means problem (2.29) is equivalent to the following trace maximization problem

$$\begin{aligned} \max_{(C_1,\dots,C_k)\in P_k,\ F\in\mathbb{R}^{n\times k}} \quad & \operatorname{tr}(F^T B^{\frac{1}{2}} K B^{\frac{1}{2}} F) \\ \text{subject to}: \quad & F^T F = I \\ & F = B^{\frac{1}{2}} H \end{aligned} \tag{2.31}$$

One can also rewrite the normalized cut problem (2.22) on a graph $G(V,W)$ as a trace maximization problem

$$\begin{aligned} \max_{F\in\mathbb{R}^{n\times k}} \quad & \operatorname{tr}(F^T D^{-\frac{1}{2}} W D^{-\frac{1}{2}} F) \\ \text{subject to}: \quad & F^T F = I. \end{aligned} \tag{2.32}$$

with an additional constraint that that F has the special form $F = D^{\frac{1}{2}} H$, where D is the diagonal matrix containing the degrees of the vertices on the diagonal and H is defined similar to (2.30) using D instead of B. By comparing (2.31) and (2.32) one sees that for the choice $B = D$, the weighted kernel k-means problem (2.29) and the normalized cut problem (2.22) are related via $K = D^{-1}WD^{-1}$. One can

go from the kernel k-means problem to normalized cut using $W = DKD$ and from normalized cut to the kernel k-means using $K = D^{-1}WD^{-1}$. We note here that the relation holds for any generic diagonal matrix D, for example, the ratio cut problem, where $D = I$.

Note that in order for the equivalence to hold, the kernel matrix K obtained via $K = D^{-1}WD^{-1}$ has to be positive semi-definite (as pointed in [11]), which does not hold for a generic weight matrix W. It is shown in [31] that one can construct a positive semi-definite kernel matrix K' from W via $K' = \sigma D^{-1} + D^{-1}WD^{-1}$ such that the equivalence still holds; here σ is large enough positive constant to ensure K' is positive semi-definite. This way it is shown that any method (e.g., a variant of the Lloyd's algorithm) for solving the weighted kernel k-means problem can be used in (approximately) minimizing the normalized cut on the weight matrix W by choosing $K = \sigma D^{-1} + D^{-1}WD^{-1}$ where D is the diagonal degree matrix and σ is a large positive constant.

One main drawback the authors of [31] highlight is that the addition of the diagonal shift σD^{-1}, although does not change the equivalence, has adverse effect on the Lloyd's algorithm. This is because for large positive values of σ the points become closer to their current means and farther from the remaining ones and consequently the Lloyd's algorithm gets stuck early [29, 31].

Graclus

Using the relation between kernel k-means and normalized cut, [31] presented a multi-level graph partitioning method similar to [66] for minimizing normalized cut. The proposed method called as Graclus proceeds in three phases. In the first phase, a sequence of coarser graphs $G_i(V_i, W_i)$ of decreasing size (in terms of the number of vertices) is generated by collapsing edges and merging vertices. This coarsening phase stops when the size of the coarser graph is smaller than $5k$ vertices, where k is the number of clusters. In the next phase, the initial k-clustering is obtained on the coarsest graph using the region-growing method of [66]. Then in the last phase known as the refinement phase, the partitioning from the coarser graph is iteratively propagated to the finer graphs and refined. More specifically, partitioning on G_i is used to construct an initial partition on G_{i-1} and then the Lloyd's algorithm is run by choosing the kernel matrix that yields the equivalence between normalized cut and kernel k-means. Since the Lloyd's algorithm is a monotonic descent method, the refinement phase is guaranteed to monotonically decrease the normalized cut.

Comments on the relation: We note here that the equivalence holds only for a limited class of balanced cuts. Moreover, on the algorithmic front, there are not many choices for solving the kernel k-means problem other than the Lloyd's algorithm which is shown [31] to have severe problems on the equivalent formulation. On the other hand, the graph-based setting gives more modeling freedom in terms of choosing application specific balancing function and recent advances in continuous optimization provide better methods [55, 56] for solving the graph-based formulations.

2.2.6 Clustering based on non-negative matrix factorization

Now we review clustering methods based on non-negative matrix factorization. Non-negative matrix factorization (NMF for short) is proposed as an unsupervised method for learning a parts-based representation of non-negative data [76, 77]. Given a non-negative data matrix $X \in \mathbb{R}_+^{d\times n}$, where each column $X_i \in \mathbb{R}^d$ represents a data point, the NMF problem is to find two low-rank non-negative factors $U \in \mathbb{R}_+^{d\times k}$ and $H \in \mathbb{R}_+^{k\times n}$, where $k < \min\{d, n\}$, that closely approximate X, i.e., $X \approx UH$. Here one can interpret the columns of U as providing a new *basis* for the data points and the columns of H as giving the encodings (coefficients) in the new basis. Since each data point has to be expressed as an additive ($H \geq 0$) combination of the non-negative basis U, one expects that U represents coherent parts of the data, thus providing a parts-of-whole interpretation. If one chooses the Frobenius norm as the measure of the quality of the approximation, then the non-negative matrix factorization problem is given by

$$\min_{U\in\mathbb{R}_+^{d\times k},\ H\in\mathbb{R}_+^{k\times n}} \|X - UH\|_F^2 . \tag{2.33}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. Note that without the non-negativity constraints the objective of the problem (2.33) can be globally minimized using the well-known singular value decomposition [41]. It has been shown that the non-negativity constraints make the problem (2.33) NP-hard [114] (more precisely, when the rank of X is k). An alternative minimization method based on a novel multiplicative update has been proposed in [78] for approximately solving the NMF problem (2.33). It has been empirically shown that NMF approach is successful in learning the parts-based representation; when applied to the data matrix containing facial images as columns, the basis U recovered different parts of the face.

A series of NMF-based clustering methods have been proposed leveraging the algorithmic ideas of [78]. In clustering one would like to represent each data point using only one of the basis vectors (e.g., prototype of a cluster) unlike the NMF formulation, where the data point is represented by a non-negative combination of the basis vectors. Furthermore, for clustering one does not necessarily need the explicit feature representation of the data points as required by NMF approach. Different methods thus differ by the way they incorporate the clustering or partitioning constraint and by the type of input matrix that is being factorized.

Orthogonal non-negative matrix factorization

An NMF-based formulation with orthogonal (more precisely, orthonormality) constraints has been proposed for clustering non-negative data $X \in \mathbb{R}_+^{d\times n}$ in [34],

$$\begin{aligned} \min_{U\in\mathbb{R}_+^{d\times k},\ H\in\mathbb{R}_+^{n\times k}} & \left\|X - UH^T\right\|_F^2 \\ \text{subject to} : & : H^T H = I_{k\times k} \end{aligned} \tag{2.34}$$

Note that for notational convenience, we used the transpose for the second factor in (2.34) in contrast to (2.33). It is shown [34] that the orthogonal-NMF problem

(2.34) (ONMF for short) is in fact equivalent to the k-means clustering problem

$$\min_{(C_1,\dots C_k)\in P_k,\ \mu_1,\dots,\mu_k\in\mathbb{R}^n} \sum_{l=1}^{k}\sum_{j\in C_l} \|\phi(x_j)-\mu_l\|^2 . \tag{2.35}$$

where $(C_1,\dots,C_k)$ is a k-partition of the ground set $V=\{1,\dots,n\}$ and $\mu_i \in \mathbb{R}^d,\ i=1,\dots,k$. One can verify that an optimal solution (U^*,H^*) of (2.34) and an optimal partition $(C_1^*,\dots,C_k^*)$ of (2.35) satisfy the following: the columns of H^* are the scaled indicator vectors of the components $C_1^*,\dots,C_k^*$ and a column i of U^* is the scaled mean vector of the component to which the data point i belongs. An alternative minimization method based on a multiplicative update adapted from [78] has been suggested in [34] for solving (2.34). The main drawback of the proposed method is that the final solution is not guaranteed to satisfy the orthogonal constraint (see the discussion at the end of Section 7 of [34]). More importantly it is not clear how well the method performs against the Lloyd's algorithm which solves the same k-means clustering problem. In their experiments, they always use the solution obtained from the Lloyd's algorithm as an initialization. More specifically, they suggest starting their method always with $H+0.2$ where H is derived from the solution of the Lloyd's algorithm. Finally, we note that in order to apply this method one needs an explicit feature representation of the data.

Symmetric orthogonal non-negative matrix factorization

The need for explicit feature representation for solving the formulation (2.34) can be eliminated by noting that an optimal solution (U^*,H^*) of (2.34) satisfy $U^*=X^*H^*$. Thus (2.34) can be equivalently rewritten as [32]

$$\max_{H\in\mathbb{R}_+^{n\times k}} \operatorname{tr}(H^TX^TXH)$$
$$\text{subject to}: H^TH=I.$$

This can again be transformed to the following non-negative matrix factorization problem [32]

$$\min_{H\in\mathbb{R}_+^{n\times k}} \left\|X^TX-HH^T\right\|_F^2 \tag{2.36}$$
$$\text{subject to}: H^TH=I.$$

Thus the clustering problem is written as factorizing the similarity matrix X^TX into two factors H and H^T. A generic 3-factor formulation was proposed in [32, 34] to factorize a symmetric similarity matrix $W\in\mathbb{R}_+^{n\times n}$ that is not necessarily positive semi-definite unlike X^TX,

$$\min_{H\in\mathbb{R}_+^{n\times k},\ S\in\mathbb{R}_+^{k\times k}} \left\|W-HSH^T\right\|_F^2 \tag{2.37}$$
$$\text{subject to}: H^TH=I.$$

A multiplicative update algorithm has been proposed in [34] for solving this formulation. As mentioned previously, the method uses a perturbed solution of k-means clustering as an initialization (i.e., $H+0.2$) and its solution is not guaranteed to satisfy the orthogonal constraint.

Non-negative matrix factorization for spectral clustering

The authors of [33] propose solving the spectral relaxation (2.23) with additional non-negativity constraints on F

$$\min_{F \in \mathbb{R}_+^{n \times k}} \operatorname{tr}(F^T L F) \qquad (2.38)$$
$$\text{subject to} : F^T D F = I.$$

They further show that this problem can be rewritten as

$$\max_{F \in \mathbb{R}_+^{n \times k}} \operatorname{tr}(F^T W F) \qquad (2.39)$$
$$\text{subject to} : F^T D F = I,$$

where W is the weight matrix of the underlying K-NN graph. A multiplicative algorithm is proposed in [33] for solving the above trace maximization problem by adapting the NMF-algorithm of [78]. In this case also it is not shown that the converged solution is guaranteed to satisfy the orthogonal constraint. Like earlier NMF-based methods, [33] also suggest starting their method with $H + 0.2$ where H is the indicator matrix of the partition found by the spectral clustering method.

Projective non-negative matrix factorization

The authors of [124] propose the so-called projective NMF formulation for factorizing a non-negative data matrix $X \in \mathbb{R}_+^{d \times n}$

$$\min_{H \in \mathbb{R}_+^{d \times k}} \left\| X - H H^T X \right\|_F^2 . \qquad (2.40)$$

If H satisfies the orthogonality constraint $H^T H = I$, then the above problem corresponds to finding a projection matrix $P = H H^T$ (hence the name projective NMF) that closely approximates the data X, i.e., $X \approx PX$. The authors note that even without the orthogonality constraint the solution of (2.40) found by their multiplicative update is approximately (but not exactly) orthogonal. We note here that without the non-negativity constraints this formulation is equivalent to the well-known principal component analysis [93, 63], if the data matrix X is centered (i.e., each row of X has mean zero).

For the clustering problem, one has to use $X \in \mathbb{R}^{n \times d}$ in the above formulation (2.40) [124]. The multiplicative update proposed by [124] needs only pairwise similarities for solving the clustering problem with this formulation. In their implementation they used the weight matrix of the K-NN graph as the similarity matrix. Similar to [34, 33] they start their method from the perturbed $(H + 0.2)$ solution of k-means clustering.

Left-stochastic matrix factorization

Building on the idea of factorizing a non-negative symmetric similarity matrix (2.36) [34], the following formulation is proposed in [9]

$$\min_{c \in \mathbb{R}} \min_{H \in \mathbb{R}_+^{n \times k}} \left\| cW - H H^T \right\|_F^2 \qquad (2.41)$$
$$\text{subject to} : : H \mathbf{1}_k = \mathbf{1}_n.$$

where W is a similarity matrix. Apart from estimating the scaling factor c this formulation requires the factor H to be a cluster probability matrix thus yielding a connection to the soft version of k-means clustering [9]. Unlike the other NMF-based approaches that rely on multiplicative updates, a geometric method is proposed in [9] for solving (2.41). A clustering (partitioning) is obtained from the solution of (2.41) by assigning i^{th} data point to the j^{th} cluster if H_{ij} achieves the maximum value in the i^{th} row of H where the ties are broken arbitrarily.

Non-negative matrix factorization using graph random walk

In contrast to the above NMF-based methods that factorize the weight matrix W of the K-NN graph $G(V, W)$, authors of [123] propose replacing W by its smoothed version A obtained via graph random walk resulting in the problem

$$\begin{aligned} &\min_{H \in \mathbb{R}_+^{n \times k}} \left\| A - HH^T \right\|_F^2 \\ &\text{subject to} : : H^T H = I \end{aligned} \tag{2.42}$$

where A is given by $A = \frac{1}{c}(I - \alpha D^{-\frac{1}{2}} W D^{-\frac{1}{2}})^{-1}$, c is a normalizing factor, D is the diagonal matrix containing the degrees of the vertices V and α is a smoothness parameter controlling the extent of the random walk.

Instead of directly solving the above problem (2.42), [123] suggest the following regularized version

$$\begin{aligned} &\min_{H \in \mathbb{R}_+^{n \times k}} - \operatorname{tr}(H^T A H) + \lambda \sum_{i=1}^{n} \left(\sum_{j=1}^{k} H_{ij}^2 \right)^2 \\ &\text{subject to} : : H^T H = I, \end{aligned} \tag{2.43}$$

where λ is a parameter. Note that the first term is obtained by expanding the objective of (2.42) and the second term is introduced to reduce the diagonal magnitudes in the approximating matrix (which correspond to self-similarities) and increase the off-diagonal correlation. In their large scale experiments, the parameters α of the random walk and λ of the regularization are set to the fixed values 0.8 and $\frac{1}{2k}$ respectively without any justification. Similar to other NMF-based methods, they derive a multiplicative update for the formulation (2.43), use the perturbed version (i.e., $H + 0.2$) of the spectral clustering solution as an initialization and do not have any guarantees that the converged solution satisfies the orthogonal constraint.

Chapter 3

Two-class clustering with constraints

In this chapter, we consider the constrained clustering problem for the two-class setting; the multi-class setting is discussed in Chapter 4. We present a generalization of the popular spectral clustering technique for integrating instance level prior information such as pairwise constraints and label constraints. The main idea of our approach is to derive a formulation that allows minimization of a trade-off between constraint violation and the clustering objective (balanced cuts in our case). Unlike the prior work, we show that this formulation allows us to guarantee a solution that satisfies all constraints in a hard-enforcement setting, while still able to handle noisy or inconsistent constraints. We then show that our formulation, a combinatorial optimization problem, can *equivalently* be rewritten as a continuous optimization problem in the same spirit as in [56]. We then present an efficient method based on the recent algorithm RatioDCA [56] for solving the continuous problem. Despite the additional complexity arising because of cannot-link constraints, we show that our method for solving the subproblem of RatioDCA has the same time complexity as that of the unconstrained problem. Another contribution of this chapter is a preconditioning technique for solving the subproblem of RatioDCA. The work presented in this chapter is published in [96].

All the constrained clustering methods can directly handle must-link and cannot-link constraints. The label constraints can be transformed to pairwise constraints by having a must-link between every pair of points with the same label and cannot-link between every pair of points with different labels. Hence for simplicity we present our method only for must-link and cannot-link constraints. However, note that the label information is stronger than the pairwise constraints. There is more than one way to assign labels to the given points while still satisfying the pairwise constraints and thus pairwise constraints alone do not reveal the true label information.

In this chapter, we denote by $G(V, W)$ the similarity graph constructed from the given pairwise similarities where $V = \{1, \ldots, n\}$ is the n-element vertex set, $W \in \mathbb{R}_+^{n \times n}$ is the symmetric weight matrix. We assume in this chapter that G is connected. We denote the entries of W by $w_{ij},\ i, j = 1, \ldots n$. Let $E = \{(i, j) \mid w_{ij} > 0\}$ be the set of edges of G and $\overrightarrow{E} = \{(i, j) \in E,\ i < j\}$ denote the *directed* edges. We assume that there are no self loops in G, i.e., $(i, i) \notin E,\ \forall i \in V$ and hence $|E| = 2\left|\overrightarrow{E}\right|$. We allow the vertices to have non-negative weights denoted by $b \in \mathbb{R}_+^n$.

We denote by $\mathbf{1}_n$ the $n \times 1$ vector of all ones and by $\mathbf{1}_C$ the indicator vector on a set C whose i^{th} entry is 1 if $i \in C$ and 0 otherwise. We denote the complement of a set C by $\overline{C} = V \backslash C$. The cut function is defined as $\text{cut}(C, \overline{C}) = \sum_{i \in C,\ j \in \overline{C}} w_{ij}$. The (generalized) volume of a set $C \subseteq V$ is defined as $\text{vol}(C) = \sum_{i \in C} b_i$, which specializes to cardinality if $b_i = 1,\ \forall i \in V$ and to the well-known volume function used in normalized cut when $b_i = d_i,\ \forall i \in V$, where $d_i = \sum_{j=1}^{n} w_{ij}$ is the degree of vertex i. Unlike the existing methods, which are designed using normalized cuts exclusively, we develop our method based on a general balanced cut problem given by

$$\min_{C \subseteq V} \frac{\text{cut}(C, \overline{C})}{\hat{S}(C)} =: \text{BCut}(C, \overline{C})$$

where $\hat{S}(C) : 2^V \to \mathbb{R}_+$ is a non-negative balancing function satisfying $\hat{S}(\emptyset) = \hat{S}(V) = 0$. We assume that the balancing function is defined in terms of the generalized volume which in turn depends on the vertex weights b. For example, in the case of ratio and normalized cut, $\hat{S}(C) = \text{vol}(C)\,\text{vol}(\overline{C})$, with $b_i = 1, \forall i \in V$ in the case of ratio cut and $b_i = d_i, \forall i \in V$ in the case of normalized cut. Note that for ratio and normalized cuts this definition differs with the general definition of the balanced cut (1.1) by a constant factor $\text{vol}(V)$. We use the convention $\frac{0}{0} = \infty$ so that the sets $\emptyset, V$ are never the solution of the balanced cut problem.

In the constrained clustering setting, we are additionally given pairwise must-link constraints, $\mathcal{M} = \{(p, q) : p \in V, q \in V\}$ and cannot-link constraints, $\mathcal{Q} = \{(p, q) : p \in V, q \in V\}$. We also consider a slightly general setting where a *degree-of-belief* is associated with each constraint. Let us define the *constraint* graphs for must-link and cannot-link constraints. Let $G^c(V, W^c)$ denote the cannot-link constraint graph with the weight matrix W^c whose entries $w^c_{ij} \in [0, 1]$, specify the degree of belief for the constrained pair $(i, j) \in \mathcal{Q}$. We make W^c symmetric by setting $w^c_{ji} = w^c_{ij}$ whenever $(i, j) \in \mathcal{Q}$. Similarly we define must-link constraint graph $G^m(V, W^m)$ with degrees of belief w^m_{ij} for each $(i, j) \in \mathcal{M}$. We assume for both the constraint graphs that the vertex weights are given by the degrees of vertices. We explicitly mention the graph under consideration when specifying the cut and the volume functions to avoid confusion; for example cut_{G^c} refers to the cut on the graph G^c. When there is no mention of the graph, it is assumed that the similarity graph G is being considered.

We first discuss the existing work for constrained clustering in Section 3.1. We then give our formulation for the constrained clustering problem in Section 3.2. Section 3.3 discusses an elegant way to directly integrate must-link constraints in a hard-enforcement setting. We present our exact continuous relaxation result in Section 3.4 and an efficient method for solving it in Section 3.5. We also discuss our preconditioning technique in Section 3.5. We finally present our experiments in Section 3.6.

3.1 State-of-the-art

Here we present the state-of-the-art spectral methods in constrained clustering.

3.1.1 Spectral learning

One of the first methods to incorporate pairwise constraints into spectral clustering framework is spectral learning (SL, for short) [64], which proposes to impose the pairwise constraints by directly modifying the similarities between the data points. The must-link points p and q are made more similar than any other pair of points by setting their similarity W^m_{pq} (and W^m_{qp}) to the maximum value and the cannot-link points r and s are made more dissimilar by setting $W^c_{rs} = W^c_{sr} = 0$. They then suggest running the usual spectral clustering algorithm on the modified weight matrix. Note that there is no guarantee that the solution of this method satisfies even a single constraint neither does it have a mechanism to control the trade-off between constraint violation and the clustering objective.

3.1.2 Flexible constrained spectral clustering

Flexible constrained spectral clustering method (CSP, for short) [117] encodes the pairwise supervision using the constraint matrix $Q = W^m - W^c$, where W^m and W^c respectively describe the degrees-of-belief for must-link and cannot-link constraints. Note that given a cluster indicator function $f \in \{+1, -1\}^n$, the quantity $\langle f, Qf \rangle$ measures the amount of constraints satisfied. Thus, the method proposes to solve the spectral relaxation with an additional constraint requiring a minimum amount of constraint satisfaction,

$$\langle f, Qf \rangle \geq \alpha,$$

where α is a user-defined parameter. The formulation proposed in [117] is given by

$$\begin{aligned} \min_{g \in \mathbb{R}^n} \ & \langle g, \bar{L} g \rangle \\ \text{subject to :} \ & \langle g, g \rangle = \text{vol}(V) \\ & \langle g, \bar{Q} g \rangle \geq \alpha \\ & g \neq D^{\frac{1}{2}} \mathbf{1}_n, \end{aligned} \tag{3.1}$$

where $\bar{L}$ is the normalized graph Laplacian, $\bar{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$, $\bar{Q}$ is the normalized constraint matrix $\bar{Q} = D^{-\frac{1}{2}}(W^m - W^c)D^{-\frac{1}{2}}$ and vol(V) is the sum of the degrees of all vertices V. They compute the final clustering by rounding the solution $f^* = D^{-\frac{1}{2}} g^*$ based on the sign of its components. Note that, in contrast to the standard spectral relaxation, they do not require g to be orthogonal to the trivial solution $D^{\frac{1}{2}}\mathbf{1}_n$. They simply require g to be different from it.

The above problem (3.1) is a non-convex problem; the authors of [117] propose to solve this using the KKT optimality conditions for constrained optimization problems [17]. They show that a set of candidates satisfying all the necessary KKT conditions can be obtained by solving a generalized eigenvalue problem for all the eigenvectors. Then the candidate with the smallest objective is an optimal solution; they explicitly filter the trivial solution $D^{\frac{1}{2}}\mathbf{1}_n$ in this step.

Improving over spectral learning, this method seems to provide a control over the constraint violation via the parameter α. However, it is not clear if this parameter α allows one to derive the full spectrum of solutions, in particular the important case of complete constraint satisfaction. It is not discussed in [117] if there exists a value of α for which all the constraints are satisfied. Although in the experiments

they suggest some value based on the the maximum and minimum eigenvalues of $\bar{Q}$, there is no proper justification for this choice. Moreover, another main drawback of this method is its prohibitive computational complexity of $O(n^3)$ because of the need to solve the full eigenvalue problem.

3.1.3 Spectral clustering with linear constraints

In image segmentation, there have been several efforts [125, 126, 42, 121] in incorporating prior information into the spectral clustering framework. The prior information mostly consisted of pixel labels and grouping of pixels, which for example can be encoded as must-links. They [125, 126, 42, 121] suggest to model this prior information by a set of linear constraints and solve the constrained spectral relaxation of the form

$$\begin{aligned} \min_{g \in \mathbb{R}^n} \ & \langle g, \bar{L} g \rangle \\ \text{subject to :} \ & \langle g, g \rangle = 1 \\ & Bg = c \end{aligned} \tag{3.2}$$

where $\bar{L}$ is the normalized graph Laplacian. Note that the usual orthogonality constraint $\left\langle g, D^{\frac{1}{2}} \mathbf{1}_n \right\rangle = 0$, which avoids the trivial solution, is encoded in the linear system $Bg = c$.

Must-link and cannot-link constraints are considered in this context by [42], who propose to formulate these as the following linear constraints,

$$g_p - g_q = 0, \quad \forall (p, q) \in \mathcal{M}$$

$$g_p + g_q = 0, \quad \forall (p, q) \in \mathcal{N}.$$

Although must-links are correctly formulated, the encoding of cannot-links has modeling drawbacks. First observe that any g that assigns zero to the points p and q where (p, q) is a cannot-link constraint, is still feasible for the problem (3.2); however any rounding of the solution of (3.2) cannot yield a clustering that satisfies this cannot-link constraint. Moreover, it is unnecessary to require g to have the same (absolute) value on the cannot-link vertices p and q; it should suffice to restrict them to have different signs. In fact from the derivation of spectral clustering (see (2.25) from Chapter 2), one notices that requiring them to have the same value with different signs corresponds to the case where both C and $\overline{C}$ have exactly the same volume. Thus the above formulation of cannot-link constraints introduces unnecessary bias, which is also confirmed in our experiments.

Unlike the methods presented previously, this is a hard-enforcement setting: must-links are guaranteed to satisfy irrespective of the rounding procedure; if the continuous solution does not assign zero to the vertices involved in cannot-link constraints, then cannot-links are guaranteed to be satisfied if the solution is rounded based on the sign, which the authors of [42, 121] also recommend.

In our experiments, we use the efficient projected power method proposed in [121] for solving the above constrained spectral relaxation (3.2). They [121] refor-

mulate the above problem as an equivalent constrained eigenvalue problem

$$\begin{aligned} \max_{g \in \mathbb{R}^n} \ & \langle g, (\alpha \mathbb{I}_{n\times n} - \bar{L}) g \rangle \\ \text{subject to :} \ & \|g\| = 1, \\ & Bg = c \end{aligned}$$

where $\mathbb{I}_{n\times n}$ is the $n \times n$ identity matrix, α is chosen large enough so that $\alpha \mathbb{I}_{n\times n} - \bar{L}$ is positive-semidefinite. One choice for α is the maximum eigenvalue of $\bar{L}$. A modified power method that projects the iterates on to the hyperplane $Bg = c$ in each iteration, is proposed in [121] for solving this linearly constrained eigenvalue problem. It is shown that the projected power method converges to the global optimal solution of the constrained eigenvalue problem.

3.1.4 Constrained clustering via spectral regularization

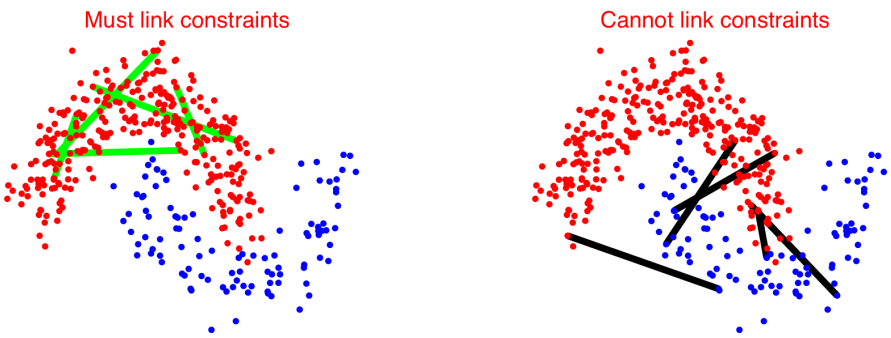
Constrained clustering via spectral regularization (CCSR, for short) [79] attempts to incorporate pairwise constraints directly into the multi-class setting. The main idea of this method is to learn a new data representation, derived from a low-dimensional spectral embedding, that is most consistent with the given pairwise constraints. Let $F_r = (v_1, \ldots, v_r),\ F_r \in \mathbb{R}^{n\times r}$ denote the spectral embedding, where v_i is the eigenvector of the normalized graph Laplacian $\bar{L}$ corresponding to the i^{th} smallest eigenvalue. The authors of [79] propose to learn a new k-dimensional data representation $F = (f_1, \ldots, f_k),\ F \in \mathbb{R}^{n\times k}$ such that $F = F_r A$, where $A \in \mathbb{R}^{r\times k}$ is the coefficient matrix to be learned. Ideally, one would like to have the new data representation F to be the cluster assignment matrix, i.e., the columns of F are indicator vectors of the k clusters. Consequently, if $y_i \in \mathbb{R}^k,\ \forall i \in V$, denote the rows of F, it should hold that $\langle y_p, y_q \rangle = 0, \forall (p,q) \in \mathcal{Q}$ and $\langle y_p, y_q \rangle = 1, \forall (p,q) \in \mathcal{M}$. Hence the following cost function is suggested for constrained clustering in [79]

$$\sum_{i=1}^{n} (\langle y_i, y_i \rangle - 1)^2 + \sum_{(p,q)\in\mathcal{M}} (\langle y_p, y_q \rangle - 1)^2 + \sum_{(p,q)\in\mathcal{Q}} (\langle y_p, y_q \rangle - 0)^2,$$

where the first term is introduced for the normalization purposes, the second and third terms encode the must-link and cannot-link constraints respectively. In terms of the coefficient matrix A the formulation is given by

$$\min_{A\in\mathbb{R}^{r\times k}} \sum_{(i,j,t_{ij})\in\mathcal{S})} (u_i^T A A^T u_j - t_{ij})^2,$$

where u_i is the i^{th} row of F_r, $\mathcal{S} = \{i, j, t_{ij}\}$ the set of pairwise constraints with $t_{ij} = 1$ for must-link pair (i,j) and $t_{ij} = 0$ for cannot-link pair (i,j). It is also assumed that $(i,i,1) \in \mathcal{S}, \forall i \in V$ for the sake of normalizing the rows of F. Introducing a new variable $M \in \mathbb{R}^{r\times r},\ M = AA^T,\ M \succcurlyeq 0$, the above problem is relaxed to a semi-definite program which can be solved in $O(r^2)$, where r is the dimension of the spectral embedding. They suggest using $r = 15$ in the experiments. Since in practice M does not necessarily yield a cluster assignment matrix F, the authors of [79] suggest applying k-means on the new data representation $F = F_r M^{\frac{1}{2}}$.



(a) (b)

Figure 3.1: (a) Must-link constraint graph where the must-links are denoted by green edges; any partition with zero cut value satisfies all the constraints. (b) cannot-link constraint graph where the cannot-links are by denoted by black edges; any partition that cuts every edge satisfies all the constraints.

The main modeling drawback is that the cost function completely ignores the orthogonality constraint required for the columns of F in order that F is a cluster indicator matrix. Moreover, it is not clear how strongly the constraints are enforced; in fact the method does not have any control on the amount of constraints violated as revealed in our experiments. The influence of the dimension of the spectral embedding r on the solution is also not studied and in practice it is not clear what would be a good value for r.

3.2 Formulation as constrained balanced cut problem

Unlike the other approaches that try to incorporate constraints into spectral clustering framework directly, our approach is derived from the first principles, in particular from the balanced cut problem. The essential idea is to reformulate the constrained clustering problem as a suitable constrained balanced cut problem. The first step is to define appropriate set functions that capture the degree of constraint violation of a partition $(C, \overline{C})$. For must-link constraints, the amount of violation of a partition $(C, \overline{C})$ is given by the cut on the must-link constraint graph, while for cannot-links, it is the opposite: total weight of the edges that are not cut by the partition $(C, \overline{C})$ on the cannot-link constraint graph (see Figure 3.1). Define $\hat{M}, \hat{Q} : 2^V \to \mathbb{R}_+$ as follows.

$$\begin{aligned} \hat{M}(C) &:= \mathrm{cut}_{G^m}(C, \overline{C}) && \text{(must-links)} \\ \hat{Q}(C) &:= \frac{1}{2}\left(\mathrm{assoc}_{G^c}(C) + \mathrm{assoc}_{G^c}(\overline{C})\right) && \text{(cannot-links)} \\ &= \frac{1}{2}\mathrm{vol}_{G^c}(V) - \mathrm{cut}_{G^c}(C, \overline{C}) \end{aligned}$$

Let us introduce $\hat{T}(C) : 2^V \to \mathbb{R}_+$ to capture the total amount of constraint

violation,

$$\hat{T}(C) := \hat{M}(C) + \hat{Q}(C).$$

Note that $\hat{T}$ is non-negative and is increasing with the amount of constraint violation. In fact each violated constraint increases $\hat{T}$ by the corresponding degree-of-belief. If a partition $(C, \overline{C})$ satisfies all the constraints, then $\hat{T}(C) = 0$.

Definition 3.1 (Consistent partition) *We define a partition $(C, \overline{C})$ of V as consistent if $\hat{T}(C) = 0$.*

The two-class clustering problem can be formulated as the following constrained balanced cut problem

$$\begin{aligned} &\min_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C})}{\hat{S}(C)} \\ &\text{subject to} : \hat{T}(C) = 0. \end{aligned} \tag{3.3}$$

In a hard-enforcement setting, we would like to guarantee that all the constraints are satisfied. On the other hand, if the constraints are unreliable and/or inconsistent, we would prefer to optimize a trade-off between the balanced cut and the amount of constraint violation. Here, we show that there is a way to address both settings in a common framework. For this, we first transform the constrained problem (3.3) into an unconstrained one using a *penalty* parameter $\gamma \in \mathbb{R}_+$,

$$\min_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C}) + \gamma \hat{T}(C)}{\hat{S}(C)}. \tag{3.4}$$

Since $\hat{T}(C) \geq 0, \forall C \subseteq V$, the constraint function $\hat{T}$ is also valid as the penalty function for the constraint $\hat{T}(C) = 0$. Moreover, since $\hat{T}(C) = 0$ for any consistent partition $(C, \overline{C})$, the problem (3.4) corresponds to a trade-off between having small balanced cut and satisfying all constraints. Among the partitions that have same amount of constraint violation, problem (3.4) prefers those with higher $\hat{S}(C)$, thus introducing bias towards more balanced partitions.

Before establishing the equivalence between problems (3.3) and (3.4) for a specific choice of γ, let us characterize the relation between the parameter γ and the amount of constraint violation of the solution of (3.4). For this, define θ to be the minimum value among all the degrees-of-belief

$$\theta = \min \left\{ \min_{(p,q) \in \mathcal{M}} \{w^m_{pq}\}, \min_{(p,q) \in \mathcal{Q}} \{w^c_{pq}\} \right\}. \tag{3.5}$$

It is reasonable to assume that $\theta > 0$, i.e., each constraint has a positive degree-of-belief.

Lemma 3.1 *Let $(A_0, \overline{A_0})$ be a consistent partition (i.e., it satisfies all the constraints) and $\chi_0 = \mathrm{BCut}(A_0, \overline{A_0})$. If $\gamma > \frac{\max_{B \subseteq V} \hat{S}(B)}{(l+1)\theta} \chi_0$, where θ is defined in (3.5), then any minimizer $(C^*, \overline{C^*})$ of the problem (3.4) violates no more than l constraints.*

Proof: First note that χ_0 is the objective value of the partition $(A_0, \overline{A_0})$ for the problem (3.4). Assume for the sake of contradiction that a minimizer $(C^*, \overline{C^*})$ of (3.4) violates at least $l+1$ constraints. Then it holds that $\hat{T}(C^*) \geq (l+1)\theta$, since θ is the minimum of all the degrees-of-belief. Let χ^* denote the objective value of $(C^*, \overline{C^*})$ for the problem (3.4). Then we have for the given value of γ,

$$\chi^* = \mathrm{BCut}(C^*, \overline{C^*}) + \gamma \frac{\hat{T}(C^*)}{\hat{S}(C^*)} \geq \frac{\gamma(l+1)\theta}{\max_{B \subseteq V} \hat{S}(B)} > \chi_0,$$

which is a contradiction since χ^* is the optimal value and χ_0 is the objective value of the partition $(A_0, \overline{A_0})$. Hence any partition $(C^*, \overline{C^*})$ which violates more than l constraints cannot be a solution of problem (3.4). □

As we will see in Section 3.3, it is easy to construct a consistent partition in $O(|V|+|\mathcal{Q}|)$ time and thus the above choice of γ in the above lemma is constructive. Now the main result concerning the equivalence between the problems is immediate, assuming the constraints are consistent.

Theorem 3.1 *Let $(A_0, \overline{A_0})$ be a consistent partition (i.e., it satisfies all the constraints) and $\chi_0 = \mathrm{BCut}(A_0, \overline{A_0})$. If $\gamma > \frac{\max_{B \subseteq V} \hat{S}(B)}{\theta} \chi_0$, where θ is defined in (3.5), it holds that*

$$\operatorname*{arg\,min}_{\substack{C \subseteq V \\ \hat{T}(C)=0}} \frac{\mathrm{cut}(C, \overline{C})}{\hat{S}(C)} = \operatorname*{arg\,min}_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C}) + \gamma \hat{T}(C)}{\hat{S}(C)}$$

and the optimum values of both problems are equal.

Proof: By Lemma 3.1 with $l = 0$, any minimizer of the unconstrained problem does not violate any constraint. Moreover $\hat{T}(C) = 0$ for any consistent partition and hence the objective values of both problem are equal for consistent partitions. Thus the equivalence holds. □

Thus the constrained balanced cut problem (3.3) can be equivalently formulated as the unconstrained problem (3.4) by choosing a suitable γ which can be computed in linear time $O(|V|+|\mathcal{Q}|)$. Note that the above equivalence holds for any given positive degrees-of-belief. Moreover, the minimum value of γ needed for the equivalence and hence for enforcing all the constraints decreases with increasing value of degree-of-belief θ. In practice, we recommend enforcing all the constraints this way only if all the degree-of-belief are set to 1, i.e., $\theta = 1$.

3.3 Direct integration of must-link constraints

In a hard-enforcement setting, we show that there exists an elegant way of integrating must-link constraints which works even in the multi-class setting (i.e., $k > 2$). The idea is to merge each pair of must-link vertices into a single vertex and solve a slightly modified *unconstrained* balanced cut problem on the reduced graph. We show that a proper redefinition of vertex weights as well as the edge weights would ensure that the unconstrained optimal cut on the reduced graph is equal to the

constrained optimum on the original graph. Moreover, any optimal partition of the original problem can be recovered from optimal solutions on the reduced graph.

Recall that $G^m(V, W^m)$ is the must-link constraint graph. Let G^m have t connected components and let $A_1, \ldots, A_t$ be the vertices of these connected components. By transitivity of must-link constraints all the vertices in a connected component should belong to the same cluster. Thus the vertices A_i in each connected component can be merged into a single vertex. The construction of a reduced graph is given below for one connected component. One can iterate the following procedure for each connected component.

Construction of the reduced graph:

1. introduce a new vertex τ_i in place of a connected component containing the vertices A_i: $V' = \{\tau_i\} \cup (V \backslash A_i)$.

2. define the edges on the reduced graph as follows: for every $r \in V' \backslash \{\tau_i\}$, add an edge between τ_i and r with the weight $\sum_{j \in A_i} w_{jr}$. The edges for any other vertices $i \neq \tau_i, j \neq \tau_i$ is defined as $w'_{ij} = w_{ij}$.

3. define the weights for the new vertex set V' as follows: $b'_{\tau_i} = \sum_{j \in A_i} b_j$ and for all $i \neq \tau_i$, $b'_i = b_i$.

Note that this construction leads to a graph with vertex weights even if the original graph did not have vertex weights. The following lemma shows that the above construction preserves all balanced cuts which respect the must-link constraints. We prove it for the case where we merge one connected component and the proof can easily be extended to the general case.

Lemma 3.2 *Let $G'(V', W')$ be the reduced graph of $G(V, W)$ obtained by merging the vertices A using the above procedure. Moreover let b' and b be the weights of the vertices of G' and G respectively. If a partition $(C, \overline{C})$ of V does not separate the vertices A (i.e., either $A \subseteq C$ or $A \subseteq \overline{C}$), then there exists a partition $(C', \overline{C'})$ of V' with same the balanced cut value,* $\mathrm{BCut}_G(C, \overline{C}) = \mathrm{BCut}_{G'}(C', \overline{C'})$.

Proof: If $(C, \overline{C})$ does not separate the vertices A, then we have either $\tau \in C$ or $\tau \in \overline{C}$. Without loss of generality, assume that $\tau \in C$. Consider the following partition of V': $C' = \{\tau\} \cup (C \backslash A)$ and $\overline{C'} = \overline{C}$. We have preserved the cut in G' because

$$\begin{aligned}\mathrm{cut}(C', \overline{C'}) &= \sum_{i \in C',\ j \in \overline{C'}} w'_{ij} = \sum_{i \in C' \backslash \{\tau\},\ j \in \overline{C'}} w'_{ij} + \sum_{j \in \overline{C'}} w'_{\tau j} = \sum_{i \in C \backslash A,\ j \in \overline{C}} w_{ij} + \sum_{j \in \overline{C}} \sum_{i \in A} w_{ij} \\ &= \mathrm{cut}(C, \overline{C}).\end{aligned}$$

Similarly, the volumes are also preserved,

$$\begin{aligned}\mathrm{vol}(C') &= \sum_{i \in C'} b'_i = b'_\tau + \sum_{i \in C' \backslash \{\tau\}} = \sum_{j \in A} b_j + \sum_{i \in C \backslash A} b_i = \sum_{i \in C} b_i = \mathrm{vol}(C), \\ \mathrm{vol}(\overline{C'}) &= \sum_{i \in \overline{C'}} b'_i = \sum_{i \in \overline{C}} b_i = \mathrm{vol}(\overline{C}).\end{aligned}$$

Thus we have $\mathrm{BCut}_G(C, \overline{C}) = \mathrm{BCut}_{G'}(C', \overline{C'})$, since the balanced cut criterion is based on the cut and the volumes of a partition. □

All partitions of the reduced graph fulfill all must-link constraints and thus any relaxation of the *unconstrained* balanced cut problem can now be used. In particular, the spectral relaxation for the constrained clustering problem with only must-link constraints can be formulated as

$$\begin{aligned} \min_{f \in \mathbb{R}^{|V'|}} \; & \langle f, L'f \rangle \\ \text{subject to :} \; & \langle f, B'f \rangle = \text{vol}(V') \\ & \left\langle f, B'\mathbf{1}_{|V'|} \right\rangle = 0 \end{aligned}$$

where L' is the unnormalized graph Laplacian $L' = D' - W'$, D', B' are the diagonal matrices containing respectively the degrees and vertex weights of G' on the diagonal. In this way we have recovered in an elegant manner the method of Yu and Shi [126] which considers only must-link constraints and requires the solution of expensive constrained eigenvalue problem to integrate them. Also our integration of must-link constraints works on the graph level and thus can be used by any other method, whereas their derivation is restricted to the normalized cut problem.

Detecting inconsistency

Note that must-link constraints are never inconsistent. However, cannot-link constraints either independently or in the presence of must-link constraints can be inconsistent. One can check if the given cannot-ink constraints are inconsistent by solving the two-coloring problem on the cannot-link constraint graph. It is easy to verify that the constraints are consistent if and only if the corresponding constraint graph can be colored with two colors. Moreover the two-coloring also yields a consistent partition. In the presence of both must-link and cannot-link constraints, one can first merge all the must-link constraints and transform the problem onto the reduced similarity graph G' and update the cannot-links accordingly. Since on the reduced problem all the must-link constraints are already satisfied, one has only the cannot-link constraints whose consistency can be checked by solving the two-coloring problem. Since the two-coloring problem can be optimally solved in linear time using breadth-first-search (see Theorem 2 in [87]), one can detect if the given pairwise constraints are consistent and also find a consistent partition efficiently in the two-class setting.

3.4 Exact continuous relaxation of constrained balanced cut problem

It has been shown in Theorem 3.1 that the constrained balanced cut problem (3.3) is equivalent to the unconstrained problem

$$\min_{C \subseteq V} \frac{\text{cut}(C, \overline{C}) + \gamma \hat{T}(C)}{\hat{S}(C)} =: \hat{F}^{(\gamma)}(C), \tag{3.6}$$

for a suitable value of γ, where

$$\hat{T}(C) = \hat{M}(C) + \hat{Q}(C) = \text{cut}_{G^m}(C, \overline{C}) + \frac{1}{2}\,\text{vol}_{G^c}(V) - \text{cut}_{G^c}(C, \overline{C}). \tag{3.7}$$

Building on the result of [56] given in Theorem 2.2, we proved in [96] the exact continuous relaxation result for the normalized cut problem with must-link and cannot-link constraints. Later on in a joint work [22], we showed that exact continuous relaxations exist for any constrained fractional set programs, i.e., optimization of a ratio of non-negative set functions subject to constraints specified by any set function. Here we show the general result for the minimization of ratio of non-negative set functions from which the exact continuous relaxation of the constrained balanced cut problem is then deduced.

Theorem 3.2 *Let $\hat{R}, \hat{S} : 2^V \to \mathbb{R}$ be any non-negative set functions and $R, S : \mathbb{R}^n \to \mathbb{R}$ be their Lovász extensions respectively. Further let $\hat{R}(\emptyset) = \hat{S}(\emptyset) = \hat{R}(V) = \hat{S}(V) = 0$. Then it holds that*

$$\inf_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)} = \inf_{f \in \mathbb{R}^n} \frac{R(f)}{S(f)} .$$

Moreover, it holds for all $f \in \mathbb{R}^n$,

$$\min_{i=1,\ldots,n-1} \frac{\hat{R}(C_i)}{\hat{S}(C_i)} \leq \frac{R(f)}{S(f)},$$

where the sets C_i are defined as

$$C_0 = V, \; C_i = \{j \in V | f_j > f_i\}, \; i = 1, \ldots, n. \tag{3.8}$$

That is a minimizer C^* of the ratio of set functions can be obtained by optimal thresholding of any minimizer f^* of the continuous problem.

Proof: Without loss of generality assume that components of $f \in \mathbb{R}^n$ are ordered in increasing order $f_1 \leq \ldots, \leq f_n$. We have by the definition of the Lovász extension (2.5),

$$\begin{aligned} R(f) &= \sum_{i=1}^{n-1} \hat{R}(C_i)(f_{i+1} - f_i) \\ &= \sum_{i=1}^{n-1} \frac{\hat{R}(C_i)}{\hat{S}(C_i)} \hat{S}(C_i)(f_{i+1} - f_i) \\ &\geq \min_{j=1,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} \left(\sum_{i=1}^{n-1} \hat{S}(C_i)(f_{i+1} - f_i) \right) = \min_{j=1,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} S(f), \end{aligned}$$

where the inequality follows because of the non-negativity of $\hat{S}$. Moreover, since $\hat{S}$ is non-negative, it follows from the definition of the Lovász extension (2.5) and the assumption $\hat{S}(V) = 0$ that $S(f) \geq 0, \forall f \in \mathbb{R}^n$. Thus dividing both sides by $S(f)$, we have for any $f \in \mathbb{R}^n$,

$$\frac{R(f)}{S(f)} \geq \min_{j=1,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)}.$$

This implies

$$\min_{f \in \mathbb{R}^n} \frac{R(f)}{S(f)} \geq \min_{f \in \mathbb{R}^n} \min_{j=1,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} \geq \min_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)}.$$

Here we used the convention $\frac{0}{0} = \infty$ for the last inequality. On the other hand, the continuous problem is a relaxation of the combinatorial problem because $\hat{R}(C) = R(\mathbf{1}_C)$ and $\hat{S}(C) = S(\mathbf{1}_C)$, $\forall C \subseteq V$. Thus by the virtue of relaxation, we have

$$\min_{f \in \mathbb{R}^n} \frac{R(f)}{S(f)} \leq \min_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)},$$

which establishes the result. □

Now we discuss the exact relaxation of the problem (3.6). Because of the constant $\frac{1}{2}\operatorname{vol}_{G^c}(V)$, the penalty term $\hat{T}$ in the objective function of (3.6) does not vanish on the empty set, a technical condition required in the definition of the Lovász extension. Moreover the above theorem requires the set functions to vanish also on the full set V; this assumption is satisfied only by the cut function and the balancing function $\hat{S}$. Thus, in order to derive the exact continuous relaxation, we introduce a new penalty function $\hat{T}_0 : 2^V \to \mathbb{R}_+$ by replacing the constant term,

$$\hat{T}_0(C) := \operatorname{cut}_{G^m}(C, \overline{C}) + \frac{1}{2}\operatorname{vol}_{G^c}(V)\hat{P}_0(C) - \operatorname{cut}_{G^c}(C, \overline{C}), \tag{3.9}$$

where

$$\hat{P}_0(C) := \begin{cases} 0 & \text{if } C = \emptyset \text{ or } C = V, \\ 1 & \text{otherwise} \end{cases}$$

Both penalty functions $\hat{T}$ and $\hat{T}_0$ agree everywhere except on the empty set and the full set V. Since empty set and the full set are never optimal for (3.6), the following problem is equivalent to (3.6)

$$\min_{C \subseteq V} \frac{\operatorname{cut}(C, \overline{C}) + \gamma \hat{T}_0(C)}{\hat{S}(C)} =: \hat{F}_0^{(\gamma)}(C). \tag{3.10}$$

Now we present the exact continuous relaxation result for this problem.

Theorem 3.3 *Let $G(V, W)$ be a weighted undirected graph and $\hat{S}$ be any non-negative balancing function with $\hat{S}(\emptyset) = 0$, $\hat{S}(V) = 0$ and $\hat{T}_0$ be the penalty term defined in* (3.9) *for the constraint graphs G^m, G^c. Then for any $\gamma \geq 0$, it holds that*

$$\min_{C \subseteq V} \frac{\operatorname{cut}(C, \overline{C}) + \gamma \hat{T}_0(C)}{\hat{S}(C)} = \min_{f \in \mathbb{R}^n} \frac{\operatorname{TV}_G(f) + \gamma T_0(f)}{S(f)},$$

where

$$\operatorname{TV}_G(f) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij} \left| f_i - f_j \right|, \tag{3.11}$$

$$T_0(f) = \operatorname{TV}_{G^m}(f) - \operatorname{TV}_{G^c}(f) + \frac{1}{2}\operatorname{vol}_{G^c}(V)\left(\max_i\{f_i\} - \min_i\{f_i\}\right), \tag{3.12}$$

and S are the Lovász extensions of the cut function on the graph G, the penalty function $\hat{T}_0$ and the balancing function $\hat{S}$ respectively. Moreover, it holds for all $f \in \mathbb{R}^n$,

$$\min_{i=1,\ldots,n-1} \frac{\operatorname{cut}(C_i, \overline{C_i}) + \gamma \hat{T}_0(C_i)}{\hat{S}(C_i)} \leq \frac{\operatorname{TV}_G(f) + \gamma T_0(f)}{S(f)},$$

where the sets C_i are defined as

$$C_0 = V,\ C_i = \{j \in V | f_j > f_i\},\ i = 1, \ldots, n. \tag{3.13}$$

That is a minimizer C^* of the problem (3.10) *can be obtained by optimal thresholding of any minimizer f^* of the continuous problem.*

Proof: It is clear from Proposition 2.1, Lemma 2.4 and Lemma 2.5 that $\mathrm{TV}_G(f) + \gamma T_0(f)$ is the Lovász extension of the set function $\mathrm{cut}(C, \overline{C}) + \gamma \hat{T}_0(C)$. Thus the result follows directly from Theorem 3.2. □

Now, we state the main result: the problem of minimizing balanced cut under must-link and cannot-link constraints is equivalent to minimizing the functional

$$\frac{\mathrm{TV}_G(f) + \gamma T_0(f)}{S(f)},$$

over real-valued f for a specific choice of γ. The proof follows directly from Theorem 3.1 and Theorem 3.3.

Theorem 3.4 *Let $G(V, W)$ be a weighted undirected graph and $\hat{S}$ be any non-negative balancing function with $\hat{S}(\emptyset) = 0$, $\hat{S}(V) = 0$ and $\hat{T}$ be the penalty term defined in* (3.7) *for the constraint graphs G^m, G^c. Further let $(A_0, \overline{A_0})$ be a consistent partition (i.e., $\hat{T}(A_0) = 0$) and $\chi_0 = \mathrm{BCut}(A_0, \overline{A_0})$. If $\gamma > \frac{\max_{B \subseteq V} \hat{S}(B)}{\theta} \chi_0$, where θ is defined in* (3.5)*, it holds that*

$$\min_{C \subseteq V,\ \hat{T}(C)=0} \frac{\mathrm{cut}(C, \overline{C})}{\hat{S}(C)} = \min_{f \in \mathbb{R}^n} \frac{\mathrm{TV}_G(f) + \gamma T_0(f)}{S(f)},$$

where $\mathrm{TV}_G(f)$, $T_0(f)$ (given in (3.11)*) and S are the Lovász extension of the cut function on the graph G, the penalty function $\hat{T}_0$ defined in* (3.9) *and the balancing function $\hat{S}$ respectively. Furthermore, an optimal partition of the constrained problem can be obtained from a minimizer of the continuous problem.*

Proof: From Theorem 3.1 we know that, for the chosen value of γ, the constrained problem is equivalent to

$$\arg\min_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C}) + \gamma \hat{T}(C)}{\hat{S}(C)}.$$

Since the empty set and the full set V are not optimal, this problem is equivalent to (3.10), which is in turn is equivalent, by Theorem 3.3, to the continuous problem in the statement. Moreover, as shown in Theorem 3.3, a minimizer C^* of this problem (and consequently the constrained problem) can be obtained by optimal thresholding of any minimizer of f^* of the continuous problem. □

A few comments on the implications of Theorem 3.4. First, it shows that the constrained balanced cut problem can be equivalently solved by solving the continuous relaxation for the given value of γ. The value of γ depends on the balanced cut value of a partition consistent with given constraints. Note that such a partition can be obtained in polynomial time by solving a two-coloring problem on the constraint graph as long as the constraints are consistent (see Section 3.3).

3.5 Algorithm for constrained balanced cut problem

3.5.1 Solution via RatioDCA

In this section, we discuss the efficient minimization of the exact continuous relaxation given by

$$\min_{f\in\mathbb{R}^n} \frac{\mathrm{TV}_G(f)+\gamma T_0(f)}{S(f)} =: F_0^{(\gamma)}(f),$$

where $T_0(f) = \mathrm{TV}_{G^m}(f) - \mathrm{TV}_{G^c}(f) + a\,(\max_i\{f_i\} - \min_i\{f_i\})$ with $a \in \mathbb{R}$ defined as $a = \frac{1}{2}\,\mathrm{vol}_{G^c}(V)$ and S is Lovász extension of $\hat{S}$. For the ease of exposition, we assume in this section that S is convex i.e., the corresponding balancing function $\hat{S}$ is submodular; this case includes all the widely used balancing function such as ratio and normalized cut along with the corresponding Cheeger variants.

A general scheme called RatioDCA (see Section 2.1.7) was proposed in [56] to minimize a non-negative ratio of 1-homogeneous, difference of convex functions,

$$\min_{f\in\mathbb{R}^n} \frac{R_1(f)-R_2(f)}{S_1(f)-S_2(f)}$$

Since $\hat{S}$ is non-negative, by the definition of the Lovász extension, one sees that S is also non-negative. The non-negativity of TV and T_0 follow similarly. Moreover, TV, S and T_0 are 1-homogeneous functions and T_0 is already expressed as a difference of convex function; note that $a\max_i\{f_i\}$ and $-a\min_i\{f_i\}$ are convex functions for any non-negative real number a [17]. We see that $F_0^{(\gamma)}$ satisfies the assumptions of RatioDCA. In particular, we have

$$\begin{aligned} R_1(f) &= \mathrm{TV}_G(f) + \gamma\mathrm{TV}_{G^m}(f) + a\gamma\left(\max_i\{f_i\} - \min_i\{f_i\}\right)\\ R_2(f) &= \gamma\mathrm{TV}_{G^c}(f),\ S_1(f) = S(f),\ S_2 \equiv 0 \end{aligned}$$

and the algorithm specialized to our setting is given in Algorithm 3.1.

Algorithm 3.1 Minimization of the ratio $F_0^{(\gamma)}$ using RatioDCA

1: **Initialization:** $f^0 \in \mathbb{R}^n$, $\lambda^0 = F_0^{(\gamma)}(f^0)$
2: **repeat**
3: $f^{t+1} = \underset{\|f\|_2\le 1}{\arg\min}\ \frac{1}{2}\sum_{i,j=1}^n \bar{w}_{ij}\,|f_i - f_j| + \max_i\{f_i\} - \min_i\{f_i\} - \langle f, v^t\rangle$,
where $v^t = \frac{1}{a\gamma}\,(\gamma\, r^t + \lambda^t s^t)$, with $r^t \in \partial\mathrm{TV}_{G^c}(f^t)$, $s^t \in \partial S(f^t)$.
4: $\lambda^{t+1} = F_0^{(\gamma)}(f^{t+1})$
5: **until** $\frac{\left|\lambda^{t+1}-\lambda^t\right|}{\lambda^t} < \epsilon$
6: **Output:** λ^t and f^t

In the algorithm the key part is solving the inner convex problem. In our case it has the form (after rescaling the objective by the constant $\frac{1}{a\gamma}$ for $a > 0, \gamma > 0$),

$$\min_{\|f\|_2 \leq 1} \frac{1}{2} \sum_{i,j=1}^{n} \bar{w}_{ij} |f_i - f_j| + \max_i \{f_i\} - \min_i \{f_i\} - \langle f, v^t \rangle,$$

where $\bar{w}_{ij} = \frac{1}{a\gamma}\left(w_{ij} + \gamma w_{ij}^m\right), v^t = \frac{1}{a\gamma}\left(\gamma\, r^t + \lambda^t s^t\right)$, with $r^t \in \partial \mathrm{TV}_{G^c}(f^t)$, $s^t \in \partial S(f^t)$ and $\lambda^t = F_0^{(\gamma)}(f^t)$. An element of the subdifferential of the Lovász extension of $\hat{S}$ at f^t can be found as (see Lemma 2.7)

$$s_{j_i}^t = \hat{S}\left(\{j_i, \ldots, j_n\}\right) - \hat{S}\left(\{j_{i+1}, \ldots, j_n\}\right), \; i = 1, \ldots, n,$$

where $(j_1, \ldots, j_n)$ is a permutation of V satisfying $f_{j_1}^t \leq f_{j_2}^t \leq \ldots \leq f_{j_n}^t$. The subdifferential of the total variation function is given by (see Lemma 2.6)

$$\partial\left(\mathrm{TV}_{G^c}(f^t)\right)_i = \left\{ \textstyle\sum_{j=1}^n w_{ij}^c u_{ij} \,|\, u_{ij} = -u_{ji}, u_{ij} \in \mathrm{sign}(f_i^t - f_j^t) \right\},$$

where $\mathrm{sign}(x) := +1$, if $x > 0$; -1 if $x < 0$; $[-1, 1]$, if $x = 0$. Note that implicitly we minimize $\hat{F}_0^{(\gamma)}$ over non-constant functions. It is shown in [56], that the RatioDCA algorithm converges to non-constant function if the initialization is non-constant, R_1 is invariant under addition of a constant and r^t, s^t satisfy $\langle r^t, \mathbf{1} \rangle = \langle s^t, \mathbf{1} \rangle = 0$. This holds in our case since

$$0 = \hat{S}(V) = S(\mathbf{1}_V) = \sum_{i=1}^{n} \left(\hat{S}\left(\{j_i, \ldots, j_n\}\right) - \hat{S}\left(\{j_{i+1}, \ldots, j_n\}\right) \right) = \langle s^t, \mathbf{1} \rangle .$$

Similarly one can show that $\langle r^t, \mathbf{1} \rangle = 0$. Hence we can guarantee that our method convergences to non-constant f.

It shown in [56] that one does not need to solve the inner problem to full accuracy to guarantee monotonic descent. In fact, we can stop solving the inner problems as soon as the objective value becomes negative and still maintain the monotonic descent property. In our experiments we found that solving the inner problems to low accuracy initially and then doing more iterations for the latter problems typically gives better results.

3.5.2 Quality guarantee for our method

Although the problem of minimizing $F_0^{(\gamma)}$ is non-convex and hence global convergence is not guaranteed, we have the following quality guarantee for our method.

Theorem 3.5 *Let $(C, \overline{C})$ be any partition of V other than the trivial partition $(\emptyset, V)$ and let $\chi_0 = \mathrm{BCut}(C, \overline{C})$. Moreover $\hat{S}$ satisfy the conditions give in Theorem 3.4. If one uses $\mathbf{1}_C$ as the initialization, then our method either terminates in one iteration or outputs a non-constant solution $\bar{f}$ which yields a partition $(A, \overline{A})$ satisfying*

$$\hat{F}^{(\gamma)}(A) < \hat{F}^{(\gamma)}(C).$$

Moreover, if $(C, \overline{C})$ is consistent (i.e., $\hat{T}(C) = 0$) and $\gamma > \frac{\max_{B \subseteq V} \hat{S}(B)}{\theta} \chi_0$ then A is also consistent and has a strictly smaller balanced cut $\mathrm{BCut}(A, \overline{A}) < \mathrm{BCut}(C, \overline{C})$.

Proof: First note that $F_0^{(\gamma)}$ and $\hat{F}_0^{(\gamma)}$ satisfy $F_0^{(\gamma)}(\mathbf{1}_C) = \hat{F}_0^{(\gamma)}(C)$ for any $C \subseteq V$. Let $f^0 = \mathbf{1}_C$ be the initialization of the RatioDCA algorithm for some $C \neq \emptyset$ and $C \neq V$. It has been shown that RatioDCA either stops in one iteration or outputs a non-constant $\bar{f}$ with a strictly smaller objective (see Proposition 2.5). In our setting this means, if the algorithm does not terminate in one iteration, that $F_0^{(\gamma)}(\bar{f}) < F_0^{(\gamma)}(f^0) = \hat{F}_0^{(\gamma)}(C)$. As shown in theorem 3.3, optimal thresholding of $\bar{f}$ results in a partition $(A, \overline{A})$ satisfying $\hat{F}_0^{(\gamma)}(A) \leq F_0^{(\gamma)}(\bar{f})$ and hence we have

$$\hat{F}_0^{(\gamma)}(A) < \hat{F}_0^{(\gamma)}(C).$$

Since $C \neq \emptyset, C \neq V$ and RatioDCA converges to non-constant $\bar{f}$ (and hence $A \neq \emptyset, A \neq V$), we have

$$\hat{F}^{(\gamma)}(A) = \hat{F}_0^{(\gamma)}(A) < \hat{F}_0^{(\gamma)}(C) = \hat{F}^{(\gamma)}(C).$$

If moreover C is consistent, we have, if not terminated in one iteration, $\hat{F}^{(\gamma)}(A) < \hat{F}^{(\gamma)}(C) = \mathrm{BCut}(C, \overline{C})$. For the chosen value of γ, using a similar argument as in Lemma 3.1, one sees that for any inconsistent subset B, $\hat{F}^{(\gamma)}(B) > \mathrm{BCut}(C, \overline{C})$ and hence A cannot be inconsistent. □

In practice best results are obtained by minimizing $F_0^{(\gamma)}$ for a sequence $\{\gamma^p\}$ of increasing values γ, starting with $\gamma^0 = 0$ (i.e., the unconstrained problem), instead of directly using the value of γ prescribed in Theorem 3.4. This way the optimization problem for each of value γ receives a good initialization, which is the solution obtained for the previous value of γ. This process is iterated until the solution violates no more than the required number of constraints. In our experiments we find the smallest γ that results in a solution satisfying all the constraints. For this, we use the strategy, $\gamma^{p+1} = 2\gamma^p$ with $\gamma^0 = 0$, and $\gamma^1 = 0.1$. Once we find γ^p that results in a consistent solution we do a binary search on the interval $[\gamma^{p-1}, \gamma^p]$ to find the smallest γ yielding a consistent solution.

Remark: Note that Theorem 3.2 and Theorem 3.4 are still valid if one introduces the constraint $f \in [0,1]^n$ in the continuous relaxation. We have shown in [22] that the inner problem in this case can be rewritten as a source-sink minimum cut problem. Thus, solutions of all the inner problems are integral (i.e., in $\{0,1\}^n$). However this blows up the choice for the subgradient of the balancing function S which we need in every iteration of RatioDCA. Note that if $f \in \{0,1\}^n$ has r zero components, then there $r!(n-r)!$ permutations of indices that achieve the ascending order of f and each such permutation yields one subgradient (see Lemma 2.7). On the other hand, using the constraint $\|f\|_2 \leq 1$, we found that the inner problems, at least in the beginning, yield solutions that are not piecewise constant, limiting the size of the subdifferential ∂S. In practice this resulted in better solutions.

3.5.3 Smooth minimization of the inner problem

Now, we discuss how to efficiently solve the inner convex problem arising in RatioDCA algorithm. Recall that the inner problem is given by

$$\min_{\|f\|_2 \leq 1} \frac{1}{2} \sum_{i,j=1}^{n} \bar{w}_{ij} |f_i - f_j| + \max_i \{f_i\} - \min_i \{f_i\} - \left\langle f, v^t \right\rangle, \tag{3.14}$$

where $\bar{w}_{ij} = \frac{1}{a}\left(w_{ij} + \gamma w_{ij}^m\right), v^t = \frac{1}{a\gamma}\left(\gamma\, r^t + \lambda^t s^t\right)$, with $r^t \in \partial \mathrm{TV}_{G^c}(f^t)$, $s^t \in \partial S(f^t)$ and $\lambda^t = F_0^{(\gamma)}(f^t)$. In the following, we will work on the graph $\bar{G}(V, \bar{W})$, where the symmetric weight matrix $\bar{W}$ is defined by $\bar{w}_{ij},\ i, j = 1, \ldots, n$. Note that the edge set of $\bar{G}$ is given by $E = \{(i,j) \mid w_{ij} > 0\} \cup \{(i,j) \mid w_{ij}^m > 0\}$.

The problem (3.14) is a non-smooth convex problem and the standard methods from non-smooth convex optimization provide poor convergence rate of $O(\frac{1}{\sqrt{t}})$, where t is the number of steps [90]. Here we show that the above problem can be efficiently minimized by a first-order optimization scheme achieving convergence rate of $O(\frac{1}{t^2})$, which is optimal for smooth convex optimization problems. In contrast to the inner problem of unconstrained balanced cut (i.e., without must-link and cannot-link constraints), our inner problem (3.14) has additional non-smooth term, $\max_i\{f_i\} - \min_i\{f_i\}$, arising because of cannot-link constraints. By efficiently handling this non-smooth term, we further show that the computational complexity of our method is same as that of [56] solving the unconstrained problem. Moreover, we present a preconditioning technique for this problem since the weights $\bar{w}$ may have large variance, which will adversely affect the convergence of gradient-based optimization methods. The proposed preconditioning technique also works for the inner problem arising in the unconstrained balanced cut problem.

Let us first characterize the set $\mathcal{D}_n = \{z \in \mathbb{R}^n : z = x - y, x, y \in \Delta_n\}$, where Δ_n is the simplex defined as $\Delta_n = \{x \in \mathbb{R}^n,\ \sum_{i=1}^n x_i = 1, x \geq 0, \forall i\}$.

Lemma 3.3 *Let $\mathcal{D}_n = \{z \in \mathbb{R}^n : z = x - y, x, y \in \Delta_n\}$ and $\bar{\Delta}_n = \{z \in \mathbb{R}^n : \langle z, \mathbf{1}_n\rangle = 0, \sum_{i=1}^n \max\{z_i, 0\} \leq 1\}$. The sets $\mathcal{D}$ and $\bar{\Delta}_n$ are equal. Moreover given any element $z \in \bar{\Delta}_n$, one can find x, y such that $z = x - y$ and $x, y \in \Delta_n$.*

Proof: $\mathcal{D}_n \subseteq \bar{\Delta}_n$**:** If $z \in \mathcal{D}_n$, then there exist $x, y \in \Delta_n$ such that $z = x - y$. Then $\sum_{i=1}^n z_i = \sum_{i=1}^n (x_i - y_i) = 0$. Now for the second condition, assume, for the sake of contradiction, that $\sum_{i=1}^n \max\{z_i, 0\} > 1$. Then

$$1 < \sum_{i:x_i > y_i} (x_i - y_i) = \sum_{i:x_i > y_i} x_i - \sum_{i:x_i > y_i} y_i.$$

Since $y_i \geq 0,\ i = 1, \ldots, n$, it should hold that $\sum_{i:x_i > y_i} x_i > 1$, which is a contradiction because $x \in \Delta_n$, thus showing $z \in \bar{\Delta}_n$.

$\bar{\Delta}_n \subseteq \mathcal{D}_n$**:** Let $z \in \bar{\Delta}_n$ and define x and y as

$$x_i = \max\{z_i, 0\} + \frac{1}{n}\left(1 - \sum_{i=1}^n \max\{z_i, 0\}\right)$$
$$y_i = x_i - z_i.$$

Note that $\forall i = 1, \ldots, n$, $x_i \geq \max\{z_i, 0\} \geq 0$ since $\sum_{i=1}^n \max\{0, z_i\} \leq 1$ and

$$\sum_{j=1}^n x_j = \sum_{j=1}^n \left(\max\{0, z_j\}\right) + 1 - \frac{1}{n}\sum_{j=1}^n \left(\sum_{i=1}^n \max\{0, z_i\}\right) = 1,$$

thus $x \in \Delta_n$. Finally, since $x_i \geq \max\{z_i, 0\} \geq z_i,\ y_i \geq 0,\ i = 1, \ldots, n$. Since $\langle z, \mathbf{1}_n\rangle = 0$ and $\langle x, \mathbf{1}_n\rangle = 1$, it holds that $\langle y, \mathbf{1}_n\rangle = 1$, thus showing $y \in \Delta_n$. Thus we have written $z = x - y$, for $x \in \Delta_n, y \in \Delta_n$, completing the proof. □

The decomposition of z into x and y given in the above lemma may not be unique. For example, let $z = x - y$ and for a pair of indices r, s and for some $\delta > 0$, if it holds that $x_r \geq \delta, y_r \geq \delta, x_s \leq 1 - \delta, y_s \leq 1 - \delta$, then there exists another decomposition, $z = \bar{x} - \bar{y}$, where $\bar{x}, \bar{y}$ are same as x, y except at r, s: $\bar{x}_r = x_r - \delta, \bar{x}_s = x_s + \delta, \bar{y}_r = y_r - \delta, \bar{y}_s = y_s + \delta$. It is easy to see that both $\bar{x}, \bar{y}$ are on the simplex Δ_n.

We will now introduce a linear operator to rewrite the total variation term similar to [56]. Let $E \subseteq V \times V$ denote the set of edges of the graph $\bar{G}(V, \bar{W})$, i.e., $E = \{(i,j) \mid \bar{w}_{ij} > 0\}$ and $\overrightarrow{E} = \{(i,j) \in E,\ i < j\}$ denote the directed edges. Since $\bar{W}$ is symmetric, we will directly work with $\overrightarrow{E}$ instead of E thus reducing the space complexity by a factor two. Let $A : \mathbb{R}^{\overrightarrow{E}} \to \mathbb{R}^V$ be a linear operator defined as

$$(A\alpha)_i := \sum_{j:(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \bar{w}_{ji}\alpha_{ji},\ \forall i \in V. \tag{3.15}$$

Lemma 3.4 *The adjoint $A^T : \mathbb{R}^V \to \mathbb{R}^E$ of the linear operator A defined in* (3.15) *is given by*

$$(A^T f)_{ij} = \bar{w}_{ij}(f_i - f_j),\ \forall (i,j) \in \overrightarrow{E} \tag{3.16}$$

and its norm is upper bounded by $\sqrt{2 \max_i \left\{\sum_{j=1}^n \bar{w}_{ij}^2\right\}}$.

Proof: From the definition (3.15) of operator A, we have for any $f \in \mathbb{R}^V$ and $\alpha \in \mathbb{R}^{\overrightarrow{E}}$,

$$\begin{aligned}
\langle f, A\alpha\rangle_{\mathbb{R}^V} &= \sum_{i=1}^n f_i \left(\sum_{j:(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \bar{w}_{ji}\alpha_{ji} \right) \\
&= \sum_{(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} f_i - \sum_{(j,i)\in\overrightarrow{E}} f_i \bar{w}_{ji}\alpha_{ji} \\
&= \sum_{(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} f_i - \sum_{(i,j)\in\overrightarrow{E}} f_j \bar{w}_{ij}\alpha_{ij} = \sum_{(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij}(f_i - f_j)
\end{aligned}$$

Using the definition of the adjoint operator $\langle f, A\alpha\rangle_{\mathbb{R}^V} = \left\langle A^T f, \alpha\right\rangle_{\mathbb{R}^{\overrightarrow{E}}}, \forall f \in \mathbb{R}^V$ and $\alpha \in \mathbb{R}^{\overrightarrow{E}}$, we see that $(A^T f)_{ij} = \bar{w}_{ij}(f_i - f_j),\ \forall (i,j) \in \overrightarrow{E}$. Next, the norm of the operator A can be derived as follows. For any given $\alpha \in \mathbb{R}^{\overrightarrow{E}}$, let $\hat{\alpha} \in \mathbb{R}^E$ be defined as follows.

$$\hat{\alpha}_{ij} = \begin{cases} \alpha_{ij} & i < j \\ -\alpha_{ji} & i > j \end{cases} \quad \forall (i,j) \in E.$$

Since the graph G is assumed to have no self loops, we have by symmetry of W,

$$(A\alpha)_i = \sum_{j:(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ji} = \sum_{\substack{(i,j)\in E,\\ i<j}} \bar{w}_{ij}\alpha_{ij} - \sum_{\substack{(i,j)\in E,\\ i>j}} \bar{w}_{ij}\alpha_{ji} = \sum_{j:(i,j)\in E} \bar{w}_{ij}\hat{\alpha}_{ij}$$

For any $\alpha \in \mathbb{R}^{\vec{E}}$, we have

$$\begin{aligned}\|A\alpha\|_2^2 &= \sum_{i=1}^{n}\left(\sum_{j:(i,j)\in E}\bar{w}_{ij}\hat{\alpha}_{ij}\right)^2 \le \sum_{i=1}^{n}\left(\sum_{j:(i,j)\in E}\bar{w}_{ij}^2\sum_{j:(i,j)\in E}\hat{\alpha}_{ij}^2\right)\\ &\le \max_i\left\{\sum_{j:(i,j)\in E}\bar{w}_{ij}^2\right\}\sum_{i=1}^{n}\sum_{j:(i,j)\in E}\hat{\alpha}_{ij}^2\end{aligned}$$

where in the second step, we used the Cauchy-Schwarz inequality. By noting that

$$\sum_{i=1}^{n}\sum_{j:(i,j)\in E}\hat{\alpha}_{ij}^2 = \sum_{\substack{(i,j)\in E,\\ i<j}}\hat{\alpha}_{ij}^2 + \sum_{\substack{(i,j)\in E,\\ i>j}}\hat{\alpha}_{ij}^2 = \sum_{(i,j)\in\vec{E}}\alpha_{ij}^2 + \sum_{(j,i)\in\vec{E}}(-\alpha_{ji})^2 = 2\|\alpha\|_2^2,$$

we have

$$\frac{\|A\alpha\|_2^2}{\|\alpha\|_2^2} \le 2\max_i\left\{\sum_{j:(i,j)\in E}\bar{w}_{ij}^2\right\}$$

Hence $\|A\|_2^2 = \sup_{\alpha\in\mathbb{R}^{\vec{E}}}\frac{\|A\alpha\|_2^2}{\|\alpha\|_2^2} \le 2\max_i\left\{\sum_{j:(i,j)\in E}\bar{w}_{ij}^2\right\}$. □

Lemma 3.5 *Let $\bar{G}(V,\bar{W})$ be a given graph with symmetric weight matrix $\bar{W}$ and A be a linear operator defined as in* (3.15)*. Then it holds that*

$$\mathrm{TV}_{\bar{G}}(f) = \max_{-1\le\alpha_{ij}\le 1,\ \forall(i,j)\in\vec{E}}\langle f, A\alpha\rangle$$

Proof: Let E be the set of edges of the graph $\bar{G}$. Then we have

$$\begin{aligned}\frac{1}{2}\sum_{(i,j)\in E}\bar{w}_{ij}\,|f_i - f_j| &= \frac{1}{2}\sum_{(i,j)\in E}\max_{-1\le\beta_{ij}\le 1,\ \forall(i,j)\in E}\bar{w}_{ij}(f_i-f_j)\beta_{ij}\\ &= \frac{1}{2}\max_{-1\le\beta_{ij}\le 1,\ \forall(i,j)\in E}\sum_{(i,j)\in E}\bar{w}_{ij}f_i\beta_{ij} - \sum_{(j,i)\in E}\bar{w}_{ji}f_i\beta_{ji},\\ &= \max_{-1\le\beta_{ij}\le 1,\ \forall(i,j)\in E}\sum_{(i,j)\in E}\frac{1}{2}\bar{w}_{ij}f_i(\beta_{ij}-\beta_{ji})\end{aligned}$$

where the last step follows because of the symmetry of $\bar{W}$. Now we introduce new variables $\alpha_{ij} = \frac{1}{2}(\beta_{ij}-\beta_{ji})$, $\forall(i,j)\in E$ in the last optimization problem. Note that there is a one-to-one correspondence between α and β and hence there is no change in the solution of the problem. Moreover, α satisfy anti-symmetric constraint

$$\alpha_{ij} = -\alpha_{ji},\ \forall(i,j)\in\vec{E},$$

where $\overrightarrow{E} = \{(i,j) \in E,\ i < j\}$. Hence we can further rewrite the last problem as

$$\begin{aligned}
\max_{\substack{-1\le\alpha_{ij}\le 1,\ \alpha_{ij}=-\alpha_{ji},\\ \forall(i,j)\in E}} \sum_{(i,j)\in E} \bar{w}_{ij} f_i \alpha_{ij} &= \max_{\substack{-1\le\alpha_{ij}\le 1,\\ \alpha_{ij}=-\alpha_{ji},\ \forall(i,j)\in E}} \sum_{i=1}^{n} \sum_{\substack{j:\ i<j,\\ (i,j)\in E}} \bar{w}_{ij} f_i \alpha_{ij} + \sum_{\substack{j:\ i>j,\\ (i,j)\in E}} \bar{w}_{ij} f_i \alpha_{ij} \\
&\overset{(s)}{=} \max_{\substack{-1\le\alpha_{ij}\le 1,\\ (i,j)\in\overrightarrow{E}}} \sum_{i=1}^{n} \sum_{\substack{j:\ i<j,\\ (i,j)\in E}} \bar{w}_{ij} f_i \alpha_{ij} - \sum_{\substack{j:\ i>j,\\ (j,i)\in E}} \bar{w}_{ji} f_i \alpha_{ji} \\
&= \max_{-1\le\alpha_{ij}\le 1,\ (i,j)\in\overrightarrow{E}} \langle f, A\alpha \rangle
\end{aligned}$$

We have used the symmetry of $\bar{W}$ as well as the anti-symmetric constraint $\alpha_{ij} = -\alpha_{ji},\ \forall (i,j) \in \overrightarrow{E}$, in the step s above. □

We now derive an equivalent formulation for the inner problem (3.14).

Proposition 3.1 *Let $E \subseteq V \times V$ denote the set of edges of the graph $\bar{G}(V, \bar{W})$, $\overrightarrow{E} = \{(i,j) \in E,\ i < j\}$ denote the directed edges. Let $A : \mathbb{R}^{\overrightarrow{E}} \to \mathbb{R}^V$ be the linear operator defined as in (3.15). Moreover, let $\bar{\Delta}_n = \{z \in \mathbb{R}^n : \langle z, \mathbf{1}_n \rangle = 0, \sum_{i=1}^n \max\{z_i, 0\} \le 1\}$. The inner problem (3.14) is equivalent to*

$$\min_{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1} \frac{1}{2} \left\| A\alpha - v^t - P_{\bar{\Delta}_n}\left(A\alpha - v^t\right)\right\|_2^2 =: \Psi(\alpha), \tag{3.17}$$

where $P_{\bar{\Delta}_n}(\cdot)$ is the projection operator defined as

$$P_{\bar{\Delta}_n}(\cdot) = \arg\min_{z\in\bar{\Delta}_n} \frac{1}{2} \|z - \cdot\|^2 .$$

The gradient of the objective function Ψ at α is given by

$$\nabla\Psi(\alpha) = A^T \left(A\alpha - v^t - P_{\bar{\Delta}_n}(A\alpha - v^t)\right),$$

where the adjoint A^T is given in Lemma 3.4. Moreover, the Lipschitz constant of the gradient of Ψ is upper bounded by $2\left\|A^T A\right\| \le 4\ \max_i \sum_{j=1}^n \bar{w}_{ij}^2$.

Proof: Note that $\max_i\{f_i\} = \max_{x\in\Delta_n} \langle x, f\rangle$, where Δ_n is the simplex defined as $\Delta_n = \{x \in \mathbb{R}^n,\ \sum_{i=1}^n x_i = 1, x \ge 0, \forall i\}$. Thus the inner problem (3.14) can be rewritten using Lemma 3.5 as

$$\begin{aligned}
&\min_{\|f\|_2\le 1} \max_{\{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1\}} \langle f, A\alpha\rangle + \max_{x\in\Delta_n} \langle f, x\rangle + \max_{y\in\Delta_n} \langle -f, y\rangle - \left\langle f, v^t\right\rangle \\
&= \min_{\|f\|_2\le 1} \max_{\substack{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1,\\ x,y\in\Delta_n}} \left\langle f, A\alpha + x - y - v^t\right\rangle \\
&\overset{(s_1)}{=} \max_{\substack{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1,\\ x,y\in\Delta_n}} \min_{\|f\|_2\le 1} \left\langle f, A\alpha + x - y - v^t\right\rangle \\
&\overset{(s_2)}{=} \max_{\substack{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1,\\ x,y\in\Delta_n}} - \left\|A\alpha + x - y - v^t\right\|_2
\end{aligned}$$

The step s_1 follows from the standard min-max theorem (see Corollary 37.3.2 in [98]) since x, y, α, and f lie in non-empty compact convex sets. In the step s_2, we used that the minimizer of the linear function over the Euclidean ball is given by

$$f^* = -\frac{A\alpha + x - y - v^t}{\|A\alpha + x - y - v^t\|_2},$$

if $\|A\alpha + x - y - v^t\|_2 \neq 0$; otherwise f^* is an arbitrary element of the Euclidean unit ball. Using Lemma 3.3 the preceding problem can be further rewritten as

$$\min_{\substack{\alpha\in\mathbb{R}^{\vec{E}},\ \|\alpha\|_\infty\leq 1,\\ x,y\in\Delta_n,\ z=y-x}} \frac{1}{2}\left\|A\alpha - v^t - z\right\|_2^2 = \min_{\substack{\alpha\in\mathbb{R}^{\vec{E}},\ \|\alpha\|_\infty\leq 1,\\ z\in\bar{\Delta}_n}} \frac{1}{2}\left\|A\alpha - v^t - z\right\|_2^2.$$

Noting that the minimization over z is a projection problem yields the problem stated in the proposition. It is shown in [100] that the function $\frac{1}{2}\|x - P_C(x)\|^2$, where $P_C(\cdot)$ is the projection onto the convex set C, is differentiable and its gradient is given by $x - P_C(x)$. Thus Ψ is differentiable and using the chain rule, its gradient is then given by

$$\nabla\Psi(\alpha) = A^T\left(A\alpha - v^t - P_{\bar{\Delta}_n}(A\alpha - v^t)\right).$$

Finally, we derive the Lipschitz constant of the gradient as follows. For any $\alpha, \beta \in \mathbb{R}^{\vec{E}}$, we have

$$\begin{aligned}
\|\nabla\Psi(\alpha) - \nabla\Psi(\beta)\| &= \left\|A^T(A\alpha - A\beta) + A^T\left(P_{\bar{\Delta}_n}(A\beta - v^t) - P_{\bar{\Delta}_n}(A\alpha - v^t)\right)\right\| \\
&\leq \left\|A^T(A\alpha - A\beta)\right\| + \left\|A^T\left(P_{\bar{\Delta}_n}(A\beta - v^t) - P_{\bar{\Delta}_n}(A\alpha - v^t)\right)\right\| \\
&\leq \left\|A^TA\right\|\|\alpha - \beta\| + \left\|A^T\right\|\left\|P_{\bar{\Delta}_n}(A\beta - v^t) - P_{\bar{\Delta}_n}(A\alpha - v^t)\right\| \\
&\leq \left\|A^TA\right\|\|\alpha - \beta\| + \left\|A^T\right\|\|A\beta - A\alpha\| \\
&\leq 2\left\|A^TA\right\|\|\alpha - \beta\|.
\end{aligned}$$

Here the first inequality follows because norm satisfies the triangle inequality, second inequality follows from the definition of the norm, third inequality follows from the non-expansiveness property of the projection operator [99] and the last inequality follows from the fact that $\|A\| = \left\|A^T\right\| = \sqrt{\|A^TA\|}$. □

We have derived an equivalent optimization problem where the objective function has Lipschitz continuous gradient and projection on to the constraint set $\{\alpha \in \mathbb{R}^{\vec{E}},\ \|\alpha\|_\infty \leq 1\}$ can be efficiently computed. Hence we can use accelerated projected gradient methods such as [15, 89, 91] for solving this problem and achieve optimal rate of convergence. However, to evaluate the objective and compute its gradient, we need to perform projection onto the set $\bar{\Delta}_n$. Now, we show that this projection can be computed in linear time $O(n)$ and involves finding two different projections onto the simplex Δ_n and one projection onto the orthogonal complement of the space spanned by the one vector $\mathbf{1}_n$.

Proposition 3.2 *Given $a \in \mathbb{R}^n$, an optimal solution (x^*, y^*) of the optimization problem*

$$\operatorname*{arg\,min}_{x\in\Delta_n,\ y\in\Delta_n} \frac{1}{2}\|x - y - a\|_2^2 \tag{3.18}$$

can be computed in $O(n)$ time. In fact it holds that

- *if* $\langle x^*, y^* \rangle = 0$, *then*

$$x^* = P_{\Delta_n}(a) = \underset{x \in \Delta_n}{\arg\min} \frac{1}{2} \|x - a\|_2^2$$
$$y^* = P_{\Delta_n}(-a) = \underset{y \in \Delta_n}{\arg\min} \frac{1}{2} \|y - (-a)\|_2^2$$

- *if* $\langle x^*, y^* \rangle > 0$, *then*

$$\forall i = 1, \ldots, n,\ x_i^* = \max\{z_i, 0\} + \frac{1}{n}\left(1 - \sum_{i=1}^{n} \max\{z_i, 0\}\right),$$
$$\forall i = 1, \ldots, n,\ y_i^* = x_i^* - z_i,\ \text{where } z = a - \frac{1}{n}\langle a, \mathbf{1}_n \rangle \mathbf{1}_n.$$

Proof: The Lagrangian of the constrained problem (3.18) is given by

$$L(x, y, \lambda, \bar{\lambda}, \mu, \bar{\mu}) = \frac{1}{2}\|x - y - a\|_2^2 - \langle \lambda, x \rangle - \langle \bar{\lambda}, y \rangle + \mu(\langle x, \mathbf{1}_n \rangle - 1) + \bar{\mu}(\langle y, \mathbf{1}_n \rangle - 1),$$

where $\lambda, \bar{\lambda} \in \mathbb{R}^n$ and $\mu, \bar{\mu} \in \mathbb{R}$. Note that the problem satisfies the Slater's condition. Hence the necessary and sufficient KKT optimal conditions for a minimizer are given by [17]

$$\begin{aligned}
x - y - a - \lambda + \mu \mathbf{1}_n &= 0, && \text{(Stationarity)}\\
-x + y + a - \bar{\lambda} + \bar{\mu} \mathbf{1}_n &= 0, && \text{(Stationarity)}\\
\langle x, \mathbf{1}_n \rangle = 1,\ \langle y, \mathbf{1}_n \rangle &= 1, && \text{(Primal feasibility)}\\
x_i \geq 0,\ y_i \geq 0,\ i &= 1, \ldots, n, && \text{(Primal feasibility)}\\
\lambda_i \geq 0,\ \bar{\lambda} \geq 0,\ i &= 1, \ldots, n, && \text{(Dual feasibility)}\\
\lambda_i x_i = 0,\ \bar{\lambda}_i y_i = 0,\ i &= 1, \ldots, n. && \text{(Complementary slackness)}
\end{aligned}$$

We have from the stationarity conditions,

$$x - y = a + \lambda - \mu \mathbf{1}_n, \quad x - y = a - \bar{\lambda} + \bar{\mu} \mathbf{1}_n.$$

This implies

$$\lambda + \bar{\lambda} = (\mu + \bar{\mu}) \mathbf{1}_n. \tag{3.19}$$

Let $(x^*, y^*, \lambda^*, \bar{\lambda}^*, \mu^*, \bar{\mu}^*)$ be an optimal primal-dual pair satisfying the above conditions. Then we distinguish two cases based on the value of $\langle x^*, y^* \rangle$. Since $x_i^* \geq 0, y_i^* \geq 0, \forall i$, $\langle x^*, y^* \rangle$ is either zero or strictly positive.

case (i): If $\langle x^*, y^* \rangle = 0$, then $x_i^* y_i^* = 0, \forall i$, since $x_i^* \geq 0$ and $y_i^* \geq 0$. Now we claim that $(x^*, \lambda^* + y^*, \mu^*)$ is an optimal primal-dual pair for the projection problem,

$$P_{\Delta_n}(a) = \underset{x \in \Delta_n}{\arg\min} \frac{1}{2}\|x - a\|_2^2 .$$

Let ξ be Lagrange multipliers for the non-negative constraints and ν for the linear constraint on x. The necessary and sufficient optimality conditions any primal-dual pair (x, ξ, ν) must satisfy are

$$\begin{aligned} x - a - \xi + \nu \mathbf{1}_n &= 0, && \text{(Stationarity)} \\ \langle x, \mathbf{1}_n \rangle = 1,\ x_i \geq 0,\ i = 1, \ldots, n, & && \text{(Primal feasibility)} \\ \xi_i &\geq 0, && \text{(Dual feasibility)} \\ \xi_i x_i &= 0,\ i = 1, \ldots, n. && \text{(Complementary slackness)} \end{aligned}$$

Clearly, $(x^*, \lambda^* + y^*, \mu^*)$ satisfies these conditions since $\langle x^*, y^* \rangle = 0$ and (x^*, y^*) is optimal for the problem (3.18). Similarly, one can show that $(y^*, \bar{\lambda}^* + x^*, \bar{\mu}^*)$ satisfy the KKT optimality conditions for the problem of projecting $-a$ onto the simplex.

Case (ii:) Let $\langle x^*, y^* \rangle > 0$. Then there exists an index r such that $x_r^* > 0, y_r^* > 0$ and complementary slackness conditions for problem (3.18) imply $\lambda_r^* = 0, \bar{\lambda}_r^* = 0$. Thus, we have from (3.19)

$$\mu^* + \bar{\mu}^* = 0.$$

On the other hand for all $i \neq r$, we still have $\lambda_i^* + \bar{\lambda}_i^* = \mu^* + \bar{\mu}^* = 0$. Since $\lambda_i^* \geq 0, \bar{\lambda}_i^* \geq 0$, this can hold only if both are zero. Thus we have $\lambda_i^* = 0, \bar{\lambda}_i^* = 0, \forall i$. This allows us to find the optimal μ^* from the primal feasibility condition:

$$1 = \langle x^*, \mathbf{1}_n \rangle = \langle y^*, \mathbf{1}_n \rangle + \langle a, \mathbf{1} \rangle + 0 - \mu^* n.$$

This gives us

$$\mu^* = \frac{1}{n} \langle a, \mathbf{1}_n \rangle .$$

Having derived the optimal Lagrange multipliers, one obtains

$$x^* - y^* = a - \frac{1}{n} \langle a, \mathbf{1}_n \rangle \mathbf{1}_n.$$

Let $z = x^* - y^*$. Since $x^* \in \Delta_n, y^* \in \Delta_n$ it must hold, by Lemma 3.3, that $\sum_{i=1}^n \max\{0, z_i\} \leq 1$. Moreover, one can recover x^*, y^* from z using Lemma 3.3.

Since projection onto the simplex Δ_n can be done in linear time [69], the computation of (x^*, y^*) in both cases takes linear time. Since the two cases are complementary, one just needs to compute (x, y) using both ways and then choose the one that has a smaller objective value and satisfies all the conditions (in particular $\langle x, y = 0 \rangle$ for case (i) and $\sum_{i=1}^n \max\{0, x_i - y_i\} \leq 1$ for case (ii)). □

Using Lemma 3.3 we can restate the above result as follows.

Proposition 3.3 *For any given $a \in \mathbb{R}^n$, the orthogonal projection z^* onto the set $\bar{\Delta}_n = \{z \in \mathbb{R}^n : \langle z, \mathbf{1}_n \rangle = 0, \sum_{i=1}^n \max\{z_i, 0\} \leq 1\}$*

$$z^* = \arg\min_{z \in \bar{\Delta}_n} \|z - a\|^2$$

can be computed in $O(n)$ time.

We again remark that the orthogonal projection z is unique; however the solution to the problem (3.18) need not be unique; see the discussion following Lemma 3.3.

3.5.4 Preconditioning for the inner problem

The inner convex problem (3.14) has been rewritten as the following optimization problem where the objective function as well as the constraint functions are smooth,

$$\min_{\alpha\in\mathbb{R}^{\overrightarrow{E}},\ \|\alpha\|_\infty\le 1} \frac{1}{2}\left\|A\alpha - v^t - P_{\bar{\Delta}_n}\left(A\alpha - v^t\right)\right\|_2^2, \tag{3.20}$$

where A is the linear operator defined in (3.15), $\bar{\Delta}_n = \{z \in \mathbb{R}^n : \langle z, \mathbf{1}_n\rangle = 0, \sum_{i=1}^n \max\{z_i, 0\} \le 1\}$ and $P_{\bar{\Delta}_n}(\cdot)$ is the orthogonal projection onto the convex set $\bar{\Delta}_n$.

If the weights $\bar{w}_{ij}$ vary a lot, then the operator A introduces a non-uniform scaling of the variables α, which adversely affects gradient-based methods. Here we propose a preconditioning technique that is based on transferring the problem to an unweighted graph (i.e., edges with unit weights). Let $\bar{G}_0(V, \bar{W}_0)$ be the unweighted version of the graph $\bar{G}(V, \bar{W})$, i.e., $(\bar{w}_0)_{ij} = 1$ if $\bar{w}_{ij} > 0$ and $(\bar{w}_0)_{ij} = 0$ otherwise. Note that the edge sets $E = \{(i,j) \mid \bar{w}_{ij} > 0\}$ and $\overrightarrow{E} = \{(i,j) \in E,\ i < j\}$ are same for both G and G_0. Let $B : \mathbb{R}^{\overrightarrow{E}} \to \mathbb{R}^V$ be a linear operator defined on the graph $\bar{G}_0(V, \bar{W}_0)$ as

$$(B\beta)_i := \sum_{j:(i,j)\in\overrightarrow{E}} \beta_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \beta_{ji},\ \forall i \in V. \tag{3.21}$$

Lemma 3.6 *The inner problem* (3.20) *is equivalent to*

$$\min_{\substack{\beta\in\mathbb{R}^{\overrightarrow{E}},\\ -\bar{w}_{ij}\le\beta_{ij}\le\bar{w}_{ij},\ \forall(i,j)\in\overrightarrow{E}}} \frac{1}{2}\left\|B\beta - v^t - P_{\bar{\Delta}_n}\left(B\beta - v^t\right)\right\|_2^2 =: \tilde{\Psi}(\beta), \tag{3.22}$$

where B is the linear operator defined as in (3.21)*. The Lipschitz constant of the gradient of $\tilde{\Psi}$ is upper bounded by $2\left\|B^TB\right\| \le 4\ \max_i\{\mathcal{N}(i)\}$, where $\mathcal{N}(i) = |\{(i,j) \mid (i,j) \in E\}|$ is the number of neighbors of vertex i in the graph $\bar{G}_0$.*

Proof: We derive an equivalent formulation by rescaling the variables: $\beta_{ij} = \bar{w}_{ij}\alpha_{ij}$, $\forall(i,j) \in \overrightarrow{E}$. Since the mapping between α and β is one-to-one, the transformation yields an equivalent problem (in the sense that minimizer of one problem can be easily derived from minimizer of the other problem). With this mapping, the constraints of the original problem (3.20) become $-w_{ij} \le \beta_{ij} \le w_{ij}$. Moreover,

$$(A\alpha)_i = \sum_{j:(i,j)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \bar{w}_{ij}\alpha_{ji} = \sum_{j:(i,j)\in\overrightarrow{E}} \beta_{ij} - \sum_{j:(j,i)\in\overrightarrow{E}} \beta_{ji} = (B\beta)_i,$$

which shows the equivalence. The bound on the Lipschitz constant can be derived in a similar fashion as in the proof of Proposition 3.1 and the last equality follows from Lemma 3.4. □

The main idea behind our preconditioning technique is to find a suitable change of variables such that $A\alpha = B\beta$ where B eliminates the issue of non-uniform scaling present in A. In our case, a simple change of variables solved the issue of non-uniform edge weights by transferring the problem into an unweighted graph. Note

that if the number of neighbors in the unweighted graph vary a lot then the operator B would still the have the same issue. Ideally one would like to find variable substitution $A\alpha = B\beta$ such that $\|B\|$ is close to 1. If A^TA is invertible then one can use the variable substitution $\alpha = (A^TA)^{-\frac{1}{2}}\beta$ to achieve this. However computing the inverse can be computationally prohibitive as in our case. Hence preconditioning methods have to trade-off between finding an ideal variable substitution and being computationally feasible. In our experiments, where we used K-NN graphs, the proposed preconditioning already improved the performance significantly. A possible future work would further explore the preconditioning techniques for graphs where the number of neighbors vary a lot.

In our implementation, we use FISTA (Fast Iterative Shrinkage-Thresholding Algorithm) [15] for solving the preconditioned inner problem (3.22). FISTA is an accelerated first-order method for minimizing a sum of two continuous convex functions F and G,

$$\min_{x\in\mathbb{R}^n} F(x) + G(x), \tag{3.23}$$

where F is convex and continuously differentiable with Lipschitz continuous gradient with the constant L while G is (possibly non-smooth) convex function. The main step of the algorithm, given below, requires the computation of the so-called proximal map with respect to G

$$\text{prox}_{\frac{1}{L}G}(\cdot) := \underset{y\in\mathbb{R}^n}{\arg\min}\, G(y) + \frac{L}{2}\|y - \cdot\|_2^2.$$

Algorithm 3.2 Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [15]

Input: Lipschitz constant L of ∇F
Initialization: $x^0 \in \mathbb{R}^n$, $\bar{x}^0 = x^0$, $t_1 = 1,\ p = 0$.
repeat
  $x^{p+1} = \text{prox}_{\frac{1}{L}G}(\bar{x}^p - \frac{1}{L}\nabla F(\bar{x}^p))$
  $t_{p+1} = \frac{1+\sqrt{1+4t_p^2}}{2}$
  $\bar{x}^{p+1} = x^{p+1} + \frac{t_p - 1}{t_{p+1}}(x^{p+1} - x^p)$
  $p = p + 1$
until convergence
Output: β^p

Comparing with the general model (3.23), we identify for our inner problem (3.22)

$$F(\cdot) = \frac{1}{2}\left\|B\cdot - v^t - P_{\bar{\Delta}_n}\left(B\cdot - v^t\right)\right\|_2^2, \quad G(\cdot) = \iota_{\mathcal{B}}(\cdot),$$

where $\mathcal{B} = \{\beta \in \mathbb{R}^{\overrightarrow{E}},\ -\bar{w}_{ij} \le \beta_{ij} \le \bar{w}_{ij},\ \forall (i,j) \in \overrightarrow{E}\}$ and $\iota_{\mathcal{B}}$ is the indicator function defined as

$$\iota_{\mathcal{B}}(\beta) = \begin{cases} 0, & \beta \in \mathcal{B} \\ \infty & \text{otherwise} \end{cases} \tag{3.24}$$

The gradient of F at β is $B^T(B\beta - v^t - P_{\bar{\Delta}_n}(B\beta - v^t))$, where the adjoint of B is given by (see Lemma 3.4)

$$(B^T f)_{ij} = f_i - f_j.$$

The Lipschitz constant L of the gradient of F is given in Lemma 3.6. Thus the proximal problem (Step 4 of Algorithm 3.2) one has to solve in our case is given by

$$\beta^{p+1} = \underset{\substack{\beta \in \mathbb{R}^{\overrightarrow{E}}, \\ -\bar{w}_{ij} \leq \beta_{ij} \leq \bar{w}_{ij},\ \forall (i,j) \in \overrightarrow{E}}}{\arg\min} \frac{1}{2} \|\beta - c^p\|_2^2$$

where $c^p = \bar{\beta}^p - \frac{1}{L} B^T \left(B\bar{\beta}^p - v^t - P_{\bar{\Delta}_n}\left(B\bar{\beta}^p - v^t\right)\right)$. This problem can be solved in closed-form for each variable independently

$$\beta_{ij}^{p+1} = P_{[-w_{ij}, w_{ij}]}(c_{ij}^p) = \max\{-\bar{w}_{ij}, \min\{w_{ij}, c_{ij}^p\}\},\ \forall (i,j) \in \overrightarrow{E},$$

where $P_{[-w_{ij}, w_{ij}]}(\cdot)$ denotes the projection onto the interval $[-w_{ij}, w_{ij}]$. The complete details are given in Algorithm 3.3. The main computational time depends on step 4

Algorithm 3.3 Solution of the preconditioned inner problem (3.22) using FISTA

1: **Input:** Lipschitz constant L of $\nabla\tilde{\Psi}$
2: **Initialization:** $\beta^0 \in \mathbb{R}^{\overrightarrow{E}}$, $\bar{\beta}^0 = \beta^0$, $t_1 = 1,\ p = 0$.
3: **repeat**
4: $\quad c^p = \bar{\beta}^p - \frac{1}{L} B^T \left(B\bar{\beta}^p - v^t - P_{\bar{\Delta}_n}\left(B\bar{\beta}^p - v^t\right)\right)$
5: $\quad \beta_{ij}^{p+1} = \max\{-\bar{w}_{ij}, \min\{w_{ij}, c_{ij}^p\}\},\ \forall (i,j) \in \overrightarrow{E}$
6: $\quad t_{p+1} = \frac{1+\sqrt{1+4t_p^2}}{2}$
7: $\quad \bar{\beta}^{p+1} = \beta^{p+1} + \frac{t_p - 1}{t_{p+1}}(\beta^{p+1} - \beta^p)$
8: $\quad p = p + 1$
9: **until** convergence
10: **Output:** β^p

where in every iteration we need to project onto the set $\bar{\Delta}_n$ apart from applying two linear operators B and B^T. As we have shown in the Proposition 3.3, the projection can be done in linear time $O(|V|)$ while the operators B and B^T take $O(\left|\overrightarrow{E}\right|)$ time. Thus each iteration takes linear time $O(|V| + \left|\overrightarrow{E}\right|)$ and the number of iterations required by FISTA for an ϵ-optimal solution is $O(\frac{1}{\sqrt{\epsilon}})$, which is optimal for the class of minimizing smooth convex functions. Practically, we can check convergence by examining the gap between the objective values of the dual problem (3.17) and the original primal problem (3.14). Note that the solution of (3.14) can be obtained from the solution α^* of (3.17) as follows (see Proposition 3.1)

$$f^* = -\frac{A\alpha^* - z^* - v^t}{\|A\alpha^* - z^* - v^t\|_2},\ \text{if}\ \left\|A\alpha^* + z^* - v^t\right\|_2 \neq 0,$$

where $z^* = P_{\bar{\Delta}_n}(A\alpha^* - v^t)$. If $\|A\alpha^* - z^* - v^t\| = 0$, then any f satisfying $\|f\|_2 \leq 1$ is a solution for (3.14). We note that one does not need to explicitly compute the

solution α^* from the optimal solution β^* of the preconditioned problem (3.22) since f^* can be directly computed as

$$f^* = -\frac{B\beta^* - z^* - v^t}{\|B\beta^* - z^* - v^t\|}, \text{ if } \left\|B\beta^* + z^* - v^t\right\| \neq 0,$$

where $z^* = P_{\bar{\Delta}_n}(B\beta^* - v^t)$ as it holds that $(A\alpha)_{ij} = (B\beta)_{ij},\ \forall (i,j) \in \overrightarrow{E}$.

3.6 Experiments

3.6.1 Constrained clustering

We compare our method against the following four related constrained clustering approaches: spectral learning (SL) [64], flexible constrained spectral clustering (CSP) [117], constrained clustering via spectral regularization (CCSR) [79] and spectral clustering with linear constraints (SCLC) [121]. We refer the reader to Section 3.1 for more details of these methods. For CSP and CCSR we use the code provided by the authors with default parameters. Since ours is a non-convex method, we run our method, in parallel, from 9 different random initializations and the solution of the standard spectral clustering. In order to compare against the existing work, in our experiments we use normalized cut as the clustering objective:

$$\mathrm{NCut}(C, \overline{C}) = \frac{\mathrm{cut}(C, \overline{C})}{\mathrm{vol}(C)} + \frac{\mathrm{cut}(C, \overline{C})}{\mathrm{vol}(\overline{C})} = \frac{\mathrm{cut}(C, \overline{C})}{\mathrm{vol}(C)\,\mathrm{vol}(\overline{C})}\,\mathrm{vol}(V).$$

Since $\mathrm{vol}(V)$ is constant for a given graph, we solve the balanced cut problem with $\hat{S}(C) = \mathrm{vol}(C)\,\mathrm{vol}(\overline{C})$ and rescale the obtained balanced cut value by $\mathrm{vol}(V)$.

Datasets: We evaluate our method on several UCI two-class classification datasets. The summary of these datasets is given in Table 3.1. The data with missing values are removed and redundant data points are removed from the spam dataset. We build a symmetric K-NN similarity graph from the data $X = (X_1, \ldots, X_n), X_i \in \mathbb{R}^d$ as in [21],

$$w_{ij} = \exp\left(-\frac{4\,\|x_i - x_j\|_2^2}{\min\{\sigma_i^2, \sigma_j^2\}}\right),$$

where σ_i is the Euclidean distance of x_i to its K^{th}-nearest neighbor. In all the experiments we ue $K = 10$. In order to illustrate the effects of highly unbalanced problems, we create a two-class problem (digit 0 versus rest) from USPS dataset.

Constraint generation: The pairwise constraints are generated from the class labels Y as follows. We first fix the number of pairwise constraints n_C to be generated. Then we randomly sample n_C pairs of points and for each pair, we introduce either a cannot-link or a must-link constraint depending on the labels of the sampled pair. We evaluate the methods for various values of n_C. We maintain the property that the constraint sets are nested; i.e., if S_1 and S_2 are the constraints generated for a given dataset for two different values of n_C, then $S_1 \subset S_2$ whenever $|S_1| < |S_2|$. This is achieved by first sampling the maximum number of pairs for each dataset and then drawing the required number of constraints from this pool.

Dataset	Size	Class proportions
Sonar	208	97:111
Breast Cancer	263	77:186
Heart	270	120:150
WDBC	569	212:357
Diabetis	768	268:500
Spam	4207	1676:2531
USPS (0 vs rest)	9298	1553:7745

Table 3.1: The UCI classification datasets used in the experiments.

Evaluation criteria: Our evaluation is based on the following three criteria: clustering error, normalized cut and fraction of constraints violated. Since we are using classification datasets, we evaluate the quality of clustering result based on how well each cluster agrees with the known class structure. For this, we label each cluster based on the ground-truth class labels of the points in the cluster; in particular each cluster receives the majority class label of the points in that cluster. Then the clustering error is computed as the error of this labeling. More precisely, let $(C, \overline{C})$ be the clustering result and Y_i denote the true class label of the data point i. Moreover let y'_1 and y'_2 be the dominant class labels in C and $\overline{C}$ respectively. Then the error of $(C, \overline{C})$ is computed as

$$\text{error}(C, \overline{C}) = \frac{1}{n} \left(\sum_{i \in C} \delta_{Y_i \neq y'_1} + \sum_{i \in \overline{C}} \delta_{Y_i \neq y'_2} \right),$$

where n is the total number of points and $\delta_c = 1$, if the condition specified in c is true and 0 otherwise.

The results, averaged over 10 trials are shown in Tables 3.2 and 3.3. The runtimes are shown in Table 3.4. In the plots our method is denoted as COSC and we enforce all constraints. Since our formulation is a non-convex problem, we use the best result (based on the achieved cut value) of 10 runs with 9 random initializations and one initialization based on the standard spectral clustering. The first column of Tables 3.2 and 3.3 shows the clustering error versus the number of pairwise constraints. In terms of clustering error, ours is the only method that consistently outperforms every other method. As discussed in Section 3.1, the method SCLC performs very poorly when the classes are highly unbalanced; see the results for USPS. However unlike the rest of the competing methods, SCLC performance is at least consistent across the datasets (except for the highly unbalanced USPS dataset). The code available for the method CSP could not run on the USPS dataset. Next in terms of normalized cut, reported on the second column of Tables 3.2 and 3.3, we see that our method produces always much better cuts than the ones found by SCLC, which is the only other method that returned solutions satisfying all the constraints. All other methods never satisfied all the given constraints; see the third column of Tables 3.2 and 3.3 where we reported the fraction of violated constraints for every method. One notices that SL and CCSR could not enforce many of the given constraints and consequently their performance in terms of the clustering error is very bad in four out of the seven datasets.

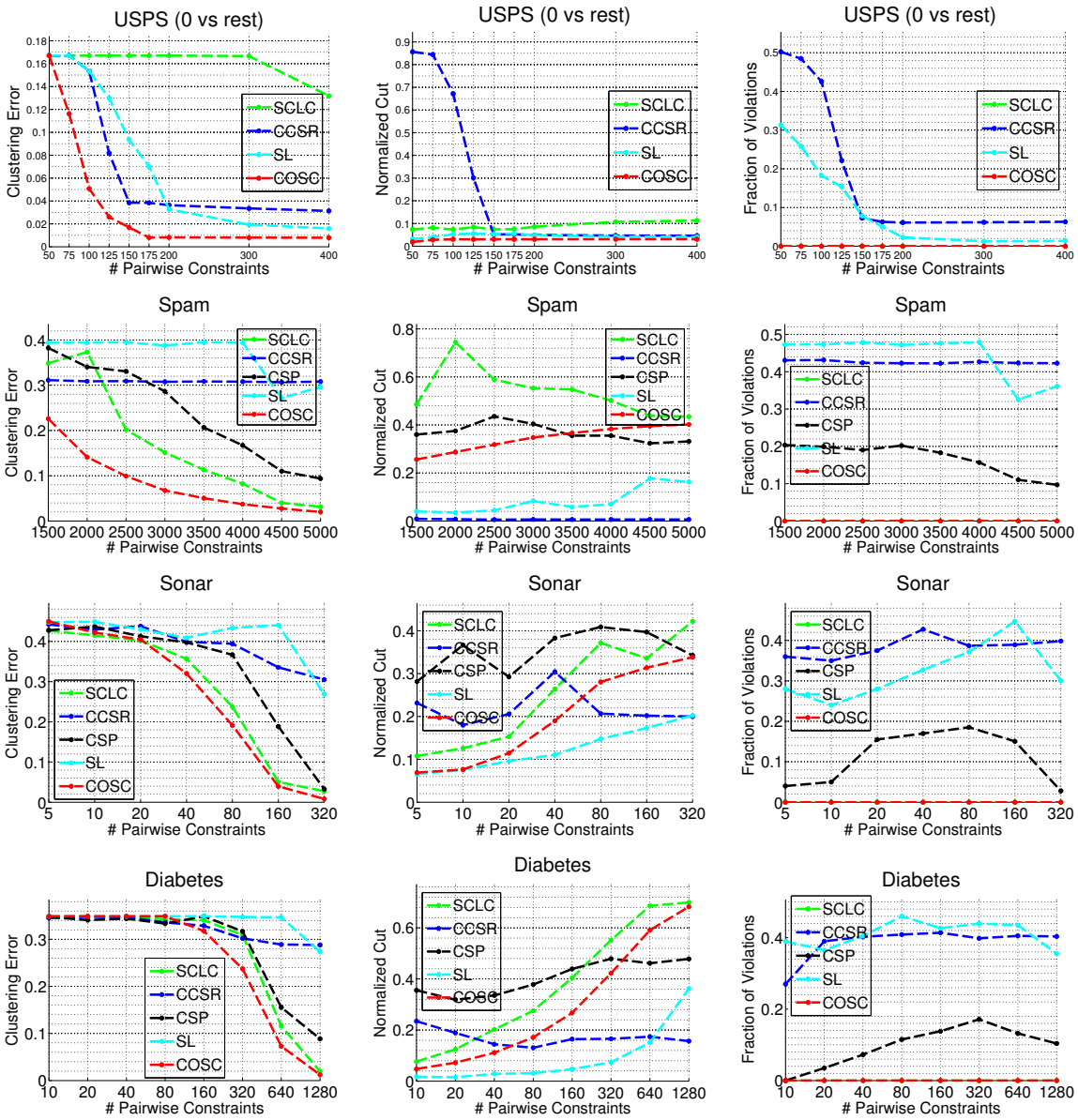


Table 3.2: Results for **two-class constrained clustering**: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

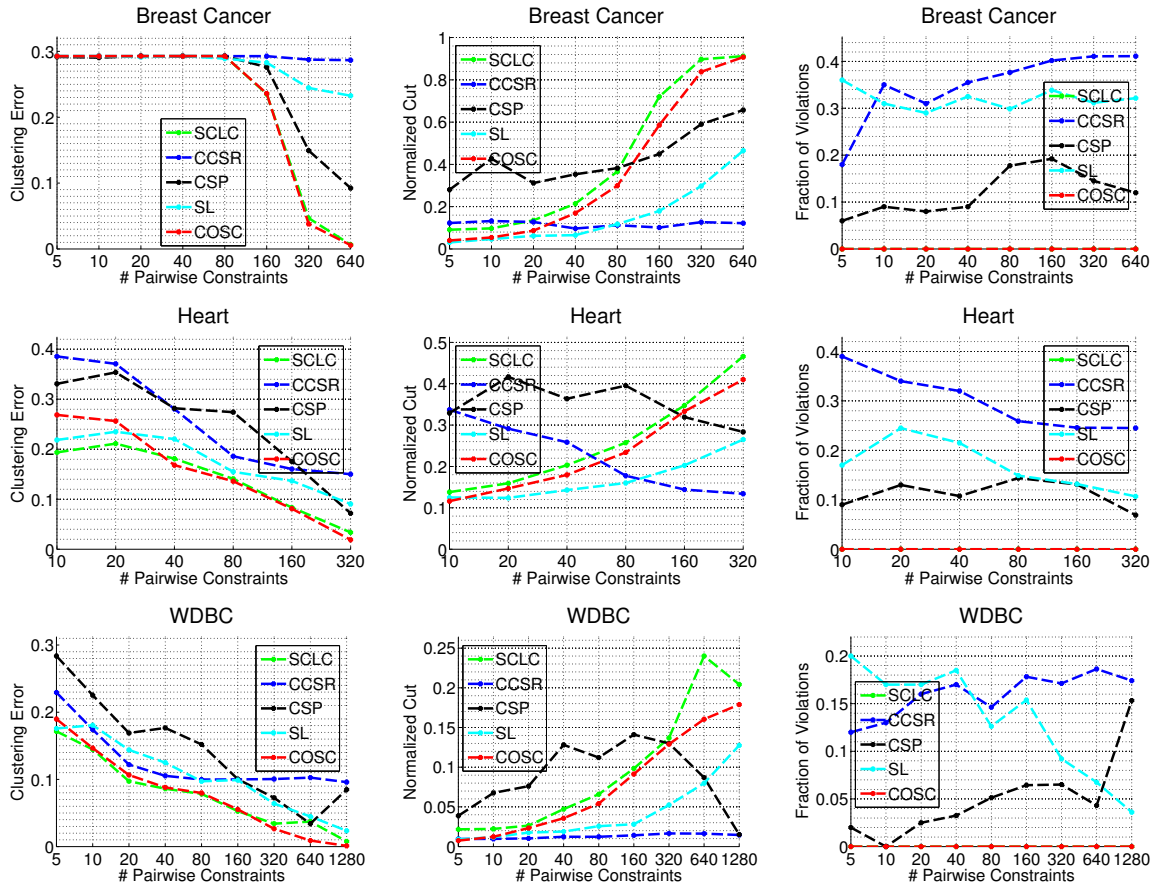


Table 3.3: Results for **two-class constrained clustering**: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

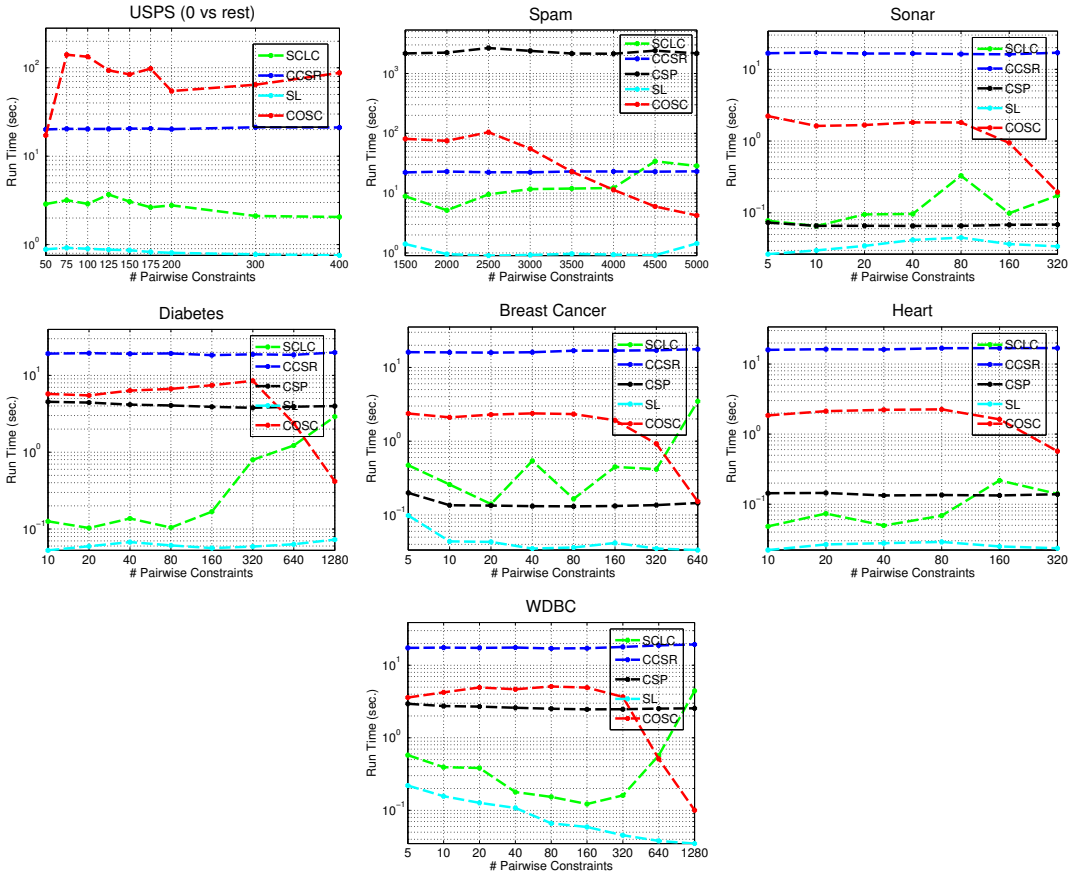


Table 3.4: Results for **two-class constrained clustering**: Runtimes versus number of constraints.

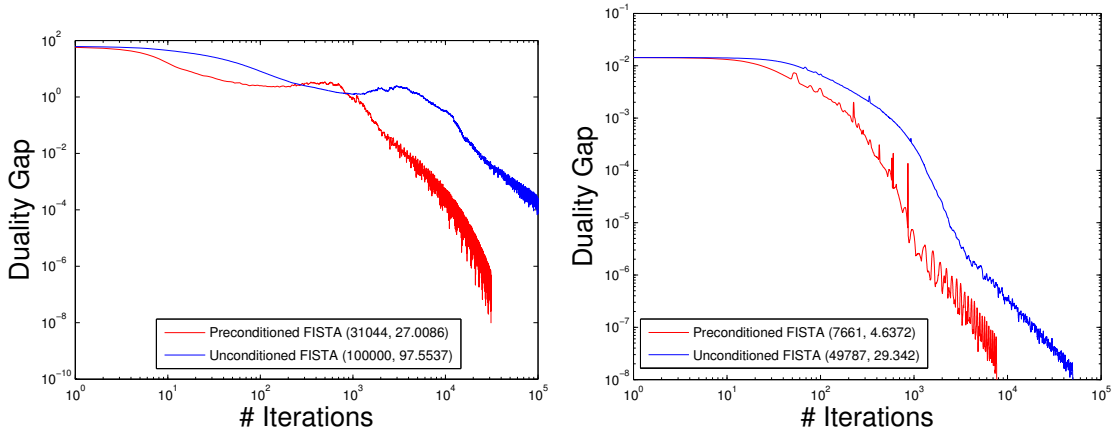


Figure 3.2: Convergence of FISTA on unconditioned and preconditioned versions of the inner problem. Duality gap is plotted against the number of iterations on log-log scale. The legend box also shows the number of iterations and the time taken until convergence (up to the accuracy 10^{-8}) or the maximum number of iterations (10^5) is reached.

3.6.2 Effectiveness of preconditioning

Here we evaluate the effectiveness of our preconditioning technique presented in Section 3.5.4. For this, we ran FISTA on unconditioned and preconditioned inner problems (3.17) and (3.22) respectively. The Figure 3.2 shows the performance of FISTA on two different problem instances. The first instance corresponds to an inner problem on the USPS dataset for $\gamma = 0.2$ whereas the second instance corresponds to that of Spam for $\gamma = 10$. The plots show the duality gap against the number of iterations on log-log scale. Here the duality gap refers to the difference between the objective value of the primal problem (3.14) and the optimal value returned by FISTA which solves the dual problems (3.17) and (3.22). We also show in the plots the number of iterations and the total time taken by each method until convergence or the maximum number of iterations (10^5) is reached. The first plot shows that FISTA could not produce a solution with reasonable accuracy even after 10^5 iterations for the unconditioned version. On the other hand, the same algorithm when run on the preconditioned version converged in 31044 iterations and produced a highly accurate solution with a duality gap of 10^{-8}. Second plot also shows that the preconditioned version produces a highly accurate solution with in 7661 iterations while the unconditioned version took more than six times as many iterations as that of the preconditioned version to get the same accuracy.

3.7 Conclusions

In this chapter we developed a common framework for incorporating soft and hard prior constraints arising in constrained clustering for the two-class setting. The main contribution is the derivation of exact continuous relaxation for the constrained balanced cut problem. The exact continuous relaxation allowed us to show that our method is guaranteed to find a solution satisfying all the constraints in the hard enforcement setting unlike the prior work. Moreover we also presented an efficient

method for solving the exact relaxation that has similar time complexity as that of the unconstrained version [56].

-

Chapter 4

Multi-class clustering

In this chapter we develop novel graph-based methods for unconstrained and constrained multi-class clustering problem. First we derive a novel continuous relaxation for a very generic multi-class balanced cut problem in Section 4.2. Our generic framework allows to minimize new application specific balancing criteria apart from the standard balance cut problems like ratio and normalized cut. Notice that in the multi-class setting the main difficultly arises from the k-way partitioning constraint and consequently many methods fail to even guarantee that their solution has k clusters. Unlike the existing work, our method is guaranteed to produce a k-way partitioning. In Section 4.3, we present our new monotonic descent algorithm for solving the resulting continuous relaxation which is a difficult sum-of-ratios minimization problem. Our thorough experiments against a wide variety of multi-class clustering methods on different balancing functions show that our method is the best solver for the balanced k-cut problem.

Our method provides a framework for incorporating prior knowledge in multi-class clustering problem. Note that none of the existing methods is able to handle even label constraints in the multi-class setting. In Section 4.4 we show how to incorporate label, must-link and cannot link constraints in the multi-class setting. Similar to two-class constrained clustering, we show that our method is able to produce a solution that satisfies all the given constraints (as long as any consistent partition can be found efficiently) apart from handling noisy or inconsistent constraints. Our monotonic descent algorithm developed for the relaxation of the unconstrained problem can readily be used to solve the constrained clustering problem. A part of the work presented in this chapter is published in [97].

We now introduce the underlying k-way graph cut problem. Let $G(V, W)$ be the similarity graph constructed from the given pairwise similarities where $V = \{1, \ldots, n\}$ is the n-element vertex set, $W \in \mathbb{R}_+^{n \times n}$ is the symmetric weight matrix. Here, we consider the following generalization of the balanced cut problem to k-class setting

$$\min_{(C_1,\ldots,C_k)\in P_k} \quad \sum_{i=1}^{k} \frac{\mathrm{cut}(C_i, \overline{C_i})}{\hat{S}(C_i)} \;=:\; \mathrm{BCut}(C_1, \ldots, C_k) \tag{4.1}$$

where P_k is the set of all k-partitions, $\hat{S} : 2^V \to \mathbb{R}_+$ is a balancing function that tries to enforce that all sets C_i are of the same "size". Here we assume that $\hat{S}(\emptyset) = 0$ and for any $C \subsetneq V$, $C \neq \emptyset$, $\hat{S}(C) \geq m$, for some $m > 0$. Several balancing functions have been proposed in the literature depending on the application. Most notable among

them are the following submodular balancing functions; we mention in brackets the name of the corresponding balanced graph cut criterion $\mathrm{BCut}(C_1, \ldots, C_k)$

$$\begin{aligned} &\hat{S}(C) = |C|, && \text{(Ratio Cut)}, \\ &\hat{S}(C) = \min\{|C|, |\overline{C}|\}, && \text{(Ratio Cheeger Cut)}, \\ &\hat{S}(C) = \min\{(k-1)|C|, \overline{C}\}, && \text{(Asymmetric Ratio Cheeger Cut)}. \end{aligned} \tag{4.2}$$

The well-studied *Ratio Cut* criteria [52, 115, 31] was proposed to produce clusters of same size. This has been generalized in the multi-class setting to *Asymmetric Ratio Cheeger Cut* [19] with the aim to bias the solution towards sets of size $\frac{|V|}{k}$. Note that $\hat{S}(C)$ attains its maximum at the sets of size $\frac{|V|}{k}$ which makes perfect sense if one expects k clusters of roughly equal size. An intermediate version between the two is the *Ratio Cheeger Cut* which has a symmetric balancing function and strongly penalizes overly large clusters. We can also handle the corresponding weighted cases e.g., $\hat{S}(C) = \mathrm{vol}(C) = \sum_{i \in C} d_i$, where $d_i = \sum_{j=1}^{n} w_{ij}$, leading to the *Normalized cut* variants

$$\begin{aligned} &\hat{S}(C) = \mathrm{vol}(C), && \text{(Normalized Cut)}, \\ &\hat{S}(C) = \min\{\mathrm{vol}(C), \mathrm{vol}(\overline{C})\}, && \text{(Normalized Cheeger Cut)}, \\ &\hat{S}(C) = \min\{(k-1)\,\mathrm{vol}(C), \mathrm{vol}(\overline{C})\}, && \text{(Asymmetric Normalized Cheeger Cut)}. \end{aligned}$$

4.1 State-of-the-art

Here we present the recent work [19] that attempted to directly solve the multi-class clustering problem. We refer to Section 2.2 for a review of other clustering methods. Recently the following *Asymmetric Ratio Cheeger Cut* problem for the multi-class setting is introduced in [19]

$$\min_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C}_l)}{\min\{(k-1)\,|C_l|\,, \left|\overline{C}_l\right|\}}. \tag{4.3}$$

Note that the balancing function $\min\{(k-1)\,|C_l|\,, \left|\overline{C}_l\right|\}$ here is motivated as follows. For the two-class problem, the Cheeger balancing function $\min\{|C|\,, \left|\overline{C}\right|\}$ encourages the two components C and $\left|\overline{C}\right|$ to have equal size. Similarly, for the k-class setting we expect each component C_l to have approximately the same size of $\frac{1}{k}\,|V|$. One can check that the newly introduced asymmetric balancing function has the maximum value for the sets of size $\frac{1}{k}\,|V|$.

The following continuous relaxation has been proposed for solving this problem in [19]

$$\begin{aligned} &\min_{\substack{F=(F_1,\ldots,F_k),\\ F\in\mathbb{R}_+^{n\times k}}} \sum_{l=1}^{k} \frac{\mathrm{TV}(F_l)}{S(F_l)} \\ &\text{subject to}: F_{(i)} \in \Delta_k, \quad i = 1, \ldots, n, \quad \text{(simplex constraints)} \end{aligned} \tag{4.4}$$

where Δ_k is the simplex defined as $\Delta_k = \{x \in \mathbb{R}^k \mid x_i \geq 0, \sum_{i=1}^{k} x_i = 1\}$ and $S(f) = \left\|f - \mathrm{quant}_{k-1}(f)\right\|_{1,(k-1)}$ is the Lovasz extension of $\hat{S}(C) = \min\{(k-1)|C|, \overline{C}\}$.

Here, $\text{quant}_\tau(f)$ is the $(j+1)^{st}$ largest value of f where $j = \lfloor \frac{n}{\tau+1} \rfloor$ and $\|f\|_{1,\tau}$ is defined as

$$\|f\|_{1,\tau} = \sum_{i=1}^{n} |f_i|_\tau \,, \qquad \text{where} \qquad |t|_\tau = \begin{cases} t\tau & t \geq 0 \\ -t & t < 0 \end{cases}$$

Note that this continuous formulation is a direct generalization of the exact relaxation proposed for the two-class problem [56]. We show in Section 4.2.1 that this relaxation (4.4) has a serious flaw. In fact this relaxation is void in the sense that it does not yield a clustering into k-components, where $k \geq 2$, for (i) any symmetric balancing function on any graph, (ii) asymmetric balancing function when the graph is disconnected or has a 2-way cut with very small value.

4.2 Continuous relaxation of the multi-class balanced cut

In this section we present a continuous relaxation for the balanced k-cut problem (4.1). We use the same idea as that of the two-class case [56, 22] where the exact relaxation results are obtained by replacing the set functions with their Lovász extensions. However, it turns out that a more difficult issue in deriving a tighter relaxation for the multi-cut problem is the choice of the constraints so that the continuous problem also yields a partition (together with a suitable rounding scheme). We first discuss how to enforce the k-partition constraint in the continuous setting. Let $F \in \mathbb{R}^{n \times k}$ and $F_l \in \mathbb{R}^n$, $l = 1, \ldots, k$ denote the l-th column of F and $F_{(i)} \in \mathbb{R}^k$ the i-th row of F. Here the rows of F correspond to the vertices of the graph and the j-th column of F corresponds to the set C_j of the k-partition. We employ the following constraints to enforce the k-partition constraint on the vertex set V,

$$\begin{aligned} &F_{(i)} \in \Delta_k, && i = 1, \ldots, n, && \text{(simplex constraints)} \\ &\max\{F_{(i)}\} = 1, && \forall i \in V, && \text{(membership constraints)} \\ &S(F_l) \geq m, && l = 1, \ldots, k, && \text{(size constraints)} \end{aligned}$$

where Δ_k denotes the simplex $\Delta_k = \{x \in \mathbb{R}^k \mid x_i \geq 0, \sum_{i=1}^k x_i = 1\}$, S is the Lovász extension of the balancing function $\hat{S}$ and $m > 0$ is the minimum value of $\hat{S}$ on non-empty sets. Here we used the notation $\max\{F_{(i)}\}$ to denote the maximum value of the vector $F_{(i)} \in \mathbb{R}^k$. The row-wise simplex and membership constraints enforce that each vertex belongs to exactly one component. Note that these constraints alone cannot still guarantee that F corresponds to a k-way partition since an entire column of F can be zero. This is avoided by the column-wise size constraints that enforce that each component has at least one vertex.

We propose the following continuous relaxation for the balanced k-cut problem (4.1)

$$\min_{\substack{F=(F_1,\ldots,F_k),\\ F \in \mathbb{R}_+^{n \times k}}} \sum_{l=1}^{k} \frac{\text{TV}(F_l)}{S(F_l)} \tag{4.5}$$

$$\begin{aligned} \text{subject to}: \; &F_{(i)} \in \Delta_k, && i = 1, \ldots, n, && \text{(simplex constraints)} \\ &\max\{F_{(i)}\} = 1, && \forall i \in I, && \text{(membership constraints)} \\ &S(F_l) \geq m, && l = 1, \ldots, k, && \text{(size constraints)} \end{aligned}$$

where S is the Lovasz extension of the set function $\hat{S}$ and $m = \min_{C \subsetneq V,\ C \neq \emptyset} \hat{S}(C)$. We have $m = 1$, for *Ratio Cut* and *Ratio Cheeger Cut* whereas $m = k - 1$ for *Asymmetric Ratio Cheeger Cut*. Note that the function TV is the Lovasz extension of the cut function. Here the index set $I \subseteq V$ controls the degree to which the partition constraint is enforced. This set is chosen adaptively by our method during the sequential minimization described in Section 4.3.

To obtain a clustering from a continuous solution F^*, we construct the sets by assigning each vertex i to the column where the i-th row attains its maximum. Formally,

$$C_l = \left\{ i \in V \,|\, l = \arg\max_{j=1,\ldots,k} \{F_{ij}\} \right\}, \quad l = 1, \ldots, k, \quad \text{(Rounding)} \tag{4.6}$$

where ties are broken randomly. If there exists a row such that the rounding is not unique, we say that the solution is weakly degenerated. If furthermore the resulting set $(C_1, \ldots, C_k)$ do not form a partition, that is one of the sets is empty, then we say that the solution is strongly degenerated.

We now state our first result showing the relation between our relaxation and the previous work of [56] for the special case $k = 2$. Indeed for symmetric balancing function such as the *Ratio Cheeger Cut*, our continuous relaxation (4.5) is exact even without membership and size constraints.

Theorem 4.1 *Let $\hat{S}$ be a non-negative symmetric balancing function i.e., $\hat{S}(C) = \hat{S}(\overline{C})$, and denote by p^* the optimal value of the problem* (4.5) *without membership and size constraints for $k = 2$. Then it holds*

$$p^* = \min_{(C_1, C_2) \in P_2} \sum_{l=1}^{2} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)}.$$

Furthermore there exists a solution F^* of (4.5) *such that $F^* = [\mathbf{1}_{C^*}, \mathbf{1}_{\overline{C^*}}]$, where $(C^*, \overline{C^*})$ is the optimal balanced 2-cut partition.*

Proof: Note that $\mathrm{cut}(C, \overline{C})$ and $\hat{S}$ (by assumption) are symmetric set functions. Thus with $C_2 = \overline{C_1}$,

$$\frac{\mathrm{cut}(C_1, \overline{C_1})}{\hat{S}(C_1)} + \frac{\mathrm{cut}(C_2, \overline{C_2})}{\hat{S}(C_2)} = 2 \frac{\mathrm{cut}(C_1, \overline{C_1})}{\hat{S}(C_1)}.$$

Moreover, since $\mathrm{TV}(V) = \hat{S}(V) = 0$ by symmetry, it holds that $\mathrm{TV}(\alpha f + \beta \mathbf{1}_n) = |\alpha|\ TV(f)$ and $S(\alpha f + \beta \mathbf{1}_n) = |\alpha|\ S(f)$ (see Proposition 2.1). The simplex constraint implies that $F_2 = \mathbf{1}_n - F_1$ and thus

$$\frac{\mathrm{TV}(F_2)}{S(F_2)} = \frac{\mathrm{TV}(\mathbf{1}_n - F_1)}{S(\mathbf{1}_n - F_1)} = \frac{\mathrm{TV}(F_1)}{S(F_1)}.$$

Thus we can write the problem (4.5) equivalently as

$$\min_{f \in [0,1]^V} 2 \frac{\mathrm{TV}(f)}{S(f)}. \tag{4.7}$$

As for all $A \subseteq V$, $\mathrm{TV}(\mathbf{1}_A) = \mathrm{cut}(A, \overline{A})$ and $S(\mathbf{1}_A) = \hat{S}(A)$, we have

$$\min_{f \in [0,1]^V} \frac{\mathrm{TV}(f)}{S(f)} \leq \min_{C \subseteq V} \frac{\mathrm{cut}(C, \overline{C})}{\hat{S}(C)}.$$

However, it has been shown in [56] that $\min_{f \in \mathbb{R}^n} \frac{\mathrm{TV}(f)}{S(f)} = \min_{C \subseteq V} \frac{\mathrm{cut}(C,\overline{C})}{\hat{S}(C)}$ and that there exists a continuous solution such that $f^* = \mathbf{1}_{C^*}$, where $C^* = \arg\min_{C \subseteq V} \frac{\mathrm{cut}(C,\overline{C})}{\hat{S}(C)}$. Thus f^* is a solution of (4.7) and consequently $F^* = [f^*, \mathbf{1}_n - f^*] = [\mathbf{1}_{C^*}, \mathbf{1}_{\overline{C^*}}]$ is optimal for (4.5). □

Note that rounding trivially yields a solution in the setting of the previous theorem.

Our second result shows that indeed our proposed optimization problem (4.5) is a relaxation of the balanced k-cut problem (4.1). Furthermore, the relaxation is exact if $I = V$.

Proposition 4.1 *The continuous problem* (4.5) *is a relaxation of the balanced k-cut problem* (4.1)*. The relaxation is exact, i.e., both problems are equivalent, if $I = V$.*

Proof: For any k-way partition $(C_1, \ldots, C_k)$, we can construct $F = (\mathbf{1}_{C_1}, \ldots, \mathbf{1}_{C_k})$. It obviously satisfies the membership and size constraints and the simplex constraint is satisfied as $\cup_i C_i = V$ and $C_i \cap C_j = \emptyset$ if $i \neq j$. Thus F is feasible for problem (4.5) and has the same objective value because

$$\mathrm{TV}(\mathbf{1}_C) = \mathrm{cut}(C, \overline{C}), \quad S(\mathbf{1}_C) = \hat{S}(C).$$

Thus problem (4.5) is a relaxation of (4.1).

If $I = V$, then the simplex together with the membership constraints imply that each row $F_{(i)}$ contains exactly one non-zero element which equals 1, i.e., $F \in \{0,1\}^{n \times k}$. Define for $l = 1, \ldots, k$, $C_l = \{i \in V \,|\, F_{il} = 1\}$ (i.e, $F_l = \mathbf{1}_{C_l}$), then it holds $\cup_l C_l = V$ and $C_l \cap C_j = \emptyset$, $l \neq j$. From the size constraints, we have for $l = 1, \ldots, k$, $0 < m \leq S(F_l) = S(\mathbf{1}_{C_l}) = \hat{S}(C_l)$. Thus $\hat{S}(C_l) > 0$, $l = 1, \ldots, k$, which by assumption on $\hat{S}$ implies that each C_l is non-empty. Hence the only feasible points allowed are indicators of k-way partitions and the equivalence of (4.1) and (4.5) follows. □

If $I = V$, it is immediate from the proof of Proposition 4.1 that the feasible set of the problem (4.5) contains only the indicator matrices of k-partitions. On the other hand, if $I = \emptyset$ (i.e., no membership constraints), and $k > 2$ it is not guaranteed that rounding of the solution of the continuous problem yields a partition. Indeed, for symmetric balancing functions one can, under these conditions, show that the solution is always strongly degenerated and rounding does not yield a partition (see Theorem 4.2 below). The idea behind our suggested relaxation is that minimizing the total variation yields piecewise constant solutions, which is well-known in image processing. In fact this follows from seeing the total variation as Lovasz extension of the cut function. Thus if $|I|$ is sufficiently large, the vertices where the values are fixed to 0 or 1 propagate this to their neighboring vertices and consequently to the whole graph. We discuss the choice of I in more detail in Section 4.3.

4.2.1 Why simplex constraints alone are not sufficient

We now show that for any symmetric balancing function in (4.1), the usage of simplex constraints alone in the optimization problem (4.4) is not sufficient to guarantee that the solution F^* can be rounded to a partition. For asymmetric balancing functions we can prove such a strong result only in the case where the graph is disconnected. However, note that if the number of components of the graph is less than the number of desired clusters k, the multi-cut problem is still non-trivial. Even in the case of asymmetric balancing functions, we show that the continuous relaxation (4.4), which uses only simplex constraints, would still fail if there exists a 2-way cut with very small value.

Theorem 4.2 *Let $\hat{S}(C)$ be any non-negative symmetric balancing function. Then the continuous relaxation*

$$\min_{\substack{F=(F_1,\ldots,F_k),\\ F\in\mathbb{R}_+^{n\times k}}} \sum_{l=1}^{k} \frac{\mathrm{TV}(F_l)}{S(F_l)} \tag{4.8}$$
$$\text{subject to : } F_{(i)} \in \Delta_k, \quad i = 1,\ldots,n, \quad \textit{(simplex constraints)}$$

of the balanced k-cut problem (4.1) *is void in the sense that the optimal solution F^* of the continuous problem can be constructed from the optimal solution of the 2-cut problem and F^* cannot be rounded to a k-way partition (see* (4.6) *for the definition of rounding). If the graph is disconnected, then the same holds also for any non-negative asymmetric balancing function.*

Proof: First, we derive a lower bound on the optimal value of the continuous relaxation (4.8). Then we construct a feasible point for (4.8) that achieves this lower bound but cannot yield a k-partition thus finishing the proof.

Let $(C^*, \overline{C^*}) = \arg\min_{C\subseteq V} \frac{\mathrm{cut}(C,\overline{C})}{\hat{S}(C)}$ be an optimal 2-way partition for the given graph. Using the exact relaxation result for the balanced 2-cut problem [56], we have

$$\min_{\substack{F:F_{(i)}\in\Delta_k\\ i=1\ldots,n}} \sum_{l=1}^{k} \frac{\mathrm{TV}(F_l)}{S(F_l)} \geq \sum_{l=1}^{k} \min_{f\in\mathbb{R}^n} \frac{\mathrm{TV}(f)}{S(f)} = \sum_{l=1}^{k} \min_{C\subseteq V} \frac{\mathrm{cut}(C,\overline{C})}{\hat{S}(C)} = k\,\frac{\mathrm{cut}(C^*,\overline{C^*})}{\hat{S}(C^*)}.$$

Now define $F_1 = \mathbf{1}_{C^*}$ and $F_l = \alpha_l \mathbf{1}_{\overline{C^*}}$, $l = 2,\ldots,k$ such that $\sum_{l=2}^{k} \alpha_l = 1, \alpha_l > 0$. Clearly $F = (F_1,\ldots,F_k)$ is feasible for the problem (4.8) and the corresponding objective value is

$$\frac{\mathrm{TV}(\mathbf{1}_{C^*})}{S(\mathbf{1}_{C^*})} + \sum_{l=2}^{k} \frac{\alpha_l \mathrm{TV}(\mathbf{1}_{\overline{C^*}})}{\alpha_l S(\mathbf{1}_{\overline{C^*}})} = \sum_{l=1}^{k} \frac{\mathrm{cut}(C^*,\overline{C^*})}{\hat{S}(C^*)},$$

where we used the (positive) 1-homogeneity of TV and S (Proposition 2.1) and the symmetry of cut and $\hat{S}$.

Thus the solution F constructed as above from the 2-cut problem is indeed optimal for the continuous relaxation (4.8) (as it achieves the lower bound derived above on the optimal value) and it is not possible to obtain a k-way partition from this solution as there will be $k-2$ sets that are empty. Finally, the argument can be

extended to asymmetric set functions if there exists a set C such that $\text{cut}(C, \overline{C}) = 0$ as in this case it does not matter that $\hat{S}(C) \neq \hat{S}(\overline{C})$ in order that the argument holds. □

The proof of Theorem 4.2 shows additionally that for any balancing function if the graph is disconnected, the solution of the continuous relaxation (4.8) is always zero, while clearly the solution of the balanced k-cut problem need not be zero. This shows that the relaxation can be arbitrarily bad in this case. In fact the relaxation for the asymmetric case can even fail if the graph is not disconnected but there exists a cut of the graph which is very small as the following corollary indicates.

Corollary 4.1 *Let $\hat{S}$ be an asymmetric balancing function and $C^* = \arg\min_{C \subseteq V} \frac{\text{cut}(C,\overline{C})}{\hat{S}(C)}$ and suppose that $\phi^* := (k-1)\frac{\text{cut}(C^*,\overline{C^*})}{\hat{S}(C^*)} + \frac{\text{cut}(C^*,\overline{C^*})}{\hat{S}(\overline{C^*})} < \min_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\text{cut}(C_l,\overline{C_l})}{\hat{S}(C_l)}$. Then there exists a feasible F with $F_1 = \mathbf{1}_{\overline{C^*}}$ and $F_l = \alpha_l \mathbf{1}_{C^*}$, $l = 2, \ldots, k$ such that $\sum_{l=2}^{k} \alpha_l = 1, \alpha_l > 0$ for (4.8) which has the objective value of $\sum_{i=1}^{k} \frac{\text{TV}(F_i)}{S(F_i)} = \phi^*$ and which cannot be rounded to a k-way partition.*

Proof: Let $F_1 = \mathbf{1}_{\overline{C^*}}$ and $F_l = \alpha_l \mathbf{1}_{C^*}$, $l = 2, \ldots, k$ such that $\sum_{l=2}^{k} \alpha_l = 1, \alpha_l > 0$. Clearly $F = (F_1, \ldots, F_k)$ is feasible for the problem (4.8) and the corresponding objective value is

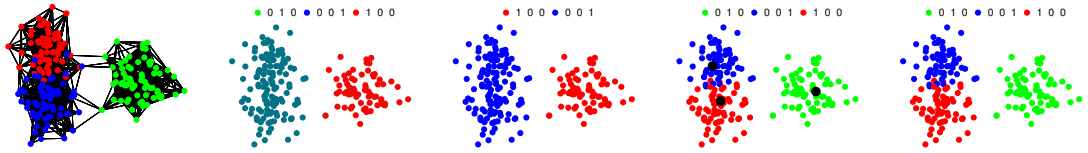
$$\sum_{l=1}^{k} \frac{\text{TV}(F_l)}{S(F_l)} = \frac{\text{TV}(\mathbf{1}_{\overline{C^*}})}{S(\mathbf{1}_{\overline{C^*}})} + \sum_{l=2}^{k} \frac{\alpha_l \text{TV}(\mathbf{1}_{C^*})}{\alpha_l S(\mathbf{1}_{C^*})} = \frac{\text{cut}(C^*, \overline{C^*})}{\hat{S}(\overline{C^*})} + (k-1)\frac{\text{cut}(C^*, \overline{C^*})}{\hat{S}(C^*)},$$

where we used the 1-homogeneity of TV and S (see Proposition 2.1) and the symmetry of cut. This F cannot be rounded to a k-way partition as there will be $k-2$ sets that are empty. □

Theorem 4.2 shows that the membership and size constraints which we have introduced in our relaxation (4.5) are essential to obtain a partition for symmetric balancing functions. For the asymmetric balancing function failure of the relaxation (4.8) and thus also of the relaxation (4.4) of [19] is only guaranteed for disconnected graphs. However, Corollary 4.1 indicates that degenerated solutions should also be a problem when the graph is still connected but there exists a dominating cut. We illustrate this with a toy example in Figure 4.1 where the algorithm of [19] for solving (4.4) fails as it converges exactly to the solution predicted by Corollary 4.1 and thus only produces a 2-partition instead of the desired 3-partition. The algorithm for our relaxation enforcing membership constraints ($|I| = 3$ in this example) converges to a continuous solution which is in fact a partition matrix so that no rounding is necessary.

4.3 Algorithm for the continuous relaxation

We now present an algorithm for solving the continuous relaxation (4.5). Unlike the method proposed by [19] for solving (4.4), our algorithm has a monotonic descent guarantee. The sequence $\{F^t\}$ produced by our method is feasible for the problem (4.5) and the corresponding objective values are monotonically decreasing. In the following, for ease of presentation, we assume that $\hat{S}$ is submodular. However, our



(a) (b) (c) (d) (e)

Figure 4.1: Toy example illustrating that the relaxation of [19] converges to a degenerate solution when applied to a graph with dominating 2-cut. (a) 10NN-graph generated from three Gaussians in 10 dimensions (b) continuous solution of (4.4) from [19] for $k = 3$, (c) rounding of the continuous solution of [19] does not yield a 3-partition (d) continuous solution found by our method together with the vertices $i \in I$ (black) where the membership constraint is enforced. Our continuous solution corresponds already to a partition. (e) clustering found by rounding of our continuous solution (trivial as we have converged to a partition). In (b)-(e), we color each data point i by treating $F_{(i)} \in \mathbb{R}^3$ as RGB triplet.

monotonic descent algorithm can easily be extended to handle any non-negative balancing function.

The key insight in order to derive a monotonic descent method for solving the sum-of-ratio minimization problem (4.5) is to eliminate the ratio by introducing a new set of variables $\beta = (\beta_1, \ldots, \beta_k)$:

$$\min_{\substack{F=(F_1,\ldots,F_k),\\ F\in\mathbb{R}_+^{n\times k},\ \beta\in\mathbb{R}_+^k}} \sum_{l=1}^{k} \beta_l \tag{4.9}$$

$$\begin{aligned}
\text{subject to}: \ & \mathrm{TV}(F_l) \leq \beta_l S(F_l), && l = 1, \ldots, k, && \text{(descent constraints)} \\
& F_{(i)} \in \Delta_k, && i = 1, \ldots, n, && \text{(simplex constraints)} \\
& \max\{F_{(i)}\} = 1, && \forall i \in I, && \text{(membership constraints)} \\
& S(F_l) \geq m, && l = 1, \ldots, k. && \text{(size constraints)}
\end{aligned}$$

Note that for an optimal solution (F^*, β^*) of this problem it holds $\mathrm{TV}(F_l^*) = \beta_l^* S(F_l^*)$, $l = 1, \ldots, k$ (otherwise one can decrease β_l^* and hence the objective) and thus equivalence holds. This is still a non-convex problem as the descent, membership and size constraints are non-convex. Our algorithm proceeds now in a sequential manner. At each iterate we do a convex inner approximation of the constraint set, that is the convex approximation is a subset of the non-convex constraint set, based on the current iterate (F^t, β^t). Then we optimize the resulting convex optimization problem and repeat the process. In this way we get a sequence of feasible points for the original problem (4.5) for which we will prove monotonic descent in the sum-of-ratios.

Convex approximation: As $\hat{S}$ is submodular, S is convex by Proposition 2.2. Let $(s_1^t, \ldots, s_k^t) \in \partial S(F^t)$ be an element of the sub-differential of S at the current iterate F^t. We have by Lemma 2.7, $(s_l^t)_{j_i} = \hat{S}(C_{l_{i-1}}) - \hat{S}(C_{l_i})$, where $C_{l_i} = \{j_{l_{i+1}}, \ldots, j_{l_n}\}$ and j_{l_i} is the index of the i^{th} smallest element of F_l^t. Moreover, using the definition of subgradient, we have $S(F_l) \geq S(F_l^t) + \langle s_l^t, F_l - F_l^t \rangle = \langle s_l^t, F_l \rangle$, where the last equality follows from Lemma 2.1.

Thus the inner approximation of size constraints is given by $\langle s_l^t, F_l \rangle \geq m,\ l = 1, \ldots, k$. For approximating the descent constraints, let $\lambda_l^t = \frac{\mathrm{TV}(F_l^t)}{S(F_l^t)}$ and introduce new variables $\delta_l = \beta_l - \lambda_l^t$ that capture the amount of change in each ratio. We further decompose δ_l as $\delta_l = \delta_l^+ - \delta_l^-,\ \delta_l^+ \geq 0,\ \delta_l^- \geq 0$. Let $M = \max_{f \in [0,1]^n} S(f) = \max_{C \subseteq V} \hat{S}(C)$, then for $S(F_l) \geq m$,

$$\begin{aligned}\mathrm{TV}(F_l) - \beta_l S(F_l) &\leq \mathrm{TV}(F_l) - \lambda_l^t \left\langle s_l^t, F_l \right\rangle - \delta_l^+ S(F_l) + \delta_l^- S(F_l) \\ &\leq \mathrm{TV}(F_l) - \lambda_l^t \left\langle s_l^t, F_l \right\rangle - \delta_l^+ m + \delta_l^- M.\end{aligned}$$

Finally, note that because of the simplex constraints, the membership constraints can be rewritten as $\max\{F_{(i)}\} \geq 1,\ i \in I$. The inner approximation is then given by $\left\langle v_i^t, F_{(i)} \right\rangle \geq 1$ where $v_i^t \in \mathbb{R}^k$ is any element of the subdifferential of $\max\{F_{(i)}\}$ at $F_{(i)}^t$. If $j_i = \arg\max_j \{F_{ij}^t\}$ (ties are broken randomly), then a vector in $\mathbb{R}^k$ which takes a value of 1 on the index j_i and zero everywhere else is a subgradient. Thus the membership constraints can be relaxed as $F_{ij_i} \geq 1$. As $F_{ij} \leq 1,\ i = 1, \ldots, n, j = 1, \ldots, k$, we get $F_{ij_i} = 1$. Note that because of the simplex constraints, $\arg\max_j \{F_{ij}^{t'}\}$ stays same in successive iterations $t' > t$ once we enforce the constraints $F_{ij_i} = 1, i \in I$. Thus the convex approximation of the membership constraints fixes the assignment of the i-th point to a cluster and thus can be interpreted as "label constraint". However, unlike the transductive setting, the labels for the vertices in I are automatically chosen by our method. The actual choice of the set I will be discussed in Section 4.3.2. We use the notation $L = \{(i, j_i) \mid i \in I\}$ for the label set generated from I (note that L is fixed once I is fixed).

Remark: Note that the approximation of membership constraints and size constraints is done using the ideas of DC programming ([35]; see Section 2.1.5). However, for descent constraints (which are not DC constraints in their current form) we have used a different approximation using the bounds of the balancing function S. A possible future work explores the ways of rewriting descent constraints as DC constraints that are amenable to efficient optimization.

Descent algorithm: Our descent algorithm for minimizing (4.9) solves at each iteration t the following convex optimization problem

$$\min_{\substack{F \in \mathbb{R}_+^{n \times k}, \\ \delta^+ \in \mathbb{R}_+^k,\ \delta^- \in \mathbb{R}_+^k}} \quad \sum_{l=1}^{k} \delta_l^+ - \delta_l^- \tag{4.10}$$

$$\begin{aligned}\text{subject to : } \mathrm{TV}(F_l) &\leq \lambda_l^t \left\langle s_l^t, F_l \right\rangle + \delta_l^+ m - \delta_l^- M, && l = 1, \ldots k, && \text{(descent con.)} \\ F_{(i)} &\in \Delta_k, && i = 1, \ldots, n, && \text{(simplex con.)} \\ F_{ij_i} &= 1, && \forall (i, j_i) \in L, && \text{(label con.)} \\ \left\langle s_l^t, F_l \right\rangle &\geq m, && l = 1, \ldots, k. && \text{(size con.)}\end{aligned}$$

As its solution F^{t+1} is feasible for (4.5) we update $\lambda_l^{t+1} = \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})}$ and $s_l^{t+1} \in \partial S(F_l^{t+1}),\ l = 1, \ldots, k$ and repeat the process until the sequence terminates, that is no further descent is possible or the relative descent in $\sum_{l=1}^k \lambda_l^t$ is smaller than a predefined ϵ. The following theorem shows the monotonic descent property of our algorithm.

Theorem 4.3 *The sequence* $\{F^t\}$ *produced by the above algorithm satisfies*

$$\sum_{l=1}^{k} \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} < \sum_{l=1}^{k} \frac{\mathrm{TV}(F_l^t)}{S(F_l^t)}$$

for all $t \geq 0$ *or the algorithm terminates.*

Proof: Let $(F^{t+1}, \delta^{+,\ t+1}, \delta^{-,\ t+1})$ be the optimal solution of the inner problem (4.10). By the feasibility of $(F^{t+1}, \delta^{+,\ t+1}, \delta^{-,\ t+1})$ and $m \leq S(F_l^{t+1}) \leq M$,

$$\begin{aligned}\frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} &\leq \frac{\lambda_l^t \left\langle s_l^t, F_l^{t+1} \right\rangle + m\delta_l^{+,\ t+1} - M\delta_l^{-,\ t+1}}{S(F_l^{t+1})} \\ &\leq \lambda_l^t + \frac{m\delta_l^{+,\ t+1} - M\delta_l^{-,\ t+1}}{S(F_l^{t+1})} \leq \lambda_l^t + \delta_l^{+,\ t+1} - \delta_l^{-,\ t+1}\end{aligned}$$

Summing over all ratios, we have

$$\sum_{l=1}^{k} \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} \leq \sum_{l=1}^{k} \lambda_l^t + \sum_{l=1}^{k} \delta_l^{+,\ t+1} - \delta_l^{-,\ t+1}$$

Noting that $\delta_l^+ = \delta_l^- = 0,\ F = F^t$ is feasible for (4.10), the optimal value $\sum_{l=1}^{k} \delta_l^{+,\ t+1} - \delta_l^{-,\ t+1}$ has to be either strictly negative in which case we have strict descent

$$\sum_{l=1}^{k} \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} < \sum_{l=1}^{k} \lambda_l^t$$

or the previous iterate F^t together with $\delta_l^+ = \delta_l^- = 0$ is already optimal and hence the algorithm terminates. □

From the proof it is clear that one does not need to solve the inner problem to full accuracy to guarantee monotonic descent. One needs to find only an iterate with a negative objective value for the inner problem to maintain the monotonic descent property. In our experiments we found that solving the inner problems to low accuracy initially and then doing more iterations for the latter problems typically gives better results.

4.3.1 Smooth minimization of the inner problem

The inner problem (4.10) is convex, but contains the non-smooth term TV in the constraints. We eliminate the non-smoothness by introducing additional variables and derive an equivalent linear programming (LP) formulation.

Lemma 4.1 *Let* $E \subseteq V \times V$ *be the set of edges of the graph* $G(V, W)$ *and* $\overrightarrow{E} = \{(i,j) \in E,\ i < j\}$ *denote the directed edges. Further let* $w \in \mathbb{R}^{|\overrightarrow{E}|}$ *be the edge weights. Then the convex inner problem* (4.10) *is equivalent to the linear optimiza-*

tion problem

$$\min_{\substack{F\in\mathbb{R}_+^{n\times k},\\ \alpha\in\mathbb{R}_+^{|\overrightarrow{E}|\times k},\\ \delta^+\in\mathbb{R}_+^k,\ \delta^-\in\mathbb{R}_+^k}} \quad \sum_{l=1}^{k} \delta_l^+ - \delta_l^- \tag{4.11}$$

$$\begin{aligned}
\text{subject to}: \quad & \langle w, \alpha_l\rangle \le \lambda_l^t \left\langle s_l^t, F_l\right\rangle + \delta_l^+ m - \delta_l^- M, && l=1,\dots,k, && \textit{(descent con.)}\\
& F_{(i)} \in \Delta_k, && i=1,\dots,n, && \textit{(simplex con.)}\\
& F_{ij_i} = 1, && \forall (i,j_i)\in L, && \textit{(label con.)}\\
& \left\langle s_l^t, F_l^t\right\rangle \ge m, && l=1,\dots,k, && \textit{(size con.)}\\
& -(\alpha_l)_{ij} \le F_{il} - F_{jl} \le (\alpha_l)_{ij}, && l=1,\dots,k, && \forall (i,j)\in\overrightarrow{E}.
\end{aligned}$$

Proof: We define new variables $\alpha_l \in \mathbb{R}^{|\overrightarrow{E}|},\ l=1,\dots,k$ and introduce constraints $(\alpha_l)_{ij} = |(F_l)_i - (F_l)_j)|$. This allows us to rewrite $\mathrm{TV}(F_l)$ as $\langle w, \alpha_l\rangle,\ l=1,\dots,k$. These equality constraints can be replaced by the inequality constraints $(\alpha_l)_{ij} \ge |(F_l)_i - (F_l)_j)|$ without changing the optimality of the problem, because at any optimal solution these constraints are active. This can be seen as follows. Let $(F,\alpha,\delta^+,\delta^-)$ be an optimal solution of the modified problem where we introduced α and replaced equality constraints on α by inequality constraints. Assume for the sake of contradiction that there exists $l\in\{1,\dots,k\}$ and $(r,s)\in\overrightarrow{E}$ such that $(\alpha_l)_{rs} = |(F_l)_r - (F_l)_s| + \epsilon$, for some $\epsilon > 0$. Define $(\hat{\alpha}_l)_{ij} = (\alpha_l)_{ij}, \forall (i,j)\in \overrightarrow{E}\backslash\{(r,s)\}$, $(\hat{\alpha}_l)_{rs} = (\alpha_l)_{rs} - \epsilon$ and $\hat{\delta}_l^- = \delta_l^- + \frac{1}{M}\epsilon w_{rs}$. Note that both $(\hat{\alpha}_l)_{rs}$ and $\hat{\delta}_l^-$ are non-negative (since w_{rs} is non-negative) and by definition $(\hat{\alpha}_l)_{rs} \ge |(F_l)_r - (F_l)_s|$. Moreover, we have

$$\begin{aligned}
\langle w, \hat{\alpha}_l\rangle &= \langle w, \alpha_l\rangle - \epsilon w_{rs} \le \lambda_l^t \left\langle s_l^t, F_l\right\rangle + \delta_l^+ m - \delta_l^- M - \epsilon w_{rs}\\
&\le \lambda_l^t \left\langle s_l^t, F_l\right\rangle + \delta_l^+ m - M(\delta_l^- + \frac{1}{M}\epsilon w_{rs})
\end{aligned}$$

Thus $(F,\hat{\alpha},\delta^+,\hat{\delta}^-)$ satisfies all the constraints of the modified problem and has a smaller objective since $w_{rs} > 0, (r,s)\in\overrightarrow{E}$, which leads to the required contradiction. Finally, the inequality constraints on α are rewritten using the fact that $|x| \le y \Leftrightarrow -y \le x \le y$, for $y \ge 0$. □

Recently, first-order primal-dual hybrid gradient descent (PDHG for short) methods have been proposed [43, 24] to efficiently solve a class of convex optimization problems that can be rewritten as the following saddle-point problem

$$\min_{x\in X}\max_{y\in Y} \langle Ax, y\rangle + G(x) - \Phi^*(y), \tag{4.12}$$

where X and Y are finite-dimensional vector spaces and $A: X\to Y$ is a linear operator and G and Φ^* are convex functions. Here Φ^* denotes the convex conjugate of Φ defined as

$$\Phi^*(y) = \sup_{x\in X} \langle x, y\rangle - \Phi(x).$$

The main step of PDHG given in Algorithm 4.1 requires the computation of so-called proximal map defined as

$$\text{prox}_{\tau G}(\cdot) := \arg\min_{x \in X} G(x) + \frac{1}{2\tau} \|x - \cdot\|_2^2 .$$

Algorithm 4.1 PDHG for solving the saddle-point problem (4.12)

1: **Initialization:** $x^{(0)} = \bar{x}^{(0)} = 0$, $y^0 = 0$, $\theta \in [0, 1]$, $\sigma, \tau > 0$ with $\sigma\tau < 1/\|A\|_2^2$
2: **repeat**
3: $\quad x^{r+1} = \text{prox}_{\tau G}(x^r - \tau A^T y^r)$
4: $\quad \bar{x}^{r+1} = x^{r+1} + \theta(x^{r+1} - x^r)$
5: $\quad y^{r+1} = \text{prox}_{\sigma \Phi^*}(y^r + \sigma A \bar{x}^{r+1})$
6: **until** relative duality gap $< \epsilon$
7: **Output:** x^{r+1}.

It has been shown that the PDHG algorithm achieves good performance in solving large scale linear programming problems that appear in computer vision applications [24, 94]. We now show how the linear programming problem

$$\begin{aligned} \min_{x \geq 0} \ & \langle c, x \rangle \\ \text{subject to} : \ & A_1 x \leq b_1 \\ & A_2 x = b_2 \end{aligned}$$

can be rewritten as a saddle-point problem so that PDHG can be applied.

By introducing the Lagrange multipliers y, the optimal value of the LP can be written as

$$\begin{aligned} & \min_{x \geq 0} \langle c, x \rangle + \max_{y_1 \geq 0, \ y_2} \langle y_1, A_1 x - b_1 \rangle + \langle y_2, A_2 x - b_2 \rangle \\ = & \min_{x} \max_{y_1, \ y_2} \langle c, x \rangle + \iota_{x \geq 0}(x) + \langle y_1, A_1 x \rangle + \langle y_2, A_2 x \rangle - \langle b_1, y_1 \rangle - \langle b_2, y_2 \rangle - \iota_{y_1 \geq 0}(y_1), \end{aligned}$$

where $\iota_{\cdot \geq 0}$ is the indicator function that takes a value of 0 on the non-negative orthant and ∞ elsewhere.

Define $b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$, $A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix}$ and $y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$. Then the saddle point problem corresponding to the LP is given by

$$\min_{x} \max_{y_1, \ y_2} \langle c, x \rangle + \iota_{x \geq 0}(x) + \langle y, Ax \rangle - \langle b, y \rangle - \iota_{y_1 \geq 0}(y_1).$$

Comparing this with the general form given in (4.12), we have

$$\begin{aligned} G(x) &= \langle c, x \rangle + \iota_{x \geq 0}(x) \\ \Phi^*(y) &= \langle b, y \rangle + \iota_{y_1 \geq 0}(y_1) \end{aligned}$$

The primal iterate can then be obtained as

$$\begin{aligned} x^{r+1} &= \text{prox}_{\tau G}(x^r - \tau A^T y^r) \\ &= \arg\min_{x \geq 0} \langle c, x \rangle + \frac{1}{2\tau} \left\| x - \left(x^r - \tau A^T y^r \right) \right\|_2^2 \\ &= \arg\min_{x \geq 0} \frac{1}{2\tau} \left\| x - \left(x^r - \tau (A^T y^r + c) \right) \right\|_2^2 \\ &= \max\{0, x^r - \tau (A^T y^r + c)\}. \end{aligned}$$

Similarly, the dual update can be derived as

$$\begin{aligned} y^{r+1} &= \text{prox}_{\sigma\Phi^*}(y^r + \sigma A\bar{x}^{r+1}) \\ &= \underset{y_1\geq 0,\ y_2}{\arg\min} \langle b, y\rangle + \frac{1}{2\sigma}\left\|y - \left(y^r + \sigma A\bar{x}^{r+1}\right)\right\|_2^2 \end{aligned}$$

where $\bar{x}^{r+1} = 2x^{r+1} - x^r$ (with the choice $\theta = 1$). We can minimize over y_1 and y_2 independently and obtain the iterates as

$$\begin{aligned} y_1^{r+1} &= \underset{y_1\geq 0}{\arg\min} \langle b_1, y_1\rangle + \frac{1}{2\sigma}\left\|y_1 - \left(y_1^r + \sigma A_1\bar{x}^{r+1}\right)\right\|_2^2 \\ &= \underset{y_1\geq 0}{\arg\min} \frac{1}{2\sigma}\left\|y_1 - \left(y_1^r + \sigma(A_1\bar{x}^{r+1} - b_1)\right)\right\|_2^2 \\ &= \max\{0, y_1^r + \sigma(A_1\bar{x}^{r+1} - b_1)\}, \\ y_2^{r+1} &= \underset{y_2}{\arg\min} \langle b_2, y_2\rangle + \frac{1}{2\sigma}\left\|y_2 - \left(y_2^r + \sigma A_2\bar{x}^{r+1}\right)\right\|_2^2 \\ &= \underset{y_2}{\arg\min} \frac{1}{2\sigma}\left\|y_2 - \left(y_2^r + \sigma(A_2\bar{x}^{r+1} - b_2)\right)\right\|_2^2 \\ &= y_2^r + \sigma(A_2\bar{x}^{r+1} - b_2). \end{aligned}$$

Here the primal and dual step sizes τ and σ are chosen such that $\tau\sigma\left\|A\right\|^2 < 1$, where $\|.\|$ denotes the operator norm.

It is observed often in practice that if the linear operator A has non-uniform values, the convergence of PDHG significantly slows down [94]. Hence one needs to apply preconditioning techniques similar to what we discussed in Section 3.5.4. Note that unlike FISTA, here one needs preconditioners for both primal as well as the dual iterates. That is one needs to address the issue of non-uniform scaling for both A and A^T. To understand the preconditioning, let us rewrite the primal and dual iterates of the general PDHG algorithm as

$$\begin{aligned} x^{r+1} &= \underset{x\in X}{\arg\min} \langle Ax, y^r\rangle + G(x) + \frac{1}{2\tau}\left\|x - x^r\right\|_2^2 \\ y^{r+1} &= \underset{y\in Y}{\arg\min} -\left\langle \bar{x}^{r+1}, A^T y\right\rangle + \Phi^*(y) + \frac{1}{2\sigma}\left\|y - y^r\right\|_2^2 \end{aligned}$$

The following preconditioned iterates are proposed in [94] for the PDHG algorithm

$$\begin{aligned} x^{r+1} &= \underset{x\in X}{\arg\min} \langle Ax, y^r\rangle + G(x) + \frac{1}{2}\boldsymbol{\tau}^{-1}\left\|x - x^r\right\|_2^2 \\ y^{r+1} &= \underset{y\in Y}{\arg\min} -\left\langle \bar{x}^{r+1}, A^T y\right\rangle + \Phi^*(y) + \frac{1}{2}\boldsymbol{\sigma}^{-1}\left\|y - y^r\right\|_2^2. \end{aligned}$$

Here $\boldsymbol{\tau}$ and $\boldsymbol{\sigma}$ are diagonal preconditioning matrices whose diagonal elements are given by

$$\boldsymbol{\tau}_j = \frac{1}{\sum_{i=1}^{n_r}|A_{ij}|}, \forall j \in \{1, \ldots, n_c\},\ \boldsymbol{\sigma}_i = \frac{1}{\sum_{i=1}^{n_c}|A_{ij}|}, \forall i \in \{1, \ldots, n_r\},$$

where $n_r,\ n_c$ are the number of rows and the number of columns of the matrix A. The idea behind these preconditioners is to compensate for the non-uniform

scaling of the primal variable x by A (resp. dual variable y by A^T) by rescaling the corresponding step sizes by the total magnitude of each column (resp. each row) of A. We note here that this approach is slightly indirect to what we have presented in Section 3.5.4 with the change of variable interpretation. However, if one applies the above rescaling for changing the variable α in problem (3.20) discussed in Section 3.5.4, one arrives at updates that are similar to what we have derived in [96] independently of the work of [94]. This can be seen by noting that the corresponding rescaling in step size for α_{ij} in problem (3.20) (with the linear operator A defined in (3.15)) is given by

$$\frac{1}{|w_{ij}| + |-w_{ij}|} = \frac{1}{2w_{ij}}, \forall (i,j) \in \overrightarrow{E}$$

This corresponds to the variable substitution

$$\alpha_{ij} = \frac{1}{2w_{ij}} \beta_{ij}.$$

This is exactly same (up to factor 2) as the one we used in our preconditioning (see Lemma 3.6).

For completeness, we now present the explicit form of the primal and dual iterates of the preconditioned PDHG for the LP (4.11). Let $\theta \in \mathbb{R}^k,\ \mu \in \mathbb{R}^n,\ \zeta \in \mathbb{R}^{|L|},\ \nu \in \mathbb{R}^k,\ \eta_l \in \mathbb{R}^{|\vec{E}|},\ \xi_l \in \mathbb{R}^{|\vec{E}|},\ \forall l \in \{1, \ldots, k\}$ be the Lagrange multipliers corresponding to the descent, simplex, label, size and the two sets of additional constraints (introduced to eliminate the non-smoothness) respectively. Let $B : \mathbb{R}^{|\vec{E}|} \to \mathbb{R}^{|V|}$ be a linear mapping defined as $(Bz)_i = \sum_{j:(i,j)\in\vec{E}} z_{ij} - z_{ji}$ and $\mathbf{1}_n \in \mathbb{R}^n$ denote a vector of all ones. Then the primal iterates for the LP (4.11) are given by

$$\begin{aligned}
F_l^{r+1} &= \max\Big\{0, F_l^r - \boldsymbol{\tau}_{F,\,l}\Big((-\theta_l^r \lambda_l^t - \nu_l^r) s_l^t + \mu^r + Z_l^r + B(\eta_l^r - \xi_l^r)\Big)\Big\}, \\
&\quad \forall l \in \{1, \ldots, k\}, \\
\alpha_l^{r+1} &= \max\Big\{0, \alpha_l^r - \boldsymbol{\tau}_{\alpha,\,l}\Big(\theta_l^r w - \eta_l^r - \xi_l^r\Big)\Big\},\ \forall l \in \{1, \ldots, k\}, \\
\delta^{+,\,r+1} &= \max\Big\{0, \delta^{+,\,r} - \boldsymbol{\tau}_{\delta^+}\Big(-m\theta^r + \mathbf{1}_k\Big)\Big\}, \\
\delta^{-,\,r+1} &= \max\Big\{0, \delta^{-,\,r} - \boldsymbol{\tau}_{\delta^-}\Big(M\theta^r - \mathbf{1}_k\Big)\Big\},
\end{aligned}$$

where $Z_l^r \in \mathbb{R}^n, l = 1, \ldots, k$, are given by $(Z_l^r)_i = \zeta_{il}^r$, if $(i,l) \in L$ and 0 otherwise. Here $\boldsymbol{\tau}_{F,\,l},\ \boldsymbol{\tau}_{\alpha,\,l},\ \boldsymbol{\tau}_{\delta^+},\ \boldsymbol{\tau}_{\delta^-}$ are the diagonal preconditioning matrices whose diagonal elements are given by

$$\begin{aligned}
(\boldsymbol{\tau}_{F,\,l})_i &= \frac{1}{(1+\lambda_l^t)\,|(s_l^t)_i| + 2d_i + \rho_{il} + 1},\ \forall i \in \{1, \ldots, n\}, \\
(\boldsymbol{\tau}_{\alpha,\,l})_{ij} &= \frac{1}{w_{ij} + 2},\ \forall (i,j) \in E, \\
(\boldsymbol{\tau}_{\delta^+})_l &= \frac{1}{m},\ \forall l \in \{1, \ldots, k\}, \\
(\boldsymbol{\tau}_{\delta^-})_l &= \frac{1}{M},\ \forall l \in \{1, \ldots, k\},
\end{aligned}$$

where d_i is the number of vertices adjacent to the i^{th} vertex and $\rho_{il} = 1$, if $(i,l) \in L$ and 0 otherwise.

The dual iterates are given by

$$\begin{aligned}
\theta_l^{r+1} &= \max\Big\{0, \theta_l^r + \boldsymbol{\sigma}_{\theta,\,l}\Big(\big\langle w, \bar{\alpha}_l^{r+1}\big\rangle - \lambda_l^t \big\langle s_l^t, \bar{F}_l^{r+1}\big\rangle - m\bar{\delta}_l^{+,\,r+1} + M\bar{\delta}_l^{-,\,r+1}\Big)\Big\}, \\
&\qquad l = 1, \ldots, k, \\
\mu^{r+1} &= \mu^r + \boldsymbol{\sigma}_\mu\Big(\bar{F}^{r+1}\mathbf{1}_k - \mathbf{1}_n\Big), \\
\zeta_{il}^{r+1} &= \zeta_{il}^r + \boldsymbol{\sigma}_\zeta\Big(\bar{F}_{il}^{r+1} - 1\Big),\ \forall (i,l) \in L, \\
\nu_l^{r+1} &= \max\Big\{0, \nu_l^r + \boldsymbol{\sigma}_{\nu,\,l}\Big(-\big\langle s_l^t, \bar{F}_l^{r+1}\big\rangle + m\Big)\Big\},\ \forall l \in \{1, \ldots, k\}, \\
\eta_l^{r+1} &= \max\Big\{0, \eta_l^r + \boldsymbol{\sigma}_{\eta,\,l}\Big(-\bar{\alpha}_l^{r+1} + \bar{F}_{il}^{r+1} - \bar{F}_{jl}^{r+1}\Big)\Big\},\ \forall l \in \{1, \ldots, k\}, \\
\xi_l^{r+1} &= \max\Big\{0, \xi_l^r + \boldsymbol{\sigma}_{\xi,\,l}\Big(-\bar{\alpha}_l^{r+1} - \bar{F}_{il}^{r+1} + \bar{F}_{jl}^{r+1}\Big)\Big\},\ \forall l \in \{1, \ldots, k\},
\end{aligned}$$

where

$$\boldsymbol{\sigma}_{\theta,\,l} = \frac{1}{\langle w, 1\rangle + \lambda_l^t \sum_{i=1}^n |(s_l^t)_i| + m + M},\ \boldsymbol{\sigma}_\zeta = 1,\ \boldsymbol{\sigma}_{\nu,\,l} = \frac{1}{\sum_{i=1}^n |(s_l^t)_i|},$$

and $\boldsymbol{\sigma}_\mu,\ \boldsymbol{\sigma}_{\eta,1},\ \boldsymbol{\sigma}_{\xi,l}$ are the diagonal preconditioning matrices whose diagonal elements are given by

$$(\boldsymbol{\sigma}_\mu)_i = \frac{1}{k},\ \forall i \in \{1, \ldots, n\},\ (\boldsymbol{\sigma}_{\eta,l})_{ij} = (\boldsymbol{\sigma}_{\xi,l})_{ij} = \frac{1}{3},\ \forall (i,j) \in E.$$

From the iterates, one sees that the computational cost per iteration is $O(\big|\overrightarrow{E}\big|)$. In our implementation, we further reformulated the LP (4.11) by directly integrating the label constraints, thereby reducing the problem size and getting rid of the dual variable ζ.

4.3.2 Choice of membership constraints I

In the previous section we have presented an algorithm for solving the continuous relaxation (4.9) for a given choice of membership constraints I. Here we discuss how we choose this set I. Since the membership constraints are relaxed as labels in the descent algorithm presented in the previous section, it is best to choose I from the currently known best k-partition. Let $(C_1, \ldots, C_k)$ be the current best k-partition obtained by rounding the continuous solution F. For each $l \in \{1, \ldots, k\}$ and $i \in C_l$ we compute

$$b_{li} = \frac{\mathrm{cut}\big(C_l \backslash \{i\},\, \overline{C_l} \cup \{i\}\big)}{\hat{S}(C_l \backslash \{i\})} + \min_{s \neq l}\left[\frac{\mathrm{cut}\big(C_s \cup \{i\},\, \overline{C_s} \backslash \{i\}\big)}{\hat{S}(C_s \cup \{i\})} + \sum_{j \neq l,\ j \neq s} \frac{\mathrm{cut}(C_j, \overline{C_j})}{\hat{S}(C_j)}\right] \tag{4.13}$$

and define $\mathcal{O}_l = \{(\pi_1, \ldots, \pi_{|C_l|}) \,|\, b_{l\pi_1} \geq b_{l\pi_2} \geq \ldots \geq b_{l\pi_{|C_l|}}\}$. The top-ranked vertices in $\mathcal{O}_l$ correspond to the ones which lead to the largest minimal increase in BCut when moved from C_l to another component and thus are most likely to belong to

their current component. So, whenever $|I| < |V|$, it is natural to fix the top-ranked vertices for each component first. In this way, the membership constraints always correspond to the vertices which lead to largest minimal increase in BCut when moved to another component. In Figure 4.1 (d) one can observe that the points where membership constraints are enforced lie close to the centers of the found clusters.

The next question is the number of the membership constraints one needs to enforce. If $|I|$ is too small, then solving the continuous relaxation (4.9) may not improve the balanced k-cut because, e..g, the solution of the problem (4.9) is weakly or strongly degenerated. On the other hand if $|I|$ is too large, the labels are fixed for many vertices leading to a poor solution (in terms of the balanced k-cut). We follow an iterative approach where we start with $I = \emptyset$ and increase its size whenever the solution of the continuous relaxation (4.9) is strongly degenerated or does not yield a k-partition with a strictly better balanced k-cut. We use the following strategy to update the size of I: if the current I is empty, then we add one membership constraint per component; otherwise we double the number of membership constraints per component. To make things simple, we add the same number of membership constraints in every component. Thus, we stop the method if all vertices in the smallest component of the currently known best k-partition were already included in the set I.

The overall scheme for solving the balanced k-cut problem (4.1) is given in Algorithm 4.2 while Algorithm 4.3 presents the descent method for solving the continuous relaxation (4.9) in each iteration of Algorithm 4.2. Note that Algorithm 4.3 requires a feasible starting point in order to guarantee monotonic descent in the objective of (4.9); hence we update F^r so that it satisfies the new membership constraints introduced in each iteration of Algorithm 4.2. We show in the following that F^r updated in this way still satisfies the size constraints. In general, one does not need to solve (4.9) to full accuracy since the membership constraints are going to change in the next iteration. In practice, best results are obtained by exiting Algorithm 4.3 as soon as a better k-partition $(C_1^{t+1}, \ldots, C_k^{t+1})$ is found which yields a different set of membership constraints from that of I. This makes sense because in this way the membership constraints (and hence labels) are always derived from the best k-partition found so far. In this case, since F^{t+1} yielded a better k-partition, we keep the size of membership constraints fixed for the next iteration $r + 1$ in Algorithm 4.2. We now show that the initialization received by Algorithm 4.3 is always feasible for the continuous relaxation (4.9).

Lemma 4.2 *The initialization received by Algorithm 4.3 in every iteration r of Algorithm 4.2 is feasible for the continuous relaxation* (4.9) *with $I = I^r$.*

Proof: If $I^r = \emptyset$, then F^r is either the initialization received by Algorithm 4.2 or the solution of the continuous relaxation (4.9) from the previous iteration. In both cases F^r is feasible for the continuous relaxation (4.9) with $I = \emptyset$. If $I^r \neq \emptyset$, then F^r updated in line 7 of Algorithm 4.2 satisfies simplex constraints and membership constraints. The descent constraints can always be satisfied by choosing sufficiently large positive $\beta_l, l = 1, \ldots, k$. We now show that F^r also satisfies size constraints. For ease of notation, let $f = F_l^r$ for any $l \in \{1, \ldots, k\}$. Note that $f \in [0, 1]^n$ and $\min_i\{f_i\} = 0, \max_i\{f_i\} = 1$ (since labels are enforced in every component). Hence,

if $f \in \{0,1\}^n$, then $A = \{i \,|f_i = 1\}$ is neither empty nor the full set V. In this case, $S(f) = \hat{S}(A) \geq m = \min_{C \subsetneq V,\, C \neq \emptyset} \hat{S}(C)$. On the other hand, if $f \notin \{0,1\}^n$, then let f be ordered in increasing order $0 = f_1 \leq f_2 \ldots \leq f_n = 1$. Further let $u > 1$ and $v < n$ denote the smallest and largest indices (after reordering) such that $f_u \in (0,1)$ and $f_v \in (0,1)$. Using the definition of the Lovász extension (2.5), we have

$$\begin{aligned} S(f) &= \hat{S}(C_{u-1}) f_u + \sum_{i=u}^{v-1} \hat{S}(C_i)(f_{i+1} - f_i) + \hat{S}(C_v)(1 - f_v) \\ &\geq m f_u + m(f_v - f_u) + m(1 - f_v) = m, \end{aligned}$$

where we used the fact that each of the thresholded sets $C_i, i = u-1, \ldots, v$ is neither empty nor the full set V and hence satisfies $\hat{S}(C_i) \geq m$. □

Algorithm 4.2 Minimization of the balanced k-cut problem (4.1)

Input: $F^0 \in \mathbb{R}_+^{n \times k}$ be such that $F^0 \mathbf{1}_k = \mathbf{1}_n$, $S(F_l) \geq m, l = 1 \ldots, k$ and rounding F^0 according to (4.6) yields a k-partition
Output: a k-partition $(C_1, \ldots, C_k)$
$\phi^0 = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^0)}{S(F_l^0)}$, $I^0 = \emptyset$, $p = 0$
$\chi^0 = \mathrm{BCut}(C_1^0, \ldots, C_k^0)$, where $(C_1^0, \ldots, C_k^0)$ be the partition obtained from F^0 via rounding (4.6)
repeat
  if $I^r \neq \emptyset$ **then**
    $F_{ij}^r = 0,\ \forall i \in I^r,\ \forall j \in \{1, \ldots, k\},\ F_{ij_i}^r = 1,\ \forall i \in I^r$ where j_i is the component index such that $i \in C_{j_i}^r$.
    $\phi^r = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^r)}{S(F_l^r)}$
  end if
  Let $(F^{r+1}, (C_1^{r+1}, \ldots, C_k^{r+1}), \texttt{degenerated})$ be the output obtained from Algorithm 4.3 with initialization F^r, $(C_1^r, \ldots, C_k^r)$ and the membership constraints I^r
  $\phi^{r+1} = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^{r+1})}{S(F_l^{r+1})},\ \chi^{r+1} = \mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1})$
  compute ordering $\mathcal{O}_l, \forall l = 1, \ldots, k$ for $(C_1^{r+1}, \ldots, C_k^{r+1})$ according to (4.13)
  if `degenerated` **then**
    $p_{\max} = \min_l \{\left|C_l^{r+1}\right|\}$
    if $p = p_{\max}$ **then**
      break (labels of all vertices in a component were enforced)
    else
      $p = \max\{\min\{p_{\max}, 2p\}, 1\}$ (increase the number of membership constraints)
    end if
  end if
  $I^{r+1} = \bigcup_{l=1}^k \mathcal{O}_l^p$, where $\mathcal{O}_l^p$ denotes p top-ranked vertices in $\mathcal{O}_l$
until $\chi^{r+1} = \phi^{r+1}$ and $\phi^{r+1} = \phi^r$
return $((C_1^{r+1}, \ldots, C_k^{r+1}), F^{r+1})$

We have the following guarantee for the overall algorithm.

Algorithm 4.3 Minimization of the continuous relaxation (4.9)

Input: I, $F^0 \in \mathbb{R}_+^{n\times k}$ feasible for the problem (4.9) and a k-partition $(C_1^0, \ldots, C_k^0)$

Output: F which is feasible for the problem (4.9) and achieves monotonic descent in the objective of (4.9), a k-partition $(C_1, \ldots, C_k)$ and `degenerated` flag

$\chi^0 = \mathrm{BCut}(C_1^0, \ldots, C_k^0)$, $\lambda_l^0 = \frac{\mathrm{TV}(F_l^0)}{S(F_l^0)}, l = 1, \ldots, k,\ \ \phi^0 = \sum_{l=1}^k \lambda_l^0$

$s_l^0 \in \partial S(F_l^0),\ l = 1, \ldots, k$

$L = (i, j_i),\ \forall i \in I$, where $j_i = \arg\max_j \{F_{ij}^0\}$ (L is fixed once I is fixed)

`degenerated` = `false` (this flag is used to indicate that the continuous solution is either strongly degenerated or does not yield a k-partition with a better balanced k-cut)

repeat

  $(F^{t+1}, \delta^{+,\ t+1}, \delta^{-,\ t+1})$ be the optimal solution of the inner problem (4.10)

  $\chi^{t+1} = \mathrm{BCut}(C_1^{t+1}, \ldots, C_k^{t+1})$, where $(C_1^{t+1}, \ldots, C_k^{t+1})$ is obtained from F^{t+1} via rounding (4.6)

  $\lambda_l^{t+1} = \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})}, l = 1, \ldots, k,\ \ \phi^{t+1} = \sum_{l=1}^k \lambda_l^{t+1}$

  $s_l^{t+1} \in \partial S(F_l^{t+1}),\ l = 1, \ldots, k$

  /* solution converged */

  if $\phi^{t+1} = \phi^t$ **then**

    return $(F^t, (C_1^t, \ldots, C_k^t), \texttt{degenerated})$

  end if

  /* early stopping */

  if $\chi^{t+1} < \chi^t$ and $|I| > 0$ **then**

    compute ordering $\mathcal{O}_l, \forall l = 1, \ldots, k$ for $(C_1^{t+1}, \ldots, C_k^{t+1})$ according to (4.13)

    $p = \frac{|I|}{k}$, $\hat{I} = \bigcup_{l=1}^k \mathcal{O}_l^p$, where $\mathcal{O}_l^p$ denotes p top-ranked vertices in $\mathcal{O}_l$

    if $\hat{I} \neq I$ **then**

      return $(F^{t+1}, (C_1^{t+1}, \ldots, C_k^{t+1}), \texttt{degenerated})$

    end if

  end if

until $\chi^{t+1} > \chi^t$ or $(C_1^{t+1}, \ldots, C_k^{t+1})$ is not a k-partition

`degenerated` = `true`

/* We still return F^{t+1} in order to check the convergence of Algorithm 4.2 */

return $(F^{t+1}, (C_1^t, \ldots, C_k^t), \texttt{degenerated})$

Theorem 4.4 *The sequence* $\{(C_1^r, \ldots, C_k^r)\}$ *produced by Algorithm 4.2 satisfies*

$$\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) \leq \mathrm{BCut}(C_1^r, \ldots, C_k^r).$$

Moreover, Algorithm 4.2 terminates after a finite number of iterations with a k-partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ *and at least one of the following conditions is satisfied at the termination.*

1. *The number of membership constraints enforced is* $k \min_l\{|C_l^{r+1}|\}$. *Moreover all vertices in the smallest component of the partition* $(C_1^{r+1}, \ldots, C_k^{r+1})$ *are part of membership constraints.*

2. *Algorithm 4.3 is fully converged for* $I = I^r$ *and* $\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^{r+1})}{S(F_l^{r+1})}$.

Furthermore, let $(C_1^0, \ldots, C_k^0)$ *be a k-partition and* $p_{\max} = \min_l\{|C_l^0|\}$. *If one uses* $F = (\mathbf{1}_{C_1^0}, \ldots, \mathbf{1}_{C_k^0})$ *as an initialization and if the Algorithm 4.2 does not terminate after solving* $\lceil \log_2(p_{\max}) \rceil + 2$ *instances of the problem* (4.10), *then it produces a k-partition* $(C_1^{r+1}, \ldots, C_k^{r+1})$ *satisfying*

$$\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) < \mathrm{BCut}(C_1^0, \ldots, C_k^0).$$

Proof: The monotonicity of the balanced k-cut follows from the observation that the partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ is same as $(C_1^r, \ldots, C_k^r)$ unless Algorithm 4.3 yields a k-partition that achieves strictly smaller balanced k-cut. Note that in every iteration of Algorithm 4.2 either the balanced k-cut decreases or the size of the set I increases. Since the number of possible cuts as well as the maximum size of I is finite, Algorithm 4.2 terminates after a finite number of iterations. At the termination at least one of the following conditions holds: (i) $p = p_{\max}$ (line 16) or (ii) $\chi^{r+1} = \phi^{r+1}$ and $\phi^{r+1} = \phi^r$. If the algorithm terminates because of case (i) then it means that in the current iteration r, Algorithm 4.3 was run with $p = \min_l\{|C_l^{r+1}|\}$ membership constraints per component. These membership constraints I^r were derived from the partition $(C_1^r, \ldots, C_k^r)$. Note, however, that $(C_1^{r+1}, \ldots, C_k^{r+1})$ also yields the same set of membership constraints as I^r; otherwise Algorithm 4.3 would have exited with `degenrated` = `false` (line 19). Thus membership constraints were enforced for all vertices in the smallest component of the partition $(C_1^{r+1}, \ldots, C_k^{r+1})$. In case (ii), Algorithm 4.3 has been run to full convergence for $I = I^r$; otherwise $\phi^{r+1} \neq \phi^r$. Moreover, the continuous objective ϕ^{r+1} and the balanced k-cut χ^{r+1} are equal in this case.

The final statement follows from the fact that the number of membership constraints is first increased by one and then doubled in every iteration until Algorithm 4.3 yields a partition with strictly smaller balanced cut or I^r contains all vertices in the smallest component of the current best k-partition. Here, we will prove the statement by contrapositive method. Assume that Algorithm 4.2 did not produce a k-partition with strictly smaller balanced cut. This implies that Algorithm 4.3 always terminated after one iteration (solving exactly one instance of problem (4.10)) and returns the same partition as its initialization along with the `degenerated` flag set to `true`. Thus, $C_l^r = C_l^0, r > 0, l = 1, \ldots, k$ and consequently the value of $p_{\max}$ is never changed. Moreover p is updated in every iteration r since `degenerated` flag is

`true`. Thus the number of iterations it takes for reaching the condition $p = p_{\max}$ is $1+1+\lceil \log_2(p_{\max}) \rceil$. The first two iterations are for the cases $p = 0$, $p = 1$ and there is an iteration for $p = p_{\max}$ as well. Hence Algorithm 4.2 should terminate after $\lceil \log_2(p_{\max}) \rceil$ +2 iterations if it did not improve the balanced k-cut, which proves the contrapositive of the statement given in the theorem. □

4.4 Multi-class clustering with constraints

In this section, we present our constrained clustering method for the multi-class setting. Similar to the two-class case, we first formulate constrained balanced k-cut problem encoding must-link and cannot-link constraints. We then derive an *equivalent* unconstrained problem which minimizes a trade-off between the balanced k-cut and the constraint violation. We further show that there exists a penalty parameter which guarantees the equivalence of the constrained balanced k-cut problem and its penalized version. We then present a continuous relaxation of the constrained problem. If hard-enforcement is desired, it is straightforward to encode label constraints in our continuous relaxation (4.5) presented in the previous section. Otherwise one can derive must-link and cannot-link constraints from labels and enforce via our soft-formulation.

4.4.1 Formulation of constrained balanced k-cut problem

For completeness, we repeat here the material from the Section 3.2. Let $\mathcal{M} = \{(p,q) : p \in V, q \in V\}$ be the given must-link constraints and $\mathcal{Q} = \{(p,q) : p \in V, q \in V\}$ be the cannot-link constraints. Let $G^c(V, W^c)$ denote the cannot-link constraint graph with the weight matrix W^c whose entries $w^c_{ij} \in [0,1]$, specify the degree of belief for the constrained pair $(i,j) \in \mathcal{Q}$. We make W^c symmetric by setting $w^c_{ji} = w^c_{ij}$ whenever $(i,j) \in \mathcal{Q}$. Similarly we define must-link constraint graph $G^m(V, W^m)$ with degrees of belief w^m_{ij} for each $(i,j) \in \mathcal{M}$. We define θ to be the minimum value among all the degrees-of-belief

$$\theta = \min \left\{ \min_{(p,q)\in\mathcal{M}} \{w^m_{pq}\}, \min_{(p,q)\in\mathcal{Q}} \{w^c_{pq}\} \right\}. \tag{4.14}$$

It is reasonable to assume that $\theta > 0$, i.e., each constraint has a positive degree-of-belief. Let us introduce functions that capture the degree of constraint violation of a given k-partition $(C_1, \ldots, C_k)$. Similarly to the two-class setting, define $\hat{M}, \hat{Q} : P_k \to \mathbb{R}_+$ as

$$\hat{M}(C_1, \ldots, C_k) := \frac{1}{2} \sum_{l=1}^{k} \mathrm{cut}_{G^m}(C_l, \overline{C_l}) \qquad \text{(must-links)}$$

$$\hat{Q}(C_1, \ldots, C_k) := \frac{1}{2} \sum_{l=1}^{k} \mathrm{assoc}_{G^c}(C_l) \qquad \text{(cannot-links)}$$

$$= \frac{1}{2} \mathrm{vol}_{G^c}(V) - \frac{1}{2} \sum_{l=1}^{k} \mathrm{cut}_{G^c}(C_l, \overline{C_l})$$

Let us introduce $\hat{T}(C_1, \ldots, C_k) : P_k \to \mathbb{R}_+$ to capture the total amount of constraint violation of a partition $(C_1, \ldots, C_k)$,

$$\hat{T}(C_1, \ldots, C_k) := \hat{M}(C_1, \ldots, C_k) + \hat{Q}(C_1, \ldots, C_k).$$

Note that similar to the two-class setting, $\hat{T}$ is non-negative and each violated constraint increases $\hat{T}$ by the corresponding degree-of-belief. If a k-partition $(C_1, \ldots, C_k)$ satisfies all the constraints, then $\hat{T}(C_1, \ldots, C_k) = 0$. We define a partition $(C_1, \ldots, C_k)$ as consistent if $\hat{T}(C_1, \ldots, C_k) = 0$.

The multi-class constrained clustering problem can then be formulated as the following constrained balanced cut problem

$$\begin{aligned} &\min_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)} \\ &\text{subject to} : \hat{T}(C_1, \ldots, C_k) = 0. \end{aligned} \tag{4.15}$$

We show that there exists an equivalent unconstrained formulation given by

$$\min_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)} + \gamma \hat{T}(C_1, \ldots, C_k) \tag{4.16}$$

for a specific choice of γ. Before establishing the equivalence let us characterize the relation between the parameter γ and the amount of constraint violation of the solution of (4.16).

Lemma 4.3 *Let $(C_1, \ldots, C_k)$ be a consistent partition (i.e., it satisfies all the constraints) and $\chi_0 = \mathrm{BCut}(C_1, \ldots, C_k)$. If $\gamma > \frac{\chi_0}{(l+1)\theta}$, where θ is defined in (4.14), then any minimizer $(C_1^*, \ldots, C_k^*)$ of the problem (4.16) violates no more than l constraints.*

Proof: First note that χ_0 is the objective value of the partition $(C_1, \ldots, C_k)$ for the problem (4.16). Assume for the sake of contradiction that a minimizer $(C_1^*, \ldots, C_k^*)$ of (4.16) violates at least $l+1$ constraints. Then it holds that $\hat{T}(C_1^*, \ldots, C_k^*) \geq (l+1)\theta$, since θ is the minimum of all the degrees-of-belief. Let ψ^* denote the objective value of $(C_1^*, \ldots, C_k^*)$ for the problem (4.16). Then we have for the given value of γ,

$$\chi^* = \mathrm{BCut}(C_1^*, \ldots, C_k^*) + \gamma \hat{T}(C_1^*, \ldots, C_k^*) \geq \gamma(l+1)\theta > \chi_0,$$

which is a contradiction since ψ^* is the optimal value and χ_0 is the objective value of the partition $(C_1, \ldots, C_k)$. Hence any partition $(C_1^*, \ldots, C_k^*)$ which violates more than l constraints cannot be a solution of problem (4.16). □

Theorem 4.5 *Let $(C_1, \ldots, C_k)$ be a consistent partition (i.e., it satisfies all the constraints) and $\chi_0 = \mathrm{BCut}(C_1, \ldots, C_k)$. If $\gamma > \frac{\chi_0}{\theta}$, where θ is defined in (4.14), then it holds that*

$$\operatorname*{arg\,min}_{\substack{(C_1,\ldots,C_k)\in P_k \\ \hat{T}(C_1,\ldots,C_k)=0}} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)} = \operatorname*{arg\,min}_{(C_1,\ldots,C_k)\in P_k} \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)} + \gamma \hat{T}(C_1, \ldots, C_k)$$

and the optimum values of both problems are equal.

Proof: By Lemma 4.3 with $l = 0$, any minimizer of the unconstrained problem does not violate any constraint. Moreover $\hat{T}(C_1, \ldots, C_k) = 0$ for any consistent partition and hence the objective values of both problem are equal for consistent partitions. Thus the equivalence holds. □

The choice of γ depends on the balanced cut value of a partition that satisfies all the given constraints. In principle, any upper bound on the balanced cut value of a consistent partition can be used to determine γ.

Note that when $k = 2$, the multi-class formulation (4.16) specializes to a slightly different form than the two-class formulation (3.4) presented in Chapter 3. In fact the multi-class formulation is better because it does not introduce any bias towards balanced partitions unlike the two-class formulation. However, as we see in the experiments, the multi-class formulation is more difficult to optimize. Hence, we recommend using the two-class formulation when $k = 2$.

Unlike the two-class setting, the problem of determining whether the constraints in the multi-class setting are consistent is NP-complete. Although the must-link constraints can be eliminated by merging them and transferring the problem to a reduced graph (see Section 3.3), finding a partition that is consistent with the cannot-link constraints in the multi-class setting is NP-hard. This is because the problem of finding whether the given cannot-link constraints are consistent can be formulated as a graph k-coloring problem which is NP-complete [65].

In practice to find the optimal coloring, one uses a greedy algorithm [118] which visits the vertices in the decreasing order of their degrees and assigns a color corresponding to the smallest integer that is not used by its neighbors. The greedy algorithm may use more than k colors. So in general, it may not be possible to find a consistent partition efficiently. Here we specify the special cases where one can use the greedy algorithm to find a partition satisfying the given set of constraints.

1. **Constraints generated from labels:** In this case enforcing all must-link results in a single k-clique from which it is straightforward to obtain a consistent partition.

2. **Sparse constraints:** Any graph with a maximum degree of d can be colored using $d + 1$ colors [118]. Thus, if the degree of any vertex in the unweighted version of the cannot-link constraint graph G^c is smaller than k then one can find a consistent partition. This happens for example if the constraints are pairwise disjoint.

Since in practice the given pairwise constraints may not necessarily fall into these categories, one needs a generic version of Theorem 4.5 providing a choice for the parameter γ when one cannot find a consistent partition. If one has access to any partition violating p constraints, one can at best hope to find a partition from the problem (4.16) that violates no more than p constraints in polynomial time. The following theorem gives the value of γ to be used to get such a guarantee. Since in the following result we are interested in enforcing as many constraints as possible we assume that the degrees-of-belief are all set to the maximum value, i.e., $\theta = 1$.

Theorem 4.6 *Let $(C_1, \ldots, C_k)$ be a k-partition violating p constraints given in $\mathcal{M}$ and $\mathcal{Q}$ where the degrees-of-belief for all constraints is 1 and $\chi = \mathrm{BCut}(C_1, \ldots, C_k)$. If $\gamma > \frac{\chi}{(l+1-p)}$, where $l \geq p$, then any minimizer $(C_1^*, \ldots, C_k^*)$ of the problem (4.16) violates no more than l constraints.*

Proof: The given condition on γ implies that

$$\gamma(l+1-p) > \chi \implies \gamma(l+1) > \chi + \gamma p.$$

Since $(C_1, \ldots, C_k)$ violates p constraints, we have $\hat{T}(C_1, \ldots, C_k) = p$ (since the degree-of-belief $\theta = 1$). Hence $\chi + \gamma p$ is the objective value of the partition $(C_1, \ldots, C_k)$ for the problem (4.16). Assume for the sake of contradiction that a minimizer $(C_1^*, \ldots, C_k^*)$ of (4.16) violates at least $l+1$ constraints. Then it holds that $\hat{T}(C_1^*, \ldots, C_k^*) \geq (l+1)$. Let ψ^* denote the objective value of $(C_1^*, \ldots, C_k^*)$ for the problem (4.16). Then we have for the given value of γ,

$$\psi^* = \mathrm{BCut}(C_1^*, \ldots, C_k^*) + \gamma \hat{T}(C_1^*, \ldots, C_k^*) \geq \gamma(l+1) > \chi + \gamma p,$$

which is a contradiction since ψ^* is the optimal value and $\chi + \gamma p$ is the objective value of the partition $(C_1, \ldots, C_k)$. Hence any partition $(C_1^*, \ldots, C_k^*)$ which violates more than l constraints cannot be a solution of problem (4.16). □

Note that Theorem 4.5 is a special case of Theorem 4.6 with $l = p = 0$. Unlike Theorem 4.5, the latter result gives us a practical choice for γ in situations where we cannot obtain a consistent partition.

4.4.2 Continuous relaxation of constrained balanced k-cut problem

In this section we discuss the continuous relaxation of the problem (4.16). In contrast to the unconstrained problem (4.1), we have an additional penalty term $\hat{T}$ in the objective of (4.16) given by

$$\hat{T}(C_1, \ldots, C_k) = \frac{1}{2} \sum_{l=1}^{k} \mathrm{cut}_{G^m}(C_l, \overline{C_l}) + \frac{1}{2} \mathrm{vol}_{G^c}(V) - \frac{1}{2} \sum_{l=1}^{k} \mathrm{cut}_{G^c}(C_l, \overline{C_l}). \quad (4.17)$$

Since the second term is a constant we can eliminate it without changing the minimizers of the problem (4.16). Similar to the unconstrained case, we will replace the remaining terms in the above sum which are set functions by their Lovász extensions. This leads to the following continuous problem

$$\min_{\substack{F=(F_1,\ldots,F_k),\\ F \in \mathbb{R}_+^{n \times k}}} \sum_{l=1}^{k} \frac{\mathrm{TV}(F_l)}{S(F_l)} + \gamma T(F) \quad (4.18)$$

$$\begin{aligned} \text{subject to}: \; & F_{(i)} \in \Delta_k, && i = 1, \ldots, n, && \text{(simplex constraints)} \\ & \max\{F_{(i)}\} = 1, && \forall i \in I, && \text{(membership constraints)} \\ & S(F_l) \geq m, && l = 1, \ldots, k, && \text{(size constraints)} \end{aligned}$$

where S is the Lovász extension of the set function $\hat{S}$ and $m = \min_{C \subsetneq V,\, C \neq \emptyset} \hat{S}(C)$ and $T(F) = \frac{1}{2} \sum_{l=1}^{k} \mathrm{TV}_{G^m}(F_l) - \frac{1}{2} \sum_{l=1}^{k} \mathrm{TV}_{G^c}(F_l)$.

Similarly to the unconstrained case, the following result shows that the problem (4.18) is a relaxation of the combinatorial problem (4.16) and the relaxation is exact if $I = V$.

Proposition 4.2 *The continuous problem* (4.18) *is a relaxation of the combinatorial optimization problem* (4.16). *If $I = V$, the relaxation is exact, i.e., both problems are equivalent in the sense that there is one-to-one correspondence between the minimizers of these problems. Moreover, let $(C_1, \ldots, C_k)$ be a partition violating at most p constraints given in $\mathcal{M}$ and $\mathcal{Q}$ where the degrees-of-belief for all constraints is* 1 *and $\chi = \mathrm{BCut}(C_1, \ldots, C_k)$. If $\gamma > \frac{\chi}{(l+1-p)}$ and $l \geq p$, then the partition constructed from any solution of the continuous relaxation* (4.18) *violates at most l constraints.*

Proof: Proof is similar to that of Proposition 4.1. The second part follows from Theorem 4.6. □

The following result shows that the constrained balanced cut problem (4.15) and the above continuous relaxation are equivalent for a specific choice of γ.

Theorem 4.7 *Let $G(V, W)$ be a weighted undirected graph and $\hat{S}$ be any non-negative balancing function with $\hat{S}(\emptyset) = 0$, $\hat{S}(C) > 0, \forall C \subsetneq V$, $C \neq \emptyset$ and $\hat{T}$ be the penalty function defined in* (4.17) *for the constraint graphs G^m and G^c. Further let $(C_1, \ldots, C_k)$ be a consistent partition (i.e., $\hat{T}(C_1, \ldots, C_k) = 0$) and $\chi_0 = \mathrm{BCut}(C_1, \ldots, C_k)$. If $\gamma > \frac{\chi_0}{\theta}$, where θ is defined in* (4.14), *and $I = V$ then the constrained balanced k-cut problem* (4.15) *and the continuous relaxation* (4.18) *are equivalent in the sense that there is a one-to-one correspondence between the minimizers of these problems.*

Proof: Proof follows from Proposition 4.2 and Theorem 4.5. □

4.4.3 Algorithm for the continuous relaxation

The monotonic descent algorithm developed in Section 4.3 can directly be used to solve the continuous problem (4.18). Similar to the unconstrained case, we first rewrite the continuous relaxation (4.18) as

$$\min_{\substack{F=(F_1,\ldots,F_k),\\ F\in\mathbb{R}_+^{n\times k},\ \beta\in\mathbb{R}_+^k}} \sum_{l=1}^{k} \beta_l + \gamma T(F) \tag{4.19}$$

$$\begin{aligned}
\text{subject to}: \ & \mathrm{TV}(F_l) \leq \beta_l S(F_l), && l = 1, \ldots, k, && \text{(descent constraints)} \\
& F_{(i)} \in \Delta_k, && i = 1, \ldots, n, && \text{(simplex constraints)} \\
& \max\{F_{(i)}\} = 1, && \forall i \in I, && \text{(membership constraints)} \\
& S(F_l) \geq m, && l = 1, \ldots, k. && \text{(size constraints)}
\end{aligned}$$

In contrast to the unconstrained case (4.9), the objective function in (4.19) is also non-convex. In fact the objective function is already expressed as a difference of convex functions. Our descent algorithm is based on linearizing the concave part of

the objective apart from approximating the constraint set by a convex set (similar to (4.10)) and solving the resulting convex problem in each iteration t,

$$\begin{aligned}\min_{\substack{F\in\mathbb{R}_+^{n\times k},\\ \delta^+\in\mathbb{R}_+^k,\ \delta^-\in\mathbb{R}_+^k}} \quad & \sum_{l=1}^k \delta_l^+ - \delta_l^- + \gamma\frac{1}{2}\sum_{l=1}^k \mathrm{TV}_{G^m}(F_l) - \gamma\frac{1}{2}\sum_{l=1}^k \left\langle r_l^t, F_l\right\rangle \qquad (4.20)\\ \text{subject to}: \ & \mathrm{TV}(F_l) \le \lambda_l^t \left\langle s_l^t, F_l\right\rangle + \delta_l^+ m - \delta_l^- M, \quad l=1,\dots k,\\ & F_{(i)} \in \Delta_k, \quad i=1,\dots,n,\\ & F_{ij_i} = 1, \quad \forall (i,j_i)\in L,\\ & \left\langle s_l^t, F_l^t\right\rangle \ge m, \quad l=1,\dots,k,\end{aligned}$$

where r_l^t and s_l^t are elements of subdifferentials of TV_{G^c} and S at F_l^t respectively and $\lambda_l^t = \frac{\mathrm{TV}(F_l^t)}{S(F_l^t)}, l=1,\dots,k$. We repeat the process until the sequence terminates, that is no further descent is possible. The following theorem shows the monotonic descent property of this procedure.

Theorem 4.8 *The sequence $\{F^t\}$ produced by the above algorithm satisfies*

$$\sum_{l=1}^k \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} + \gamma T(F^{t+1}) < \sum_{l=1}^k \frac{\mathrm{TV}(F_l^t)}{S(F_l^t)} + \gamma T(F^t)$$

for all $t \ge 0$ or the algorithm terminates.

Proof: Let $(F^{t+1}, \delta^{+,\,t+1}, \delta^{-,\,t+1})$ be the optimal solution of the inner problem (4.20). One can proceed similar to the proof of Theorem 4.3 to first show that

$$\sum_{l=1}^k \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} \le \sum_{l=1}^k \lambda_l^t + \sum_{l=1}^k \delta_l^{+,\,t+1} - \delta_l^{-,\,t+1}. \qquad (4.21)$$

Since TV is a convex 1-homogeneous function and $\gamma \ge 0$, we have by Lemma 2.1 and the definition of subgradient,

$$-\gamma\frac{1}{2}\sum_{l=1}^k \mathrm{TV}_{G^c}(F_l^{t+1}) \le -\gamma\frac{1}{2}\sum_{l=1}^k \left\langle r_l^t, F_l^{t+1}\right\rangle. \qquad (4.22)$$

Moreover, we have by Lemma 2.1

$$\frac{1}{2}\sum_{l=1}^k \left\langle r_l^t, F_l^t\right\rangle = \frac{1}{2}\sum_{l=1}^k \mathrm{TV}_{G^c}(F_l^t). \qquad (4.23)$$

Noting that $\delta_l^+ = \delta_l^- = 0,\ F = F^t$ is feasible for (4.20), the optimal value has to be either strictly smaller than

$$\gamma\frac{1}{2}\sum_{l=1}^k \mathrm{TV}_{G^m}(F_l^t) - \gamma\frac{1}{2}\sum_{l=1}^k \left\langle r_l^t, F_l^t\right\rangle$$

or the previous iterate F^t together with $\delta_l^+ = \delta_l^- = 0$ is already optimal and hence the algorithm terminates. In the latter case there is nothing to prove. In the former case, we have by (4.23)

$$\begin{aligned}\gamma T(F^t) &= \gamma\frac{1}{2}\sum_{l=1}^{k}\mathrm{TV}_{G^m}(F_l^t) - \gamma\frac{1}{2}\sum_{l=1}^{k}\langle r_l^t, F_l^t\rangle \\ &> \sum_{l=1}^{k}\delta_l^{+,\,t+1} - \delta_l^{-,\,t+1} + \gamma\frac{1}{2}\sum_{l=1}^{k}\mathrm{TV}_{G^m}(F_l^{t+1}) - \gamma\frac{1}{2}\sum_{l=1}^{k}\langle r_l^t, F_l^{t+1}\rangle \\ &\geq \sum_{l=1}^{k}\frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})} - \sum_{l=1}^{k}\lambda_l^t + \gamma\frac{1}{2}\sum_{l=1}^{k}\mathrm{TV}_{G^m}(F_l^{t+1}) - \gamma\frac{1}{2}\sum_{l=1}^{k}\mathrm{TV}_{G^c}(F_l^{t+1}),\end{aligned}$$

where the last inequality follows from (4.21) and (4.22). Rearranging the terms would then yield the desired result.

□

Similar to the inner problem presented in the previous section (see Lemma 4.1), the above problem can be rewritten as a linear programming problem and can be solved by PDHG.

The overall algorithmic scheme for solving the combinatorial problem (4.16) is given in Algorithm 4.4. The method for solving the continuous relaxation (4.19) in each iteration of Algorithm 4.4 is given in Algorithm 4.5. We follow the same procedure as discussed in 4.3.2 for selecting the subset I of membership constraints except that the ordering $\mathcal{O}_l$ is now computed based on the objective function of the problem (4.16) instead of the balanced cut,

$$\begin{aligned}b_{li} &= \frac{\mathrm{cut}\big(C_l\backslash\{i\},\, \overline{C_l}\cup\{i\}\big)}{\hat{S}(C_l\backslash\{i\})} + \gamma\hat{T}_0(C_l\backslash\{i\}) \qquad (4.24)\\ &+ \min_{s\neq l}\left[\frac{\mathrm{cut}\big(C_s\cup\{i\},\, \overline{C_s}\backslash\{i\}\big)}{\hat{S}(C_s\cup\{i\})} + \gamma\hat{T}_0(C_s\cup\{i\}) + \sum_{j\neq l,\ j\neq s}\frac{\mathrm{cut}(C_j, \overline{C_j})}{\hat{S}(C_j)} + \gamma\hat{T}_0(C_j)\right],\end{aligned}$$

where $(C_1\ldots, C_k)$ is the current best partition $\hat{T}_0 : 2^V \to \mathbb{R}$ is defined as

$$\hat{T}_0(C) := \frac{1}{2}\mathrm{cut}_{G^m}(C, \overline{C}) - \frac{1}{2}\mathrm{cut}_{G^c}(C, \overline{C}). \qquad (4.25)$$

By the same reasoning given in Lemma 4.2, the initialization F^r received by Algorithm 4.5 in every iteration r of Algorithm 4.4 is feasible for the continuous relaxation (4.19) with $I = I^r$. Similarly to Algorithm 4.2, Algorithm 4.4 terminates after a finite number of iterations with a k-partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ and at least one of the following conditions is satisfied at the termination: (i) the number of membership constraints enforced is $k\min_l\{\big|C_l^{r+1}\big|\}$ and all vertices in the smallest component of the partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ are part of membership constraints (ii) Algorithm 4.5 is fully converged for $I = I^r$ and $\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) + \gamma\hat{T}(C_1^{r+1}, \ldots, C_k^{r+1}) = \sum_{l=1}^{k}\frac{\mathrm{TV}(F_l^{r+1})}{S(F_l^{r+1})} + \gamma T(F^{r+1})$. We now present the additional guarantee Algorithm 4.4 provides for the constrained balanced k-cut problem.

Algorithm 4.4 Minimization of the constrained balanced k-cut problem (4.16)

Input: γ and $F^0 \in \mathbb{R}_+^{n\times k}$ be such that $F^0\mathbf{1}_k = \mathbf{1}_n$, $S(F_l) \geq m, l = 1\ldots,k$ and rounding F^0 according to (4.6) yields a k-partition
Output: a k-partition $(C_1,\ldots,C_k)$
$\phi^0 = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^0)}{S(F_l^0)} + \gamma T(F^0)$, $I^0 = \emptyset$, $p = 0$
$\chi^0 = \mathrm{BCut}(C_1^0,\ldots,C_k^0) + \gamma\hat{T}(C_1^0,\ldots,C_k^0)$, where $(C_1^0,\ldots,C_k^0)$ be the partition obtained from F^0 via rounding (4.6)
$p_{\max} = \min_l\{|C_l^0|\}$
repeat
  if $I^r \neq \emptyset$ **then**
    $F_{ij}^r = 0,\ \forall i \in I^r,\ \forall j \in \{1,\ldots,k\},\ F_{ij_i}^r = 1,\ \forall i \in I^r$ where j_i is the component index such that $i \in C_{j_i}^r$.
    $\phi^r = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^r)}{S(F_l^r)} + \gamma T(F^r)$
  end if
  Let $(F^{r+1},(C_1^{r+1},\ldots,C_k^{r+1}),\texttt{degenerated})$ be the output obtained from Algorithm 4.5 with initialization F^r, $(C_1^r,\ldots,C_k^r)$ and the membership constraints I^r
  $\phi^{r+1} = \sum_{l=1}^k \frac{\mathrm{TV}(F_l^{r+1})}{S(F_l^{r+1})} + \gamma T(F^{r+1})$, $\chi^{r+1} = \mathrm{BCut}(C_1^{r+1},\ldots,C_k^{r+1}) + \gamma\hat{T}(C_1^{r+1},\ldots,C_k^{r+1})$
  compute ordering $\mathcal{O}_l, \forall l = 1,\ldots,k$ for $(C_1^{r+1},\ldots,C_k^{r+1})$ according to (4.24)
  if `degenerated` **then**
    $p_{\max} = \min_l\{\left|C_l^{r+1}\right|\}$
    if $p = p_{\max}$ **then**
      break (labels of all vertices in a component were enforced)
    else
      $p = \max\{\min\{p_{\max}, 2p\}, 1\}$ (increase the number of membership constraints)
    end if
  end if
  $I^{r+1} = \bigcup_{l=1}^k \mathcal{O}_l^p$, where $\mathcal{O}_l^p$ denotes p top-ranked vertices in $\mathcal{O}_l$
until $\chi^{r+1} = \phi^{r+1}$ and $\phi^{r+1} = \phi^r$
return $((C_1^{r+1},\ldots,C_k^{r+1}), F^{r+1})$

Algorithm 4.5 Minimization of the continuous relaxation (4.19) **of the constrained balanced k-cut problem**

Input: I, $F^0 \in \mathbb{R}_+^{n\times k}$ feasible for the problem (4.19), a k-partition $(C_1^0, \ldots, C_k^0)$ and γ

Output: F which is feasible for the problem (4.19) and achieves monotonic descent in the objective of (4.19), a k-partition $(C_1, \ldots, C_k)$ and `degenerated` flag

$\chi^0 = \mathrm{BCut}(C_1^0, \ldots, C_k^0) + \gamma \hat{T}(C_1^0, \ldots, C_k^0),\ \lambda_l^0 = \frac{\mathrm{TV}(F_l^0)}{S(F_l^0)}, l = 1, \ldots, k,\quad \phi^0 = \sum_{l=1}^k \lambda_l^0 + \gamma T(F^0)$

$s_l^0 \in \partial S(F_l^0),\ r_l^0 \in \partial \mathrm{TV}_{G^c}(F^0),\ l = 1, \ldots, k$

$L = (i, j_i),\ \forall i \in I$, where $j_i = \arg\max_j \{F_{ij}^0\}$ (L is fixed once I is fixed)

`degenerated` = `false`

repeat

  $(F^{t+1}, \delta^{+,\,t+1}, \delta^{-,\,t+1})$ be the optimal solution of the inner problem (4.20)

  $\chi^{t+1} = \mathrm{BCut}(C_1^{t+1}, \ldots, C_k^{t+1}) + \gamma \hat{T}(C_1^{t+1}, \ldots, C_k^{t+1})$, where $(C_1^{t+1}, \ldots, C_k^{t+1})$ is obtained from F^{t+1} via rounding (4.6)

  $\lambda_l^{t+1} = \frac{\mathrm{TV}(F_l^{t+1})}{S(F_l^{t+1})}, l = 1, \ldots, k,\ \ \phi^{t+1} = \sum_{l=1}^k \lambda_l^{t+1} + \gamma T(F^{t+1})$

  $s_l^{t+1} \in \partial S(F_l^{t+1}),\ r_l^{t+1} \in \partial \mathrm{TV}_{G^c}(F^{t+1}),\ l = 1, \ldots, k$

  /* solution converged */

  if $\phi^{t+1} = \phi^t$ **then**

    return $(F^{t+1}, (C_1^{t+1}, \ldots, C_k^{t+1}), \texttt{degenerated})$

  end if

  /* early stopping */

  if $\chi^{t+1} < \chi^t$ and $|I| > 0$ **then**

    compute ordering $\mathcal{O}_l, \forall l = 1, \ldots, k$ for $(C_1^{t+1}, \ldots, C_k^{t+1})$ according to (4.24)

    $p = \frac{|I|}{k}$, $\hat{I} = \bigcup_{l=1}^k \mathcal{O}_l^p$, where $\mathcal{O}_l^p$ denotes p top-ranked vertices in $\mathcal{O}_l$

    if $\hat{I} \neq I$ **then**

      return $(F^{t+1}, (C_1^{t+1}, \ldots, C_k^{t+1}), \texttt{degenerated})$

    end if

  end if

until $\chi^{t+1} > \chi^t$ or $(C_1^{t+1}, \ldots, C_k^{t+1})$ is not a k-partition

`degenerated` = `true`

/* We still return F^{t+1} in order to check the convergence in Algorithm 4.4 */

return $(F^{t+1}, (C_1^t, \ldots, C_k^t), \texttt{degenerated})$

Theorem 4.9 *The sequence* $\{(C_1^r, \ldots, C_k^r)\}$ *produced by Algorithm 4.4 satisfies*

$$\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) + \gamma \hat{T}(C_1^{r+1}, \ldots, C_k^{r+1}) \leq \mathrm{BCut}(C_1^r, \ldots, C_k^r) + \gamma \hat{T}(C_1^r, \ldots, C_k^r).$$

Let $(C_1^0, \ldots, C_k^0)$ *be a* k*-partition violating* v *constraints specified in* $\mathcal{M}$ *and* $\mathcal{Q}$ *where the degree-of-belief for all constraints is* 1*. Moreover, let* $p_{\max} = \min_l\{|C_l^0|\}$*. If one uses* $F = (\mathbf{1}_{C_1^0}, \ldots, \mathbf{1}_{C_k^0})$ *as an initialization with* $\gamma > \mathrm{BCut}(C_1^0, \ldots, C_k^0)$ *and if our algorithm does not terminate in* $\lceil \log_2(p_{\max}) \rceil + 2$ *iterations, then it produces a* k*-partition* $(C_1^{r+1}, \ldots, C_k^{r+1})$ *that satisfies at least one of the following conditions:*

1. $\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) < \mathrm{BCut}(C_1^0, \ldots, C_k^0)$ *while violating at most* v *constraints.*
2. $(C_1, \ldots, C_k)$ *violates at most* $v - 1$ *constraints.*

Proof: The monotonicity of the objective of problem (4.16) follows from the observation that the partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ is same as $(C_1^r, \ldots, C_k^r)$ unless Algorithm 4.5 yields a k-partition that achieves strictly smaller objective of problem (4.16). By similar reasoning given in the proof of Theorem 4.4, one can show that if Algorithm 4.4 does not terminate in $\lceil \log_2(p_{\max}) \rceil + 2$ iterations, then the partition $(C_1^{r+1}, \ldots, C_k^{r+1})$ it found achieves strict descent in the objective of problem (4.16),

$$\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) + \gamma \hat{v} < \mathrm{BCut}(C_1^0, \ldots, C_k^0) + \gamma v, \tag{4.26}$$

where $\hat{v}$ is the number of constraints violated by $(C_1^{r+1}, \ldots, C_k^{r+1})$. It must hold that $\hat{v} \leq v$; otherwise the penalty term is at least $\gamma(v+1)$ and

$$\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) + \gamma \hat{v} \geq \gamma(v+1) > \gamma v + \mathrm{BCut}(C_1^0, \ldots, C_k^0),$$

which is a contradiction since (4.26) holds. Thus we have $\hat{v} \leq v$. We have two possibilities: (i) $\hat{v} = v$ in which case (4.26) implies that $\mathrm{BCut}(C_1^{r+1}, \ldots, C_k^{r+1}) < \mathrm{BCut}(C_1^0, \ldots, C_k^0)$; (ii) $\hat{v} < v$ in which case our solution satisfies at least one more constraint than the initialization; moreover a strict descent in the balanced cut is also possible in this case. □

A special case of the above result corresponds to the choice $p = 0$. If any consistent partition is used as an initialization, then our algorithm, if not terminated in $\lceil \log_2(I_{\max}) \rceil + 2$ iterations, is guaranteed to produce a consistent partition with strictly smaller balanced cut.

In practice, similarly to the two-class case, we solve the continuous relaxation (4.18) for a sequence of γ starting with $\gamma^0 = 0$. We use the same strategy as described in Section 3.5.2 for updating γ except that here we use an upper bound on γ to terminate if we are unable to find a consistent partition. This upper bound can easily be derived as follows. If one uses any $\gamma > \max_{(C_1,\ldots,C_k)\in P_k} \mathrm{BCut}(C_1, \ldots, C_k)$, then it is clear that such a γ is also larger than the balanced cut value of a consistent partition (if one exists) and there is no need to further increase γ as it already yields equivalence (provided constraints are consistent). Thus we use a value larger than $\max_{(C_1,\ldots,C_k)\in P_k} \mathrm{BCut}(C_1, \ldots, C_k)$ as an upper bound on γ. For normalized cut (as well as for its Cheeger variants) one possible upper bound is k, since each ratio $\frac{\mathrm{cut}(C_l, \overline{C_l})}{\hat{S}(C_l)} \leq 1,\ l = 1, \ldots, k$. Note that if our algorithm does not return a consistent solution for this value of γ then it does not mean that the constraints are inconsistent because our algorithm could not have converged to the global optimum.

4.5 Experiments

4.5.1 Unconstrained clustering

We evaluate our method against a diverse selection of state-of-the-art clustering methods like spectral clustering (Spec) [115], 1-Spectral clustering (1Spec) which computes the k-clustering using recursive bi-partitioning where each two-class problem is solved via the exact continuous relaxation given in [56], Graclus[1] [31], NMF-based approaches like PNMF [124], NSC [33], ONMF [34], LSD [9], NMFR [123] and MTV [19] which optimizes (4.4). The details of these methods are given in Section 2.2.6. We used the publicly available code [123, 19] with default settings. We run our method using 5 random initializations, 7 initializations based on the spectral clustering solution similar to [19]. These multiple runs for our method are performed in parallel in all the experiments.

Datasets:

	Iris	Wine	Vertebral	Ecoli	4Moons	WebKb4
n	150	178	310	336	4000	4196
k	3	3	3	6	4	4

	OptDigits	USPS	PenDigits	20News	MNIST
n	5620	9298	10992	19928	70000
k	10	10	10	20	10

In addition to the datasets provided in [19], we also selected a variety of datasets from the UCI repository. For all the datasets not in [19], symmetric K-NN graphs are built with Gaussian weights $w_{ij} = \exp\left(-\frac{s\|x_i - x_j\|_2^2}{\min\{\sigma_i^2, \sigma_j^2\}}\right)$, where σ_i is the Euclidean distance of x_i to its K^{th}-nearest neighbor. We chose the parameters s and K in a *method independent way* by testing for each dataset several graphs using all the methods over different choices of $s \in \{0.1, 1, 4\}$ and $K \in \{3, 5, 7, 10, 15, 20, 40, 60, 80, 100\}$. The best choice in terms of the clustering error across all the methods and datasets, is $s = 1, k = 15$. Since we are using classification datasets, we evaluate the quality of clustering result based on how well each cluster agrees with the known class structure. More precisely, let $(C_1, \ldots, C_k)$ be the clustering result and Y_i denote the true class label of the data point i. Moreover, let y'_l denote the dominant class label in C_l, $l = 1, \ldots, k$. Then the error of the k-partition $(C_1, \ldots, C_k)$ is computed as

$$\text{error}(C_1, \ldots, C_k) = \frac{1}{n} \sum_{l=1}^{k} \sum_{i \in C_l} \delta_{Y_i \neq y'_l}, \tag{4.27}$$

where n is the total number of points and $\delta_c = 1$, if c is true and 0 otherwise.

[1] Since [31], a multi-level algorithm directly minimizing Rcut/Ncut, is shown to be superior to METIS [66], we do not compare with [66].

Quantitative results: In our first experiment we evaluate our method in terms of solving the balanced k-cut problem for various balancing functions, data sets and graph parameters. In practice, we found that ratio/normalized cut typically require more number of membership constraints than the corresponding (symmetric) Cheeger versions and consequently result in sub-optimal solutions. This difference is due to the fact that balancing functions of ratio/normalized cut are modular (which prefer the full set V as solution for each ratio when there is no partition constraint) while the Cheeger versions have submodular balancing functions (where the full set V is heavily penalized). Note that ratio and normalized cuts and their Cheeger versions are equal whenever each component of a k-partition satisfies $|C_l| \leq \frac{|V|}{2}$ respectively $\mathrm{vol}(C_l) \leq \frac{\mathrm{vol}(V)}{2}$, $l = 1, \ldots, k$. Since this is a reasonable assumption in the multi-class setting (i.e., no cluster is larger than half the "size" of the dataset), we used Cheeger balancing functions for solving ratio and normalized cuts too. However we report the actual ratio/normalized cut values of the solutions found using Cheeger balancing functions.

Table 4.1 reports the fraction of times a method achieves the best as well as strictly best balanced k-cut over all constructed graphs and datasets (in total 30 graphs per dataset). For reference, we also report in *italic* the obtained cuts for other clustering methods although they do not directly minimize this criterion; methods that directly optimize the criterion are shown in normal font. Our algorithm can handle all balancing functions and significantly outperforms all other methods across all criteria. For ratio and normalized cut cases we achieve better results than [115, 56, 31] which directly optimize this criterion. This shows that the greedy recursive bi-partitioning affects badly the performance of [56], which, otherwise, was shown to obtain the best cuts on several benchmark datasets [109]. This further shows the need for methods that directly minimize the multi-cut. It is striking that the competing method of [19], which directly minimizes the asymmetric ratio cut, is beaten significantly by Graclus as well as our method. As this clear trend is less visible in the qualitative experiments, we suspect that extreme graph parameters lead to fast convergence to a degenerate solution.

Qualitative results: We evaluate all methods on the graphs built with the parameters $K = 15$, $s = 1$ for all datasets. In Table 4.2, we report the clustering errors, the balanced k-cuts as well as the runtimes (in seconds) of all methods. As the main goal is to compare to MTV we choose their balancing function (RCC-asym). Our method always achieved the best cuts across all datasets; for 7 out of the 11 datasets our method achieved strictly best cuts[2]. Although, MTV directly optimizes the asymmetric ratio Cheeger cut, it achieved best cut value only for two datasets. For WebKb4 and 20News, its cut values are worse than the cuts found by the standard spectral clustering. For 20News dataset, its cut value is 1.6 times higher than the cut value found by our method where as for WebKb4, its cut value is approximately 1.5 times higher than ours. It is interesting to note that 1Spec achieved the best cut value for three datasets in spite of not minimizing the asymmetric ratio Cheeger cut directly. Although Graclus is very fast, its cut values for some of the datasets (20News, WebKb4) are much worse. The NMF-based

[2]We remark that the cuts shown here are rounded to two decimal points; the reported values for MTV and our method for 4Moons dataset look same but the actual values are different.

		Ours	MTV	1Spec	Spec	Graclus	PNMF	NSC	ONMF	LSD	NMFR
RCC-asym	Best (%)	**80.54**	25.50	*23.49*	*7.38*	*38.26*	*2.01*	*5.37*	*2.01*	*4.03*	*1.34*
	S. Best (%)	**44.97**	10.74	*1.34*	*0.00*	*4.70*	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*
RCC-sym	Best (%)	**94.63**	*8.72*	*19.46*	*6.71*	*37.58*	*0.67*	*4.03*	*0.00*	*0.67*	*0.67*
	S. Best (%)	**61.74**	*0.00*	*0.67*	*0.00*	*4.70*	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*
NCC-asym	Best (%)	**93.29**	*13.42*	*20.13*	*10.07*	*38.26*	*0.67*	*5.37*	*2.01*	*4.70*	*2.01*
	S. Best (%)	**56.38**	*2.01*	*0.00*	*0.00*	*2.01*	*0.00*	*0.00*	*0.67*	*0.00*	*1.34*
NCC-sym	Best (%)	**98.66**	*10.07*	*20.81*	*9.40*	*40.27*	*1.34*	*4.03*	*0.67*	*3.36*	*1.34*
	S. Best (%)	**59.06**	*0.00*	*0.00*	*0.00*	*1.34*	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*
Rcut	Best (%)	**85.91**	*7.38*	20.13	10.07	32.89	*0.67*	*4.03*	*0.00*	*1.34*	*1.34*
	S. Best (%)	**58.39**	*0.00*	2.68	2.01	8.72	*0.00*	*0.00*	*0.00*	*0.00*	*0.67*
Ncut	Best (%)	**95.97**	*10.07*	20.13	9.40	37.58	*1.34*	*4.70*	*0.67*	*3.36*	*0.67*
	S. Best (%)	**61.07**	*0.00*	0.00	0.00	*4.03*	*0.00*	*0.00*	*0.00*	*0.00*	*0.00*

Table 4.1: Quantitative results: Fraction of times a method achieves the best and strictly best cut value on the K-NN graphs used in selecting the graph parameters. Only our method can optimize all the six balanced cut criteria shown here. Results for methods that do not directly optimize the cut criteria are shown in *Italic*.

approaches did not obtain the best cut value for any of the datasets. The method NMFR failed to return a clustering result for MNIST dataset.

For 4 out of the 11 datasets, the best cut also corresponds to the best clustering performance. In case of Vertebral, 20News, and WebKb4 the best cuts actually result in high errors. However, we see in our next experiments that integrating ground-truth label information or pairwise constraints helps in these cases to improve the clustering performance significantly.

Transductive Setting: In the next experiment, we evaluate our method against MTV in a transductive setting. As in [19], we randomly sample either one label or a fixed percentage of labels per class from the ground truth. For our method, we used 6 out of the 12 initializations for first computing the unconstrained solution; then we started our transductive method from the unconstrained solution as well as the remaining 6 initializations. We report in Table 4.3 clustering errors, cuts (RCC-asym) as well as the runtimes for both methods for different choices of labels. Again our method achieved the best cut value in 41 out of the 44 problem instances. In 29 out of those 41 cases, the best cut also corresponds to the smallest clustering error. MTV achieved best cuts in only 6 out of the 44 instances. Note that in some cases MTV seems to fail completely (Iris and 4Moons for one label per class case).

Overall, the performance in terms of clustering error generally improved with increased label information. For some datasets these improvements are significantly high. For example, for 20News our method reduced the error by more than half with 10% label information; also for Vertebral and WebKb4 the improvements achieved by our method are high.

		Iris	Wine	Vertebral	Ecoli	4Moons	WebKb4	OptDigits	USPS	PenDigits	20News	MNIST
1Spec	Err(%)	23.33	37.64	50.00	19.35	36.33	60.46	11.30	20.09	17.59	84.21	11.82
	BCut	**1.495**	6.417	**1.890**	2.550	0.634	**1.056**	0.386	0.822	0.081	0.966	0.471
	Time(s)	7	10	14	26	277	242	1056	1588	2197	6220	13068
Spec	Err(%)	**22.00**	20.22	48.71	14.88	31.45	60.27	7.81	21.06	11.27	79.17	22.67
	BCut	1.783	5.820	1.950	2.759	0.917	1.467	0.442	0.8723	0.141	1.187	0.702
	Time(s)	0.04	0.046	0.062	0.099	0.684	1.421	1.137	2.37	2.941	21.864	35.547
PNMF	Err(%)	22.67	27.53	50.00	16.37	35.23	60.94	10.37	24.07	17.93	66.00	12.80
	BCut	1.508	4.916	2.250	2.652	0.737	3.520	0.548	1.180	0.415	2.924	0.934
	Time(s)	1	1	1	2	10	20	137	133	405	981	1136
NSC	Err(%)	23.33	17.98	50.00	14.88	32.05	59.49	8.24	20.53	19.81	78.86	21.27
	BCut	1.518	5.140	2.046	2.754	0.933	3.566	0.482	0.850	0.101	2.233	0.688
	Time(s)	4	4	5	14	21	62	384	534	484	1722	3433
ONMF	Err(%)	23.33	28.09	50.65	16.07	35.35	60.94	10.37	24.14	22.82	69.02	27.27
	BCut	1.518	4.881	2.371	2.633	0.725	3.621	0.548	1.183	0.548	3.058	1.575
	Time(s)	1	1	2	3	10	10	46	120	115	764	1401
LSD	Err(%)	23.33	17.98	39.03	18.45	35.68	47.93	8.42	22.68	13.90	67.81	24.49
	BCut	1.518	5.399	2.557	2.523	0.782	2.082	0.483	0.918	0.188	2.056	0.959
	Time(s)	7	6	11	27	328	267	1618	1031	513	697	3564
NMFR	Err(%)	**22.00**	11.24	38.06	22.92	36.33	40.73	2.08	22.17	13.13	**39.97**	fail
	BCut	1.627	4.318	2.713	2.556	0.840	1.467	0.369	0.992	0.240	1.241	-
	Time(s)	2	2	8	7	8589	9514	20681	9461	7284	19991	-
Graclus	Err(%)	22.67	18.54	**34.52**	22.02	7.72	48.40	4.11	15.13	20.55	72.18	3.77
	BCut	1.508	5.556	2.433	2.500	0.774	2.346	0.374	0.940	0.193	3.291	0.458
	Time(s)	**0.019**	**0.02**	**0.022**	**0.022**	**0.126**	**0.09**	**0.104**	**0.162**	**0.134**	**0.265**	**1.123**
MTV	Err(%)	23.33	8.43	49.68	16.37	**0.45**	**39.97**	**1.67**	19.75	**10.93**	60.69	2.43
	BCut	1.534	4.293	**1.890**	2.414	0.589	1.581	**0.350**	0.815	0.092	1.431	0.440
	Time(s)	28	3	21	51	68	37	227	1070	584	4158	6846
Ours	Err(%)	23.33	**7.87**	50.00	**14.58**	0.47	60.46	1.71	**12.15**	19.95	79.64	**2.35**
	BCut	**1.495**	**4.166**	**1.890**	**2.373**	**0.589**	**1.056**	**0.350**	**0.779**	**0.079**	**0.894**	**0.439**
	Time(s)	5	11	8	24	374	495	3855	2446	6117	14621	45895

Table 4.2: Clustering errors, balanced cuts (RCC-asym) as well as the runtimes (seconds) of different methods evaluated on the K-NN graphs built using the parameter values $s = 1$, $K = 15$. The best results are shown in the **bold face.**

Labels			Iris	Wine	Vertebral	Ecoli	4Moons	WebKb4	OptDigits	USPS	PenDigits	20news	MNIST
1	MTV	Err(%)	30.00	14.61	**44.90**	15.18	19.49	46.38	**1.70**	**6.88**	12.57	**50.21**	2.45
		BCut	3.408	4.432	2.256	2.420	0.658	1.527	0.350	0.797	0.107	1.258	0.439
		Time(s)	**7**	**1**	**2**	**10**	**18**	**18**	**78**	**304**	**282**	**1385**	**2420**
	Ours	Err(%)	**22.60**	**7.02**	48.68	**14.73**	**7.62**	**45.69**	1.70	12.60	**12.12**	55.50	**2.37**
		BCut	**1.558**	**4.202**	**2.234**	**2.404**	**0.599**	**1.460**	**0.350**	**0.786**	**0.100**	**1.146**	**0.439**
		Time(s)	24	40	53	144	1938	2411	7321	15360	21029	74751	140074
1%	MTV	Err(%)	30.27	10.62	**42.23**	15.48	**0.45**	44.65	**1.69**	**5.34**	**8.33**	39.59	2.33
		BCut	3.381	4.420	2.291	2.440	**0.590**	1.566	**0.352**	0.788	0.122	1.199	0.443
		Time(s)	**7**	**1**	**3**	**8**	**21**	**19**	**61**	**251**	**191**	**1466**	**1549**
	Ours	Err(%)	**22.60**	**7.25**	46.61	**14.73**	**0.45**	**43.11**	**1.69**	5.42	10.13	**37.44**	**2.32**
		BCut	**1.558**	**4.185**	**2.253**	**2.407**	**0.590**	**1.490**	**0.352**	**0.788**	**0.119**	**1.145**	**0.442**
		Time(s)	25	43	52	139	1870	1949	7895	15940	22619	67606	130321
5%	MTV	Err(%)	20.67	8.65	38.06	14.61	**0.43**	41.59	**1.60**	**4.88**	**2.32**	32.38	2.16
		BCut	**1.692**	4.283	**2.691**	2.486	**0.590**	1.778	0.362	0.805	0.175	1.271	0.454
		Time(s)	**1**	**1**	**1**	**6**	**12**	**14**	**43**	**222**	**120**	**890**	**1136**
	Ours	Err(%)	**20.60**	**7.30**	**37.29**	**14.35**	**0.43**	**40.74**	1.60	4.94	2.72	**30.76**	**2.14**
		BCut	1.694	**4.199**	2.707	**2.483**	**0.590**	**1.681**	**0.362**	**0.804**	**0.174**	**1.224**	**0.454**
		Time(s)	19	39	57	130	888	1593	12564	14132	13511	61751	137905
10%	MTV	Err(%)	**19.33**	8.20	36.32	14.05	0.41	39.99	**1.48**	**4.38**	1.59	27.56	1.96
		BCut	**1.856**	4.308	3.156	2.695	0.592	2.036	0.371	0.826	0.190	1.360	0.466
		Time(s)	**1**	**1**	**1**	**7**	**10**	**11**	**32**	**203**	**100**	**792**	**899**
	Ours	Err(%)	19.53	**6.69**	**34.87**	**13.69**	**0.41**	**37.24**	1.49	4.40	**1.59**	**26.77**	**1.94**
		BCut	1.856	**4.214**	**3.154**	**2.651**	**0.592**	**1.935**	**0.371**	**0.826**	**0.190**	**1.323**	**0.466**
		Time(s)	19	36	57	98	684	1467	13850	15087	12530	49589	115857

Table 4.3: Transductive setting: Clustering errors, balanced cuts (RCC-asym) as well as the runtimes (seconds) of MTV and our method. The Labels column shows the number of labels available per class. The best results are shown in the **bold face.**

4.5.2 Constrained clustering

Here we evaluate our constrained clustering method against the following approaches designed for the multi-class setting: spectral learning (SL) [64], constrained clustering via spectral regularization (CCSR) [79] (see Section 3.1 for details of these methods). For reference, we also report the results of our two-class constrained clustering method (COSC) presented in Chapter 3. We use recursive bi-partitioning to compute k-clustering from COSC. Note that unlike must-link constraints which can be directly integrated even in the multi-class setting (see Section 3.3), cannot-link constraints pose difficulty for the recursive bi-partitioning procedure. The main issue here that it is not clear which cannot-link constraints should be considered for the two-way split in each step.

To address this issue for COSC, we use the soft-version of our formulation (see Lemma 3.1), which allows us to find a solution with at most l violations. We derive this number l for each two-way split assuming the following simple uniform model of the data and constraints. We assume that all classes have equal size and there is an equal number of cannot link constraints between all pairs of classes; i.e., if there are N cannot-link constraints and k classes then on average every pair of classes has $\frac{N}{k(k-1)/2}$ constraints. Under this assumption, among all the possible two-way splits of the given graph, the one that separates a single class from the rest will violate the maximum number of cannot-link constraints. The expected value of the maximum number of violations for the first two-way split is $\frac{(k-1)(k-2)/2}{k(k-1)/2}N$. One can similarly estimate the maximum number of violation for all the successive two-way splits. For each successive split after the first bi-partitioning, N is known (it is the number of cannot-link constraints present on the subgraph being split), while k can again be derived (assuming the uniform model) as $\frac{k}{n}\tilde{n}$ where $\tilde{n}$ is the size of the subgraph being split.

Constraint generation: We use the same procedure described in Chapter 3 to generate pairwise constraints from the class labels Y. We first fix the number of pairwise constraints n_C and then randomly sample n_C pairs. For each pair, we introduce either a cannot-link or a must-link constraint based on the true labels of the sampled pair. We evaluate the methods for various values of n_C. Similar to the two-class constrained clustering case, we maintain the property that the constraint sets are nested; i.e., if S_1 and S_2 are the constraints generated for a given dataset for two different values of n_C, then $S_1 \subset S_2$ whenever $|S_1| < |S_2|$.

In order to compare against the existing work, in our experiments we use normalized cut as the clustering objective:

$$\mathrm{NCut}(C_1, \ldots, C_k) = \sum_{l=1}^{k} \frac{\mathrm{cut}(C_l, \overline{C_l})}{\mathrm{vol}(C_l)}.$$

Starting with $\gamma = 0$, we ran our method for a sequence of increasing values of γ until all constraints are satisfied or γ reached an upper bound (see the discussion at the end of Section 4.4.3). Similar to the unconstrained case, we used 5 random initializations and 7 initializations based on the spectral clustering solution for the case $\gamma = 0$. For the subsequent iterations of γ, we used the results obtained from the previous value of γ as initializations.

Our evaluation is based on the following three criteria: clustering error (see (4.27)), normalized cut and fraction of constraints violated. The results averaged over 10 trials are shown in Tables 4.4 and 4.5. The runtimes for these experiments are given in Table 4.7. We first note that our method returned solutions satisfying all the given constraints for all datasets except for Vertebral and WebKb4, where the fraction of constraints violated is close to zero as well; see the third column of Tables 4.4 and 4.5. Note that we have not used any initializations based on a consistent partition for our method. In spite of that our method is able to produce consistent solutions most of the time. The fraction of constraints violated by the solutions of other methods is high for most of the datasets. In particular both SL and CCSR failed to find a consistent solution in all cases whereas COSC succeeded a few times.

Our method produced much better cuts than CCSR in many cases although our method solves a constrained problem and yielded results satisfying almost all constraints. The cuts found by CCSR are much worse than those found by other methods for the last five datasets. So, we provide additional plots in Table 4.6 highlighting the differences among the cut values found by SL, COSC and our method for these datasets. Note that for OptDigits and MNIST datasets, our method is able to find solutions satisfying all constraints and yet achieve much better cut values than the other methods. For OptDigits, COSC yielded much higher cuts and hence not included in the additional plot. Our method failed considerably for the USPS and PenDigits datasets. Since the constraint sets are nested, the optimal normalized cut values are monotonically increasing with the number of constraints. However, for USPS and PenDigits, our method achieved much higher cuts for the intermediate cases (800, 1600, 3200) than the cut obtained for the last case (6400). Our optimization algorithm seems to have stuck at suboptimal solution while enforcing the cannot-link constraints for the intermediate cases; recall that must-link constraints are directly integrated as explained in Section 3.3. However for the last case (6400), the presence of more must-link constraints might have eliminated such suboptimal solutions and hence our method achieved better cut. One needs more number of initializations to improve the cuts for the intermediate cases.

In terms of clustering error, shown on the first column of Tables 4.4 and 4.5, our method consistently outperforms the other methods. Only for the USPS dataset, the recursive bi-partitioning variant of our two-class constrained clustering method (COSC) performed much better than all the methods (in terms of clustering error) including our direct multi-class method. For other datasets the recursive procedure failed considerably; see results for OptDigits and WebKb4. Note that similar to the transductive case, incorporating prior knowledge improves the clustering performance. These improvements are significant for several datasets; see the results for Iris, WebKb4, 20News datasets. We note here that the clustering errors reported here are not directly comparable with those given in the transductive experiments because we use different balanced cut criteria in these experiments.

In our next experiment, we evaluate the clustering performance when the pairwise constraints are generated from the class labels of a given set of points. Recall that in the previous case, we first sampled a given number of pairs and then decided the constraint type based on the class labels of the pairs. In contrast, here we first sample a fixed number of points and then deduce all possible pairwise con-

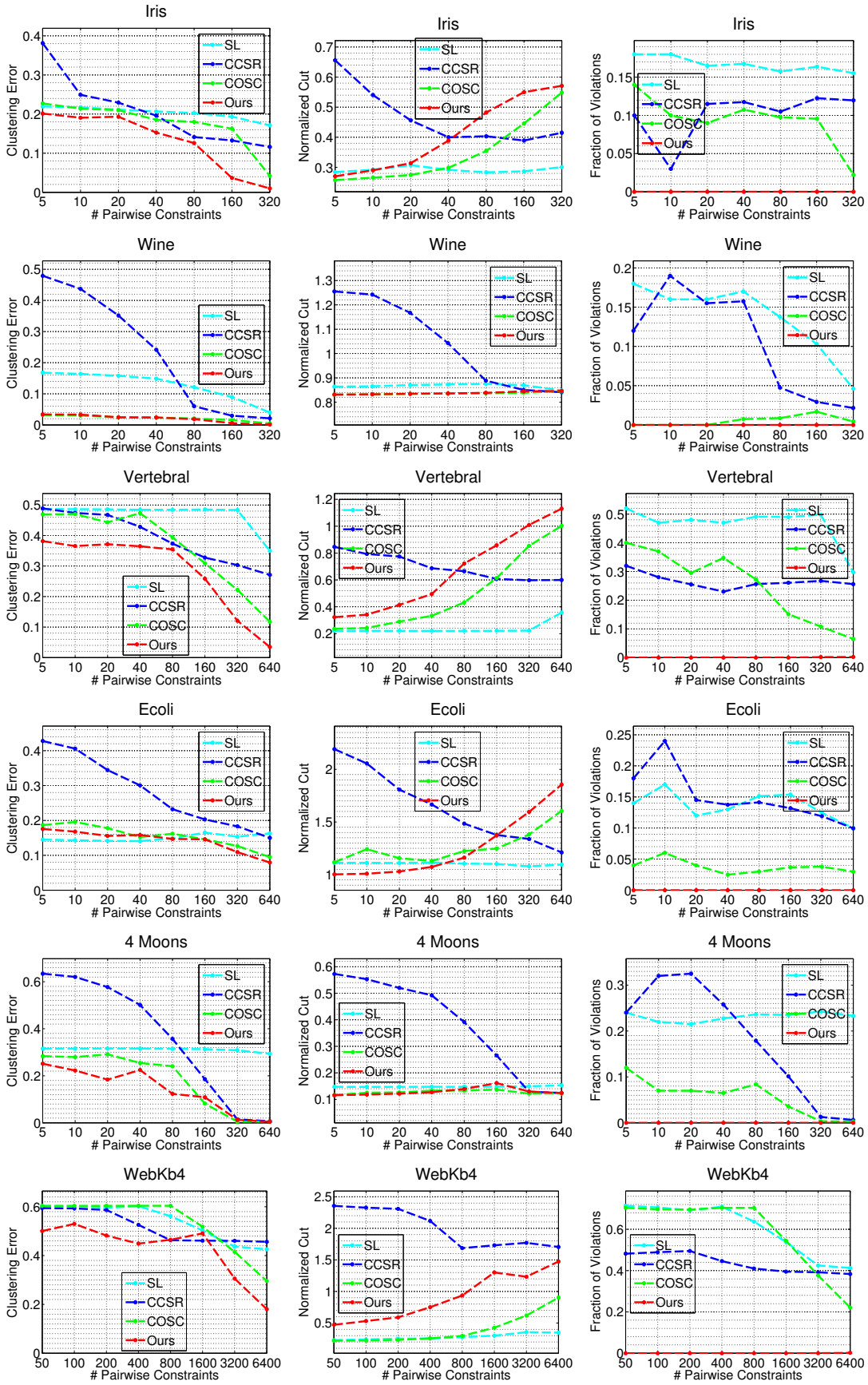


Table 4.4: Results for **multi-class constrained clustering**: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

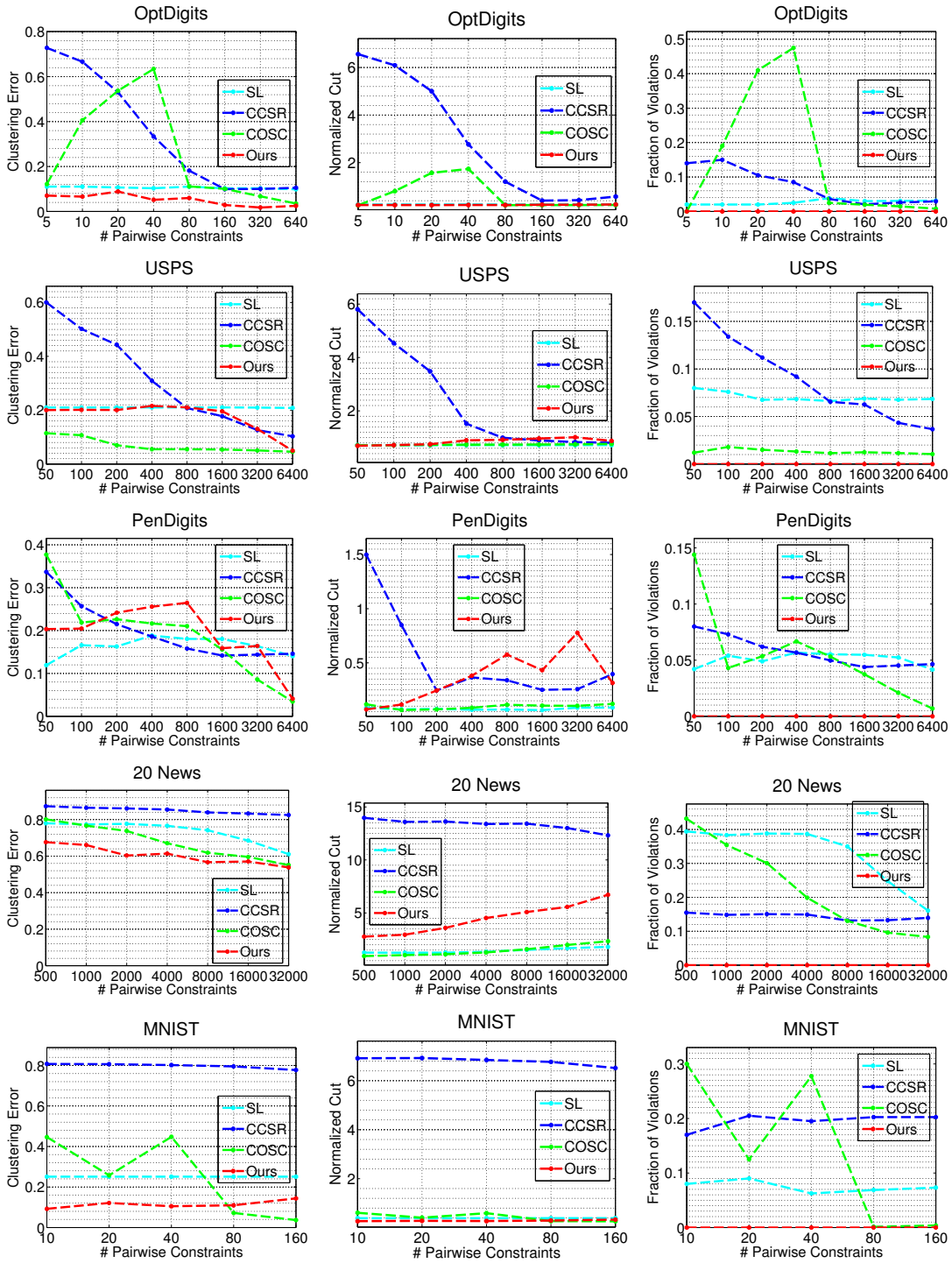


Table 4.5: Results for **multi-class constrained clustering**: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

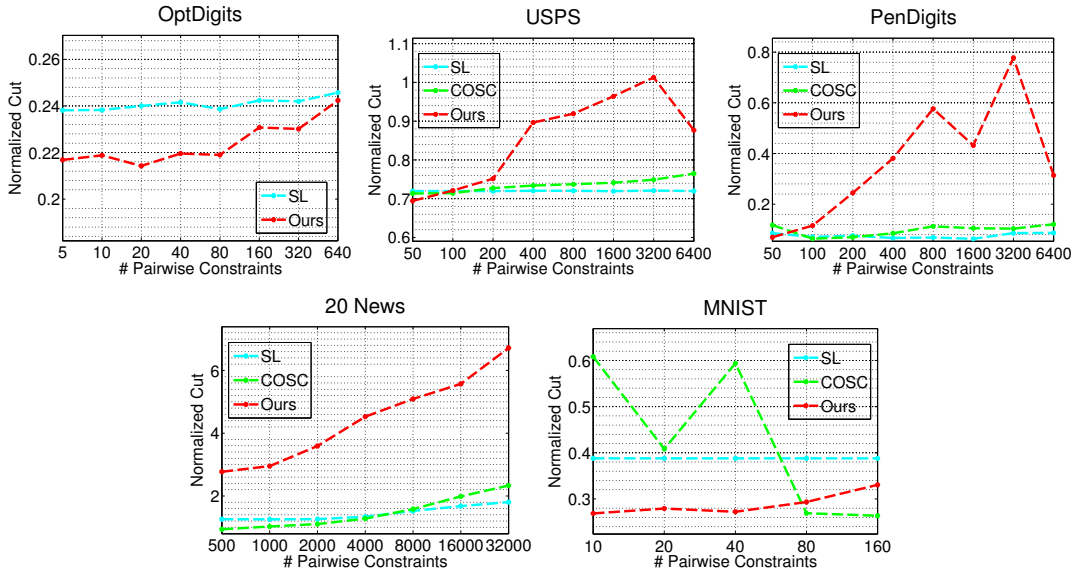


Table 4.6: Results for **multi-class constrained clustering**: Additional plots highlighting the differences among the normalized cut values found by SL, COSC and our method.

straints from the class labels of the sampled points. Note that one can generate $\binom{p}{2}$ constraints from p labeled points. Note that the former way of sampling results in constraints that span more number of points whereas in the latter case the constraint information is concentrated on a few points.

In order to compare the clustering performance to the previous case, the number of sampled points are chosen in such a way that the total number of pairwise constraints generated from them are similar to the number of constraints used in the previous experiment. The results for this experiment are shown in Tables 4.8, 4.9 and 4.11. For reference, we also report the results obtained by our transductive method which is given the same label information that was used in generating the pairwise constraints. First notice that the performance of all methods is similar to that of the previous experiment. The cut values obtained by our constrained clustering method are closer to those obtained from the transductive method. For clarity, we provide an additional plots in Table 4.10 highlighting the differences among the cut values found by SL, COSC and our method for the last five datasets. The clustering error typically reduces with the increased number of constraints as in the previous case. However, for many datasets, the amount of reduction in error is much smaller than that obtained in the previous case although both experiments use similar number of pairwise constraints. This suggests that in practice it is beneficial to obtain pairwise constraint information from several different points than deriving all possible constraints from a small number of points.

Special case $k = 2$: In our final experiment we evaluate our multi-class formulation for constrained clustering for the special case $k = 2$ against the one proposed in Chapter 3. We ran our method on the same datasets and constraints used in Chapter 3. The results are reported in Tables 4.12, 4.13 and 4.14. Our two-class formulation (COSC) performs better than our multi-class formulation across the three different evaluation criteria. Although the multi-class formulation was able

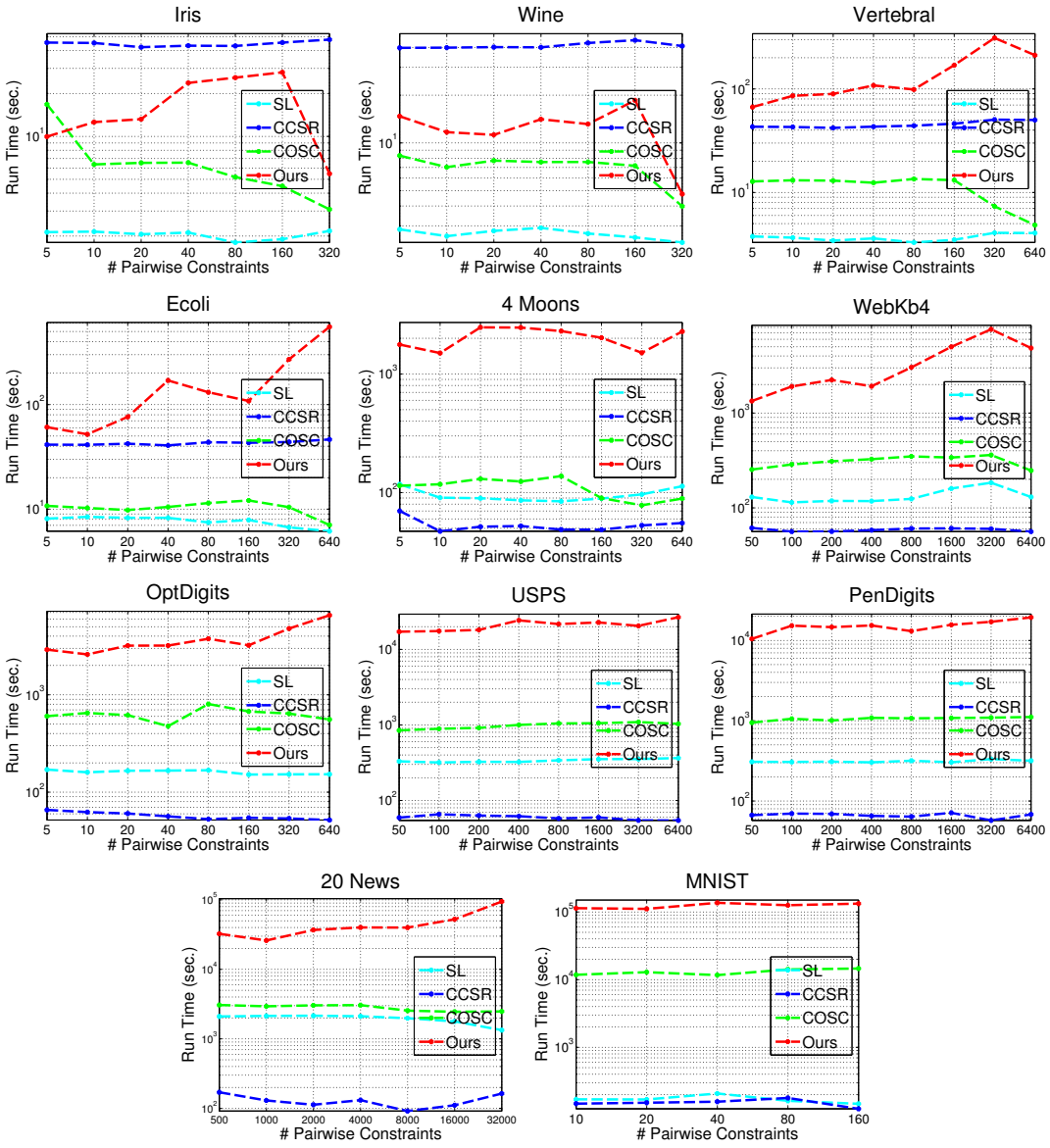


Table 4.7: Results for **multi-class constrained clustering**: Runtimes versus number of constraints.

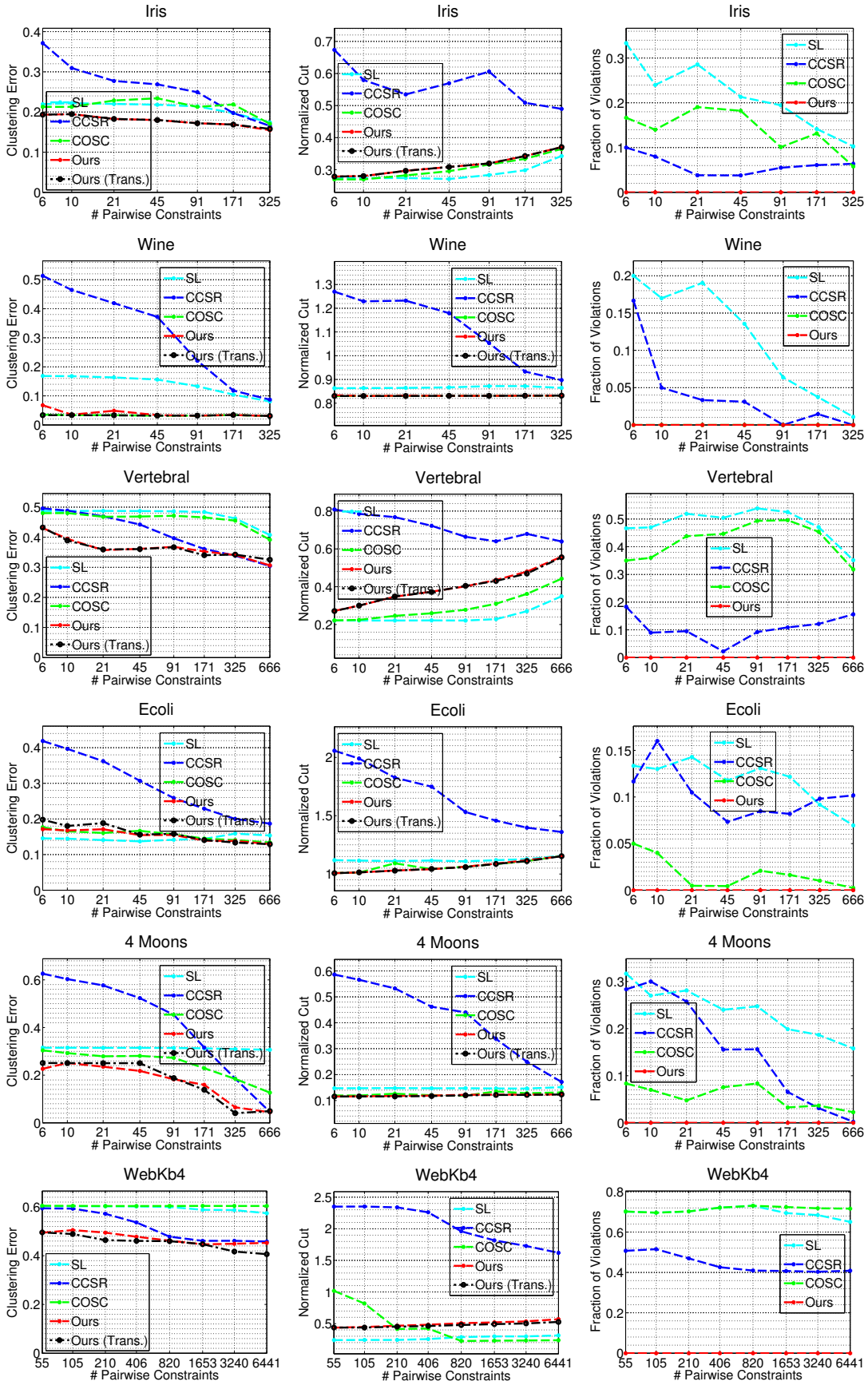


Table 4.8: Results for **multi-class constrained clustering** where pairwise constraints are generated from a transductive setting. Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

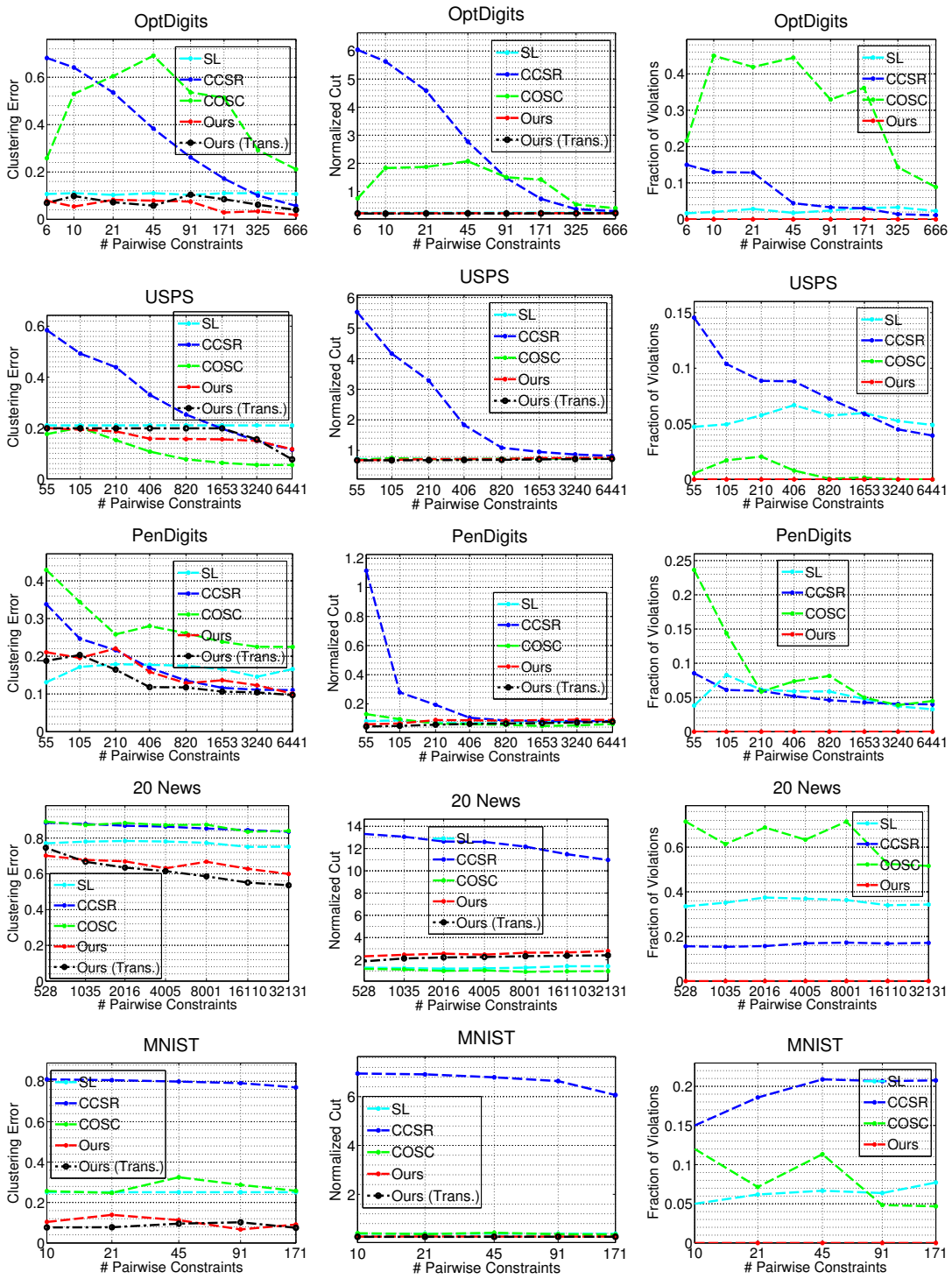


Table 4.9: Results for **multi-class constrained clustering** where pairwise constraints are generated from a transductive setting. Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

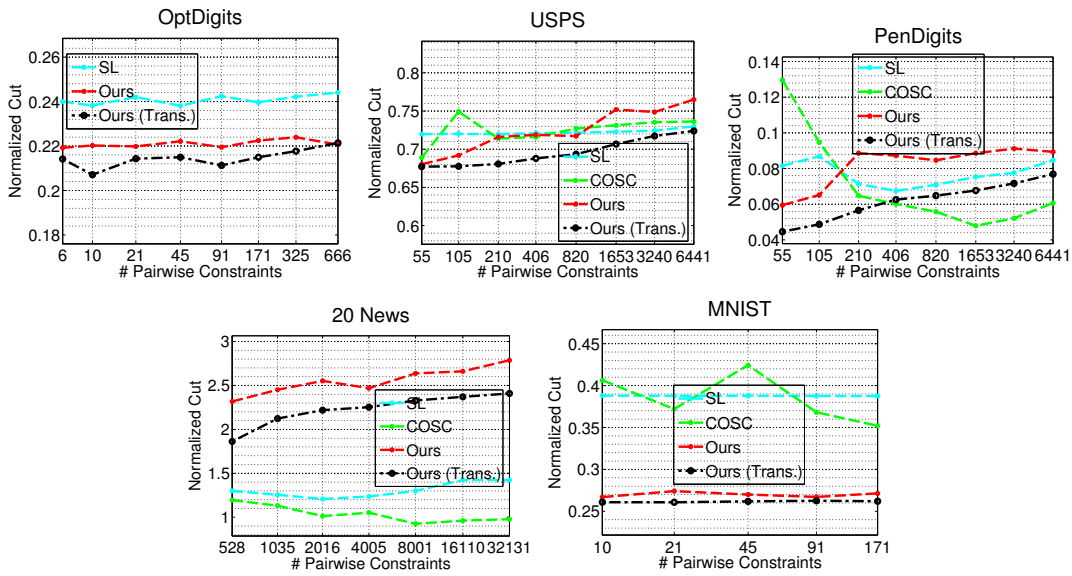


Table 4.10: Results for **multi-class constrained clustering** where pairwise constraints are generated from a transductive setting. Additional plots highlighting the differences among the normalized cut values found by SL, COSC and our method.

to obtain cuts closer to those obtained by COSC in some of the datasets (e.g., Breast Cancer, WDBC and Heart), it fails considerably in case of USPS and Spam datasets. Moreover, unlike COSC, the multi-class formulation sometimes failed to satisfy the given constraints. Although the multi-class formulation is better compared to COSC (in the sense that it does not introduce any bias), its optimization turns out to be much more difficult than minimizing the ratio formulation of COSC. It well-known in operations research community that minimization of sum-of-ratios is a much harder problem than the optimization of a single ratio [104]. Hence we recommend using COSC for constrained clustering when $k = 2$.

Scaling our method to large datasets: Our multi-class formulation, although achieves the best performance, is computationally expensive. A possible future direction would explore applying the multi-level strategy used in Graclus. In particular, one generates a sequence of coarser graphs from the given graph respecting the cannot-link constraints and applies our method first on the coarsest graph. Since the coarsest graph is typically small, one can quickly compute a good solution from our method possibly starting from more initializations. Then the resulting solution is used as an initialization for refining the partition on the finer graph. Since our methods (both unconstrained as well as constrained versions) satisfy monotonic descent property one can monotonically improve the solution obtained from the coarsest graph. Since we use only one initialization for the subsequent iterations on the finer graphs we expect the resulting method to scale well to large datasets. Moreover, unlike Graclus, the resulting method would still enjoy the guarantees shown here such as satisfaction of constraints.

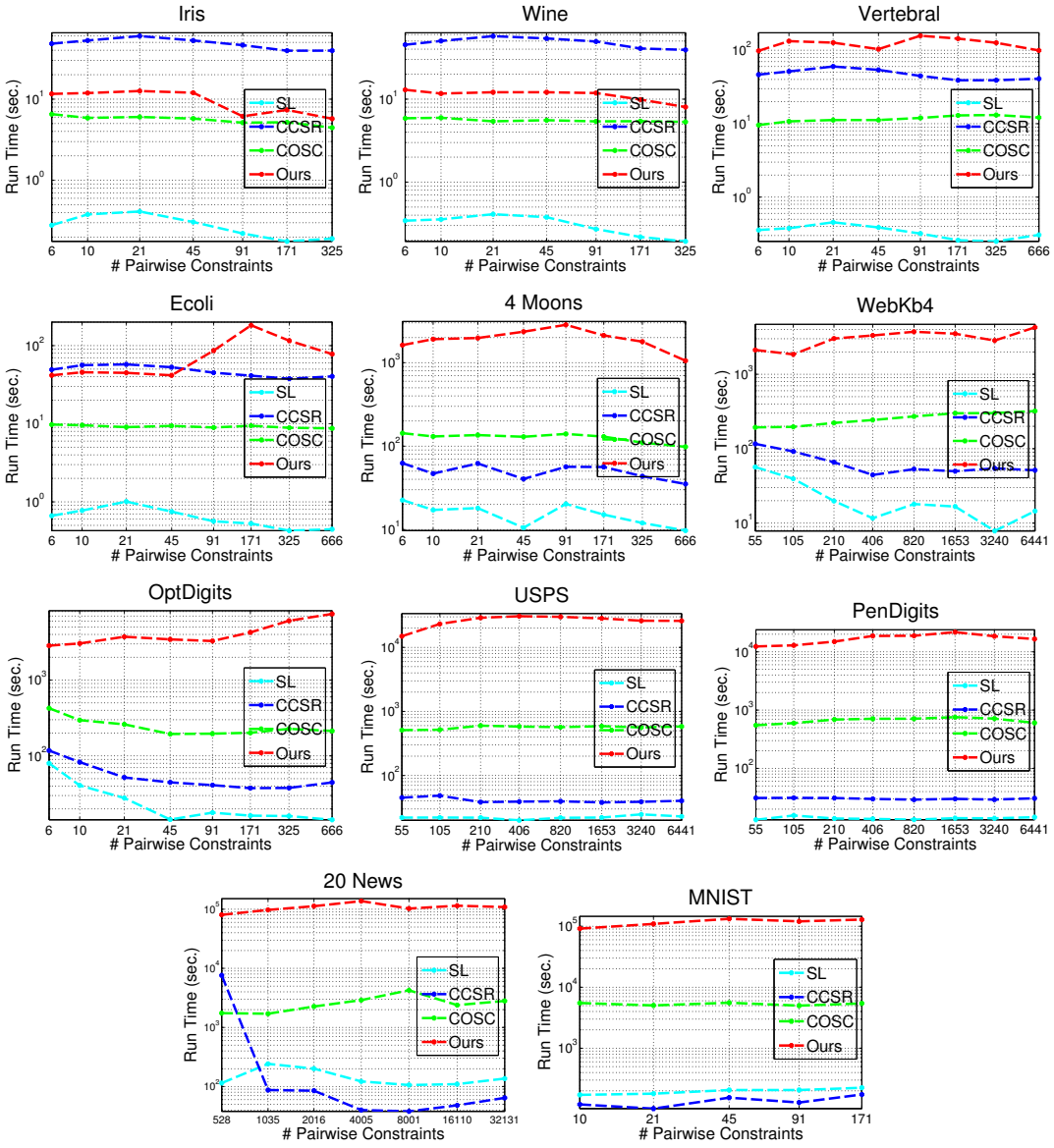


Table 4.11: Results for **multi-class constrained clustering** where pairwise constraints are generated from a transductive setting: Runtimes versus number of constraints.

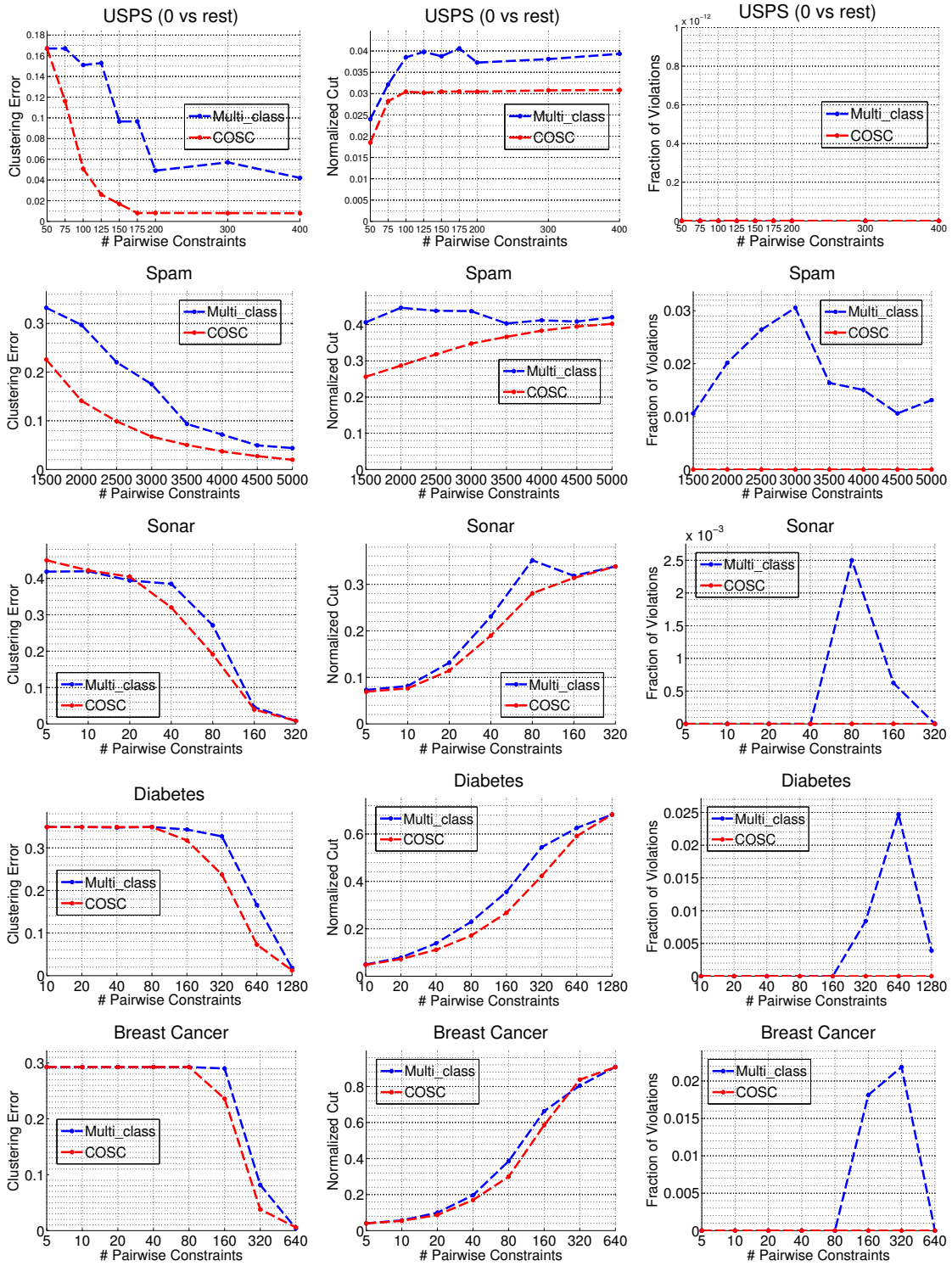


Table 4.12: Multi-class formulation versus COSC for $k = 2$: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

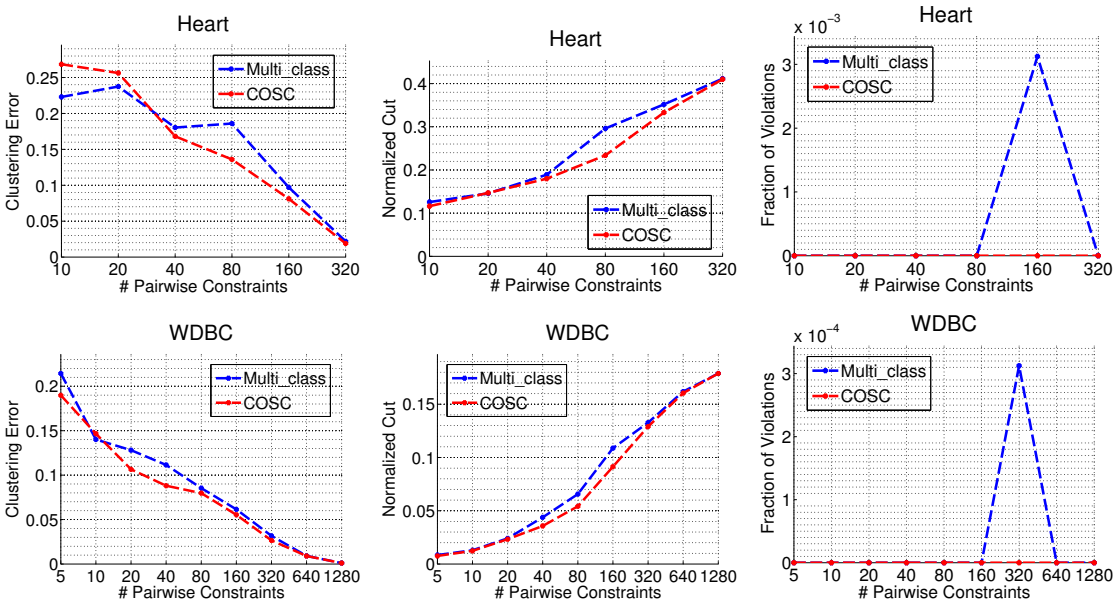


Table 4.13: Multi-class formulation versus COSC for $k = 2$: Left: clustering error versus number of constraints, Middle: normalized cut versus number of constraints, Right: fraction of violated constraints versus number of constraints.

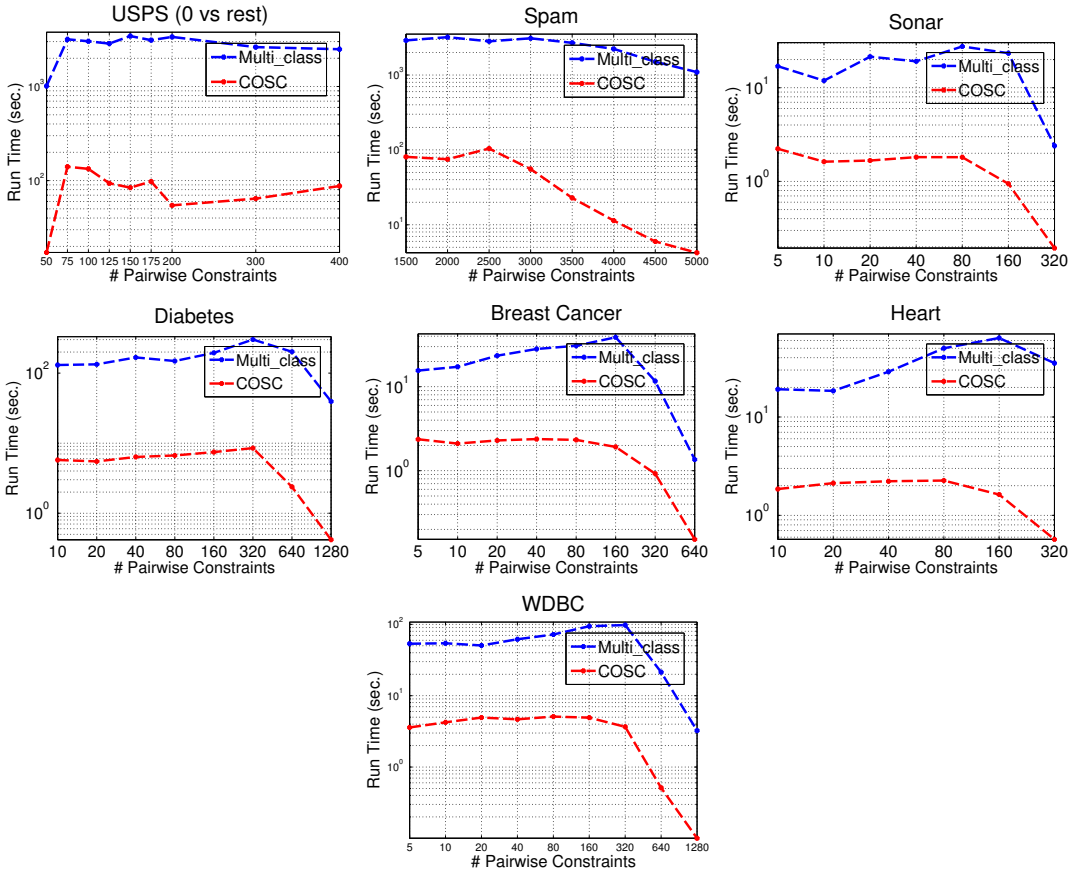


Table 4.14: Multi-class formulation versus COSC for $k = 2$: Runtimes versus number of constraints.

4.6 Conclusions

In this chapter, we presented a generic method for directly minimizing the balanced k-cut problem based on a new continuous relaxation. Apart from ratio/normalized cut, our method can also handle new application-specific balancing functions. We also developed constrained clustering method for the multi-class setting that guarantees satisfaction of constraints under a mild assumption that any consistent partition can be found efficiently. Finally, the monotonic descent algorithm proposed for the difficult sum-of-ratios problem is another key contribution that is of independent interest.

Chapter 5

Community detection via densest subgraphs

In this chapter we consider the one-class problem where the goal is to extract a a densely connected subset in a given graph. The problem of finding dense regions in graphs arises in several applications e.g., in bioinformatics [106, 111, 12, 60, 102] and network analysis [72, 40, 47]. In bioinformatics, one can model the problem of identifying protein families or molecular complexes as the problem of finding densely connected components in a protein-protein interaction network [12]. More generally, community detection, an important tool in understanding the structure of complex networks, can be modeled as the problem of finding dense regions in a graph.

A natural graph-theoretic measure for well-connectedness is density defined as the ratio of the total weight of the edges in the subgraph induced by the given subset and the size of the subset. This criteria leads to an important and well-studied graph-theoretic problem known as densest subgraph problem which can be solved in polynomial time [49]. Often in applications one may have to enforce size restrictions since the densest subset found maybe too small or too large. It has been shown that introducing prior restriction on the size of the subset makes the problem NP-hard [45, 4, 68]. Other possible forms of prior in this setting are based on the locality of the subset being extracted [102]: (i) finding a subset around a given source or seed set (ii) enforcing locality based on an independent source beyond the input graph; e.g., requiring every pair of vertices in the extracted subset to be with in a given distance where the distance function can be defined using an entirely different network.

Here we consider a generic version of the densest subgraph problem to allow for more modeling freedom. Similarly to the two-class clustering problem, we first derive an exact continuous relaxation for the generalized densest subgraph problem proposed. Then we present an efficient method for solving the resulting continuous relaxation. The main feature of our method is the guarantee that the obtained solution satisfies all the given constraints. We also discuss an application problem arising in social network analysis. The work presented in this chapter is published in [95].

Let us introduce the notation before formally defining the generalized densest subgraph problem. Let $G(V, W)$ be an undirected graph where $V = \{1, \ldots, n\}$ is the n-element vertex set, $W \in \mathbb{R}_+^{n \times n}$ is the symmetric weight matrix. We denote

the entries of W by $w_{ij},\ i,j = 1,\dots n$. We allow the vertices to have non-negative weights denoted by $b \in \mathbb{R}^n_+$. We denote by $\mathbf{1}_n$ the $n \times 1$ vector of all ones and by $\mathbf{1}_C$ the indicator vector on a set C whose i^{th} entry is 1 if $i \in C$ and 0 otherwise. Given $g \in \mathbb{R}^n_{++}$, the (generalized) volume of a set $C \subseteq V$ is defined as $\text{vol}_g(C) = \sum_{i \in C} g_i$, which specializes to cardinality if $g_i = 1,\ \forall i \in V$ and to the well-known volume function used in normalized cut when $g_i = d_i,\ \forall i \in V$, where $d_i = \sum_{j=1}^n w_{ij}$ is the degree of vertex i.

The (generalized) density of a subset $C \subseteq V$ is defined as

$$\text{density}(C) := \frac{\text{assoc}(C)}{\text{vol}_g(C)} = \frac{\sum_{i,j \in C} w_{ij}}{\sum_{i \in C} g_i} \tag{5.1}$$

We recover the classical definition of density via $g_i = 1, \forall i \in V$. We use the relation, $\text{assoc}(C) = \text{vol}_d(C) - \text{cut}(C, V \backslash C)$, where d_i is the degree of vertex i and $\text{cut}(A,B) := \sum_{i \in A, j \in B} w_{ij}$.

5.1 Generalized densest subgraph problem

We consider the following generic version of the densest subgraph problem

$$\begin{aligned} \max_{C \subseteq V} \ & \frac{\text{assoc}(C)}{\text{vol}_g(C)} \\ \text{subject to} : : \ & U \subseteq C \\ & \kappa_j \leq \text{vol}_{M_j}(C) \leq \iota_j, \quad \forall j \in \{1,\dots,p\} \\ & \text{dist}(u,v) \leq d_0, \quad \forall u,v \in C. \end{aligned} \tag{5.2}$$

Here U is the seed set and $M_j \in \mathbb{R}^n_+,\ j = 1,\dots,p$ denote vertex weights which allow us to model the "size" constraint in different ways. Note that our model allows all forms of size constraints: lower bound, upper bound and equality constraints. The distance constraint is modeled via a general non-negative, symmetric distance function dist; here dist need not satisfy the triangle inequality. Here $\kappa_j, \iota_j,\ j = 1,\dots,p$ and d_0 are problem-specific constants. Since U is required to be part of the solution, we can assume that $dist(u,v) \leq d_0, \forall u,v \in U$, otherwise the above problem is infeasible. The distance constraint also implies that any $u \in V$ for which $\text{dist}(u,s) > d_0$, for some $s \in U$, cannot be a part of the solution. Thus, we again assume without loss of generality that there is no such $u \in V$; otherwise such vertices can be eliminated without changing the solution of problem (5.2).

The classical densest subgraph problem studied in the literature, a special case of this problem, is given by

$$\max_{C \subseteq V} \frac{\text{assoc}(C)}{\text{vol}_g(C)}, \tag{5.3}$$

where $g_i = 1,\ \forall i \in V$. This problem can be optimally solved in polynomial time [49]. The densest-k-subgraph problem, which requires the solution to contain exactly k vertices is shown to be an NP-hard problem [45]. In our generalized setting, this problem corresponds to the special case (with distance constraint removed): $U = \emptyset, p = 1, M_1 = \mathbf{1}_n$ and $\kappa_1 = \iota_1 = k$. This problem has been shown not

to admit a polynomial time approximation scheme [67]. This means that there is no polynomial time algorithm that gives a solution within a factor of $1-\epsilon$ of the optimal solution for arbitrary small constant $\epsilon > 0$. The best known approximation algorithm has an approximation ratio of $O(V^{\delta})$, for some $\delta < \frac{1}{3}$ [45].

Recently, it has been shown that the densest subgraph problem with an upper bound on the size, i.e., $|C| \leq k$, is as hard as the densest-k-subgraph problem [68]. However, it is also shown in [68] that the densest subgraph problem with a lower bound constraint, i.e., $|C| \geq k$, is still NP-hard but has a 2-approximation algorithm. Both these problems are special cases of the model considered in (5.2).

5.1.1 Relation to local clustering

Local clustering is a related area where one is interested in finding a cluster around a given seed set. Here we briefly mention the relationship between the generalized densest subgraph problem (5.2) and formulations considered in local clustering. Given a graph $G(V, W)$ and a seed set U, the goal in local clustering is to exact a subset around U minimizing the balanced cut

$$\begin{aligned} \min_{C \subseteq V} \ & \frac{\text{cut}(C, \overline{C})}{\hat{S}(C)} \\ \text{subject to} : \ & U \subseteq C \\ & \text{vol}(C) \leq k, \end{aligned} \tag{5.4}$$

where $\hat{S}$ is a balancing function and $\overline{C} = V \backslash C$. This problem has attracted a lot of interest [110, 6, 5, 26, 7, 86, 54, 85]. Typical choices for $\hat{S}$ considered in the literature are $\text{vol}(C)\,\text{vol}(\overline{C})$ and $\min\{\text{vol}(C), \text{vol}(\overline{C})\}$ leading to local normalized cut and local normalized Cheeger cut respectively.

We note that for the choice, $\hat{S}(C) = \min\{\text{vol}(C), \text{vol}(\overline{C})\}$ and $k \leq \frac{1}{2}\,\text{vol}(V)$, the local clustering problem reduces to an instance of the generalized densest subgraph problem (5.2). To see this, note that when $k \leq \frac{1}{2}\,\text{vol}(V)$, the balancing function can be rewritten as $\text{vol}(C)$. Using the relation

$$\text{cut}(C, \overline{C}) = \text{vol}(C) - \text{assoc}(C),$$

one rewrites the objective of (5.4) as

$$1 - \frac{\text{assoc}(C)}{\text{vol}(C)},$$

which leads to an instance of the generalized densest subgraph problem (5.2). Moreover, the method developed for solving (5.2) in this chapter can easily be modified to solve the generic version of the local clustering problem (5.4). For details we refer the reader to our joint work [22].

5.2 An equivalent unconstrained formulation

We first show that the seed constraint can be efficiently handled by directly integrating it into the objective. For this, note that any subset $C \subseteq V$ that contains

U can be written as $C = A \cup U$, for some $A \subseteq V'$ where $V' = V \backslash U$. Thus, we can reformulate the problem (5.2) directly on the subgraph $G'(V', W')$ where the elements of W' are given by $w'_{ij} = w_{ij},\ i, j \in V'$. We introduce the notation $m = |V'|$, and we assume without loss of generality that the first m entries of V are the ones in V'. Moreover we explicitly mention the graph while specifying the functions $\text{cut}_{G'}$ and $\text{assoc}_{G'}$ if there is any ambiguity; when there is no mention of the graph, it is assumed that the original graph G is being considered.

Lemma 5.1 *Let ϕ^* be the optimal value of the generalized densest subgraph problem* (5.2) *and let $G'(V', W')$ be the subgraph induced by the vertex set $V' = V \backslash U$ where U is the seed set. Then it holds that*

$$\begin{aligned} \phi^* = \max_{A \subseteq V'} \frac{\text{vol}_d(A) - \text{cut}_{G'}(A, V' \backslash A) + \text{assoc}(U) + \text{vol}_{d^U}(A)}{\text{vol}_g(A) + \text{vol}_g(U)} =: \hat{G}(A) \qquad (5.5) \\ \text{subject to}: k_j \leq \text{vol}_{M_j}(A) \leq l_j, \quad \forall j \in \{1, \ldots, p\} \\ \text{dist}(u, v) \leq d_0, \quad \forall u, v \in A, \\ \text{dist}(u, v) \leq d_0, \quad \forall u, v \in U \\ \text{dist}(u, v) \leq d_0, \quad \forall u \in U, \forall v \in A \end{aligned}$$

where $d_i^U = \sum_{j \in U} w_{ij},\ \forall i \in V'$ and $k_j = \kappa_j - \text{vol}_{M_j}(U),\ l_j = \iota_j - \text{vol}_{M_j}(U),\ j = 1, \ldots, p$. Moreover, A^* is an optimal solution of (5.5) *if and only if $C^* = A^* \cup U$ is an optimal solution of the problem* (5.2).

Proof: For any $C \subseteq V$ containing the seed set U, using the relation $C = A \cup U$, for some $A \subseteq V'$, we have

$$\begin{aligned} \text{assoc}(C) &= \text{assoc}(A) + \text{assoc}(U) + 2\ \text{cut}(A, U), \qquad (5.6) \\ &= \text{vol}_d(A) - \text{cut}_G(A, V \backslash A) + \text{assoc}(U) + 2\ \text{cut}_G(A, U) \\ &= \text{vol}_d(A) - \text{cut}_G(A, V' \backslash A) + \text{assoc}(U) + \text{cut}_G(A, U) \\ \text{vol}_g(C) &= \text{vol}_g(A) + \text{vol}_g(U) \end{aligned}$$

Also note that $\text{cut}_G(A, V' \backslash A) = \text{cut}_{G'}(A, V' \backslash A)$ for any $A \subseteq V'$. Moreover, we can write $\text{cut}_G(A, U) = \text{vol}_{d^U}(A)$, where $d_i^U = \sum_{j \in U} w_{ij}$ denotes the degree of vertex i in V', restricted to the subset U in the original graph G.

Since the vertices U are already included in the solution, we update the bounds on the skill as $k_j = \kappa_j - \text{vol}_{M_j}(U),\ l_j = \iota_j - \text{vol}_{M_j}(U),\ j = 1, \ldots, p$. The distance constraint is similarly updated. □

In the following we rewrite the maximization problem (5.5) as an equivalent minimization problem. So we define the penalty functions in such a way that they take a value of zero on feasible sets and strictly positive values on infeasible sets. As discussed in Section 5.1, given the set U, one can assume without loss of generality that the constraints $\text{dist}(u, v) \leq d_0, \forall u, v \in U$ and $\text{dist}(u, v) \leq d_0, \forall u \in U, \forall v \in A \subseteq V'$ are always satisfied. Thus we drop these constraints from our presentation below. Let $\hat{T}^{(1)}, \hat{T}^{(2)} : 2^{V'} \to \mathbb{R}_+$ denote the penalty functions for upper and lower bound

constraints respectively. We define them as

$$\begin{aligned}\hat{T}^{(1)}(A) &:= \sum_{j=1}^{p} \max\{0, \mathrm{vol}_{M_j}(A) - l_j\} \\ &= \sum_{j=1}^{p} \mathrm{vol}_{M_j}(A) - \min\{l_j, \mathrm{vol}_{M_j}(A)\} \quad \text{(upper bound constraints)} \qquad (5.7)\\ \hat{T}^{(2)}(A) &:= \sum_{j=1}^{p} \max\{0, k_j - \mathrm{vol}_{M_j}(A)\} \\ &= \sum_{j=1}^{p} k_j - \min\{k_j, \mathrm{vol}_{M_j}(A)\} \quad \text{(lower bound constraints)} \qquad (5.8)\end{aligned}$$

Note that the penalty function for upper bound constraints takes a value of zero whenever A satisfies all the constraints and for every constraint violated the penalty increases by the corresponding violation $\mathrm{vol}_{M_j}(A) - l_j$. Similarly for lower bound constraints, the penalty increases by $k_j - \mathrm{vol}_{M_j}(A)$ if A violates the constraint $\mathrm{vol}_{M_j}(A) \geq k_j$.

We define the penalty function for the distance constraint as

$$\hat{T}^{(3)}(A) := \sum_{u,v \in A} \max\{0,\ \mathrm{dist}(u,v) - d_0\},$$

so that for every pair of vertices in A that violate the constraint, a penalty of $\mathrm{dist}(u,v) - d_0$ is added. Note that we can rewrite this penalty function as the association of the set A on the graph $H(V', D)$ where $D_{uv} = \max\{0, \mathrm{dist}(u,v) - d_0\},\ \forall (u,v) \in V' \times V'$. That is

$$\hat{T}^{(3)}(A) = \mathrm{assoc}_H(A) = \mathrm{vol}_{\bar{d}}(A) - \mathrm{cut}_H(A, V' \backslash A) \quad \text{(distance constraint)}, \qquad (5.9)$$

where $\bar{d}$ is the degree of the vertices of the graph H defined as $\bar{d}_u = \sum_{v \in V'} D_{uv}$. Let us introduce $\hat{T} : 2^{V'} \to \mathbb{R}_+$ to capture the total amount of constraint violation,

$$\hat{T}(A) := \hat{T}^{(1)}(A) + \hat{T}^{(2)}(A) + \hat{T}^{(3)}(A). \qquad (5.10)$$

Note that the penalty function $\hat{T}$ is zero only when A satisfies all the constraints; otherwise it is strictly positive and is proportional to the amount of violation. We define θ to be the minimum value of the penalty function on infeasible sets

$$\theta = \min_{\hat{T}(A) > 0} \hat{T}(A). \qquad (5.11)$$

In general, depending on the constraints, obtaining θ can be hard. However, our method does not compute it explicitly as we solve a sequence of unconstrained problems for increasing values of penalty parameter.

Now we show that there exists an unconstrained problem equivalent to the constrained optimization problem (5.5).

Theorem 5.1 *Let $A_0 \subseteq V'$ be any subset satisfying the constraints of problem (5.5) such that $\hat{G}(A_0) > 0$, where $\hat{G}(A_0)$ is the objective value of problem* (5.5) *at A_0. If*

$\gamma > \frac{\mathrm{vol}_d(V)}{\theta \hat{G}(A_0)}$, where θ is the minimum value of infeasibility defined in (5.11)*, then the generalized densest subgraph problem (5.5) is equivalent to the unconstrained problem*

$$\min_{A \subseteq V'} \frac{\mathrm{vol}_g(A) + \mathrm{vol}_g(U) + \gamma \hat{T}(A)}{\mathrm{vol}_d(A) - \mathrm{cut}_{G'}(A, V' \backslash A) + \mathrm{assoc}(U) + \mathrm{vol}_{d^U}(A)} =: \hat{F}^{(\gamma)}(A) \tag{5.12}$$

in the sense that there is a one-to-one correspondence between optimal solutions of problems (5.5) *and* (5.12)*. Moreover if ϕ^* is the optimal value of problem* (5.5) *then $\phi^* = 1/\min_{A \subseteq V'} \hat{F}^{(\gamma)}(A)$.*

Proof: First note that maximizing $\hat{G}$ over the given feasible set is same as minimizing $\frac{1}{\hat{G}}$ over the same feasible set. For any feasible set $A \subseteq V'$ of problem (5.5), the objective value of (5.12) is equal to $F_\gamma(A) = \frac{1}{\hat{G}(A)}$, since the penalty term $\hat{T}(A)$ is zero. Thus if we show that any set $A \subseteq V'$ that does not satisfy all the constraints of problem (5.2) cannot be a solution of problem (5.12) then the equivalence follows. Suppose, for the sake of contradiction, that a minimizer A^* of (5.12) is infeasible for problem (5.5). Let $\hat{F}^{(\gamma)}$ denote the objective function of the problem (5.12) and $\hat{S}$ denote the denominator of $\hat{F}^{(\gamma)}$. Since $g_i > 0, \forall i$, we have for the given value of γ,

$$\hat{F}^{(\gamma)}(A^*) \geq \frac{\gamma \hat{T}(A^*)}{\max_{A \subseteq V'} \hat{S}(A)} \geq \frac{\gamma \theta}{\mathrm{vol}_d(V)} > \frac{1}{\hat{G}(A_0)} = \hat{F}^{(\gamma)}(A_0)$$

which is a contradiction because A^* is optimal for the problem (5.12). □

The following result connecting the generalized densest subgraph problem on the given graph G and an unconstrained ratio problem on the graph G' is immediate from Lemma 5.1 and Theorem 5.1

Theorem 5.2 *Let $G(V, W)$ be the given graph and let $G'(V', W')$ be the subgraph induced by the vertex set $V' = V \backslash U$ where U is the seed set. Let $A_0 \subseteq V$ be any subset satisfying the constraints of the generalized densest subgraph problem (5.2) such that* $\mathrm{density}(A_0) > 0$*. Moreover let ϕ^* be the optimal value of the problem* (5.2)*. If $\gamma > \frac{\mathrm{vol}_d(V)}{\theta \mathrm{density}(A_0)}$ where θ is the minimum value of infeasibility defined in* (5.11) *of any set $A \subseteq V'$, then*

$$\phi^* = \frac{1}{\min_{A \subseteq V'} \hat{F}^{(\gamma)}(A)}.$$

Moreover, A^* is an optimal solution of (5.12) *if and only if $C^* = A^* \cup U$ is an optimal solution of the problem* (5.2)*.*

5.3 Exact continuous relaxation of the generalized densest subgraph problem

In Theorem 3.2 we have derived exact relaxation result for the minimization of ratio of set functions where the set functions vanish on the empty set as well as on the full set V. In order to derive exact continuous relaxation of (5.12), we generalize this result to set functions not vanishing on the full set V.

Theorem 5.3 *Let $\hat{R}, \hat{S} : 2^V \to \mathbb{R}$ be any non-negative set functions and $R, S : \mathbb{R}^n \to \mathbb{R}$ be their Lovász extensions respectively, where $n = |V|$. Further let $\hat{R}(\emptyset) = \hat{S}(\emptyset) = 0$. Then it holds that*

$$\inf_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)} = \inf_{f \in \mathbb{R}^n_+} \frac{R(f)}{S(f)} \ .$$

Moreover, it holds for all $f \in \mathbb{R}^n_+$,

$$\min_{i=0,\ldots,n-1} \frac{\hat{R}(C_i)}{\hat{S}(C_i)} \leq \frac{R(f)}{S(f)},$$

where the sets C_i are defined as

$$C_0 = V, \ C_i = \{j \in V | f_j > f_i\}, \ i = 1, \ldots, n. \tag{5.13}$$

That is a minimizer C^* of the ratio of set functions can be obtained by optimal thresholding of any minimizer f^* of the continuous problem.

Proof: The proof is similar to that of Theorem 3.2 except that the terms involving $\hat{R}(V)$ and $\hat{S}(V)$ do not vanish here. Without loss of generality, we assume that components of $f \in \mathbb{R}^n_+$ are ordered in increasing order $f_1 \leq \ldots, \leq f_n$. We have from the definition of the Lovász extension (2.5)

$$\begin{aligned}
R(f) &= \sum_{i=1}^{n-1} \hat{R}(C_i)(f_{i+1} - f_i) + f_1 \hat{R}(V) \\
&= \sum_{i=1}^{n-1} \frac{\hat{R}(C_i)}{\hat{S}(C_i)} \hat{S}(C_i)(f_{i+1} - f_i) + f_1 \frac{\hat{R}(V)}{\hat{S}(V)} \hat{S}(V) \\
&\geq \min_{j=0,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} \left(\sum_{i=1}^{n-1} \hat{S}(C_i)(f_{i+1} - f_i) + f_1 \hat{S}(V) \right) = \min_{j=0,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} S(f).
\end{aligned}$$

Note that the inequality follows because of the non-negativity of f as well as $\hat{S}$. Since $\hat{S}$ is non-negative, it follows from the definition of the Lovász extension (2.5) that $S(f) \geq 0, \forall f \in \mathbb{R}^n_+$. Thus dividing both sides by $S(f)$, we have for any $f \in \mathbb{R}^n_+$,

$$\frac{R(f)}{S(f)} \geq \min_{j=0,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)}.$$

This implies

$$\min_{f \in \mathbb{R}^n_+} \frac{R(f)}{S(f)} \geq \min_{f \in \mathbb{R}^n_+} \min_{j=0,\ldots,n-1} \frac{\hat{R}(C_j)}{\hat{S}(C_j)} \geq \min_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)}.$$

Here we used the convention $\frac{0}{0} = \infty$ for the last inequality. On the other hand, the continuous problem is a relaxation of the combinatorial problem because $\hat{R}(C) = R(\mathbf{1}_C)$ and $\hat{S}(C) = S(\mathbf{1}_C), \ \forall C \subseteq V$. Thus by the virtue of relaxation, we have

$$\min_{f \in \mathbb{R}^n_+} \frac{R(f)}{S(f)} \leq \min_{C \subseteq V} \frac{\hat{R}(C)}{\hat{S}(C)},$$

which establishes the result. □

Now we discuss the exact continuous relaxation result for the combinatorial optimization problem (5.12). First note that because of the constants terms, the objective function of the problem (5.12) does not vanish on the empty set, a technical condition required in the definition of the Lovász extension. Thus, in oder to derive the exact continuous relaxation, we replace every constant term c in the objective by the set function $c\hat{P}_0(A)$, where $\hat{P}_0 : 2^V \to \mathbb{R}_+$ is defined as

$$\hat{P}_0(A) := \begin{cases} 0 & \text{if } A = \emptyset, \\ 1 & \text{otherwise} \end{cases}$$

Since the penalty function $\hat{T}^{(2)}$ contains the constant term $\sum_{j=1}^{p} k_j$, let us define a new penalty function $\hat{T}_0^{(2)}$ in its place as

$$\hat{T}_0^{(2)}(A) := \sum_{j=1}^{p} k_j \hat{P}_0(A) - \sum_{j=1}^{p} \min\{k_j, \mathrm{vol}_{M_j}(A)\}.$$

Note that $\hat{T}^{(2)}$ and $\hat{T}_0^{(2)}$ agree everywhere except on the empty set. We also define a new penalty function capturing the total amount of constraint violation by any non-empty set as

$$\hat{T}_0(A) := \hat{T}^{(1)}(A) + \hat{T}_0^{(2)}(A) + \hat{T}^{(3)}(A),$$

which again agrees with $\hat{T}$ on any non-empty set. Similarly, by replacing the constant terms, we change the objective function of problem (5.12) as

$$\hat{F}_0^{(\gamma)}(A) := \frac{\mathrm{vol}_g(A) + \mathrm{vol}_g(U)\hat{P}_0(A) + \gamma \hat{T}_0(A)}{\mathrm{vol}_d(A) - \mathrm{cut}_{G'}(A, V'\backslash A) + \mathrm{assoc}(U)\hat{P}_0(A) + \mathrm{vol}_{d^U}(A)}.$$

In the following we derive the exact relaxation result for the problem

$$\min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A). \tag{5.14}$$

The relation between the unconstrained problem (5.12) derived in the previous section and the modified problem (5.14) is given in the following lemma.

Lemma 5.2 *The problems* (5.12) *and* (5.14) *are related as*

$$\min_{A \subseteq V'} \hat{F}^{(\gamma)}(A) = \min\left\{ \hat{F}^{(\gamma)}(\emptyset), \min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A) \right\}.$$

Moreover, if $\hat{F}^{(\gamma)}(\emptyset) < \min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A)$*, then the empty set is a solution of the problem* (5.12)*; otherwise optimal solutions of the problems* (5.12) *and* (5.14) *are same.*

Proof: Note that the objective values of problems (5.12) and (5.14) agree everywhere except on the empty set. If $\hat{F}^{(\gamma)}(\emptyset) < \min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A)$, then the empty set is a minimizer of the problem (5.12) since $\hat{F}_0^{(\gamma)}(\emptyset) = \frac{0}{0} := \infty$. Otherwise there is a non-empty set A that minimizes both (5.12) and (5.14). □

Thus it is sufficient to derive a method for solving the problem (5.14). We need the following result to present the exact continuous relaxation of the problem (5.14).

Lemma 5.3 *Let $\hat{Q}_{(l,h)} : 2^{V'} \to \mathbb{R}$ be defined as $\hat{Q}_{(l,h)}(A) = \min\{l, \mathrm{vol}_h(A)\}$ for any $l \in \mathbb{R},\ h \in \mathbb{R}^m_+$. The function $\hat{Q}_{(l,h)}$ is submodular. Moreover, an element of the subdifferential of the Lovász extension of $\hat{Q}_{(l,h)}$ is given by*

$$\left(q_{(l,h)}(f)\right)_{j_i} = \begin{cases} 0 & \mathrm{vol}_h(A_i) > l \\ l - \mathrm{vol}_h(A_i) & \mathrm{vol}_h(A_{i-1}) \geq l,\ \mathrm{vol}_h(A_i) \leq l \\ h_{j_i} & \mathrm{vol}_h(A_{i-1}) < l \end{cases}, \tag{5.15}$$

where j_i denotes the index of the i^{th} smallest component of f and the sets $A_i,\ i = 0, 1, \ldots, m$ are defined as

$$A_0 = V',\ A_i = \{j_{i+1}, j_{i+2}, \ldots, j_m\},\ i = 1, \ldots, m. \tag{5.16}$$

Proof: It has been shown in Lemma 2.4 that $\mathrm{vol}_h(A)$ is a submodular function. It is easy to derive from the definition of submodularity that pointwise minimum of a constant and an increasing submodular function is again submodular. Thus the function $\hat{Q}_{(l,h)}$ is submodular and its Lovász extension is convex by Proposition 2.2. By Lemma 2.7, an element of the subdifferential of the Lovász extension of $\hat{Q}$ is given by

$$q_{j_i} = \hat{Q}\left(\{j_i, \ldots, j_m\}\right) - \hat{Q}\left(\{j_{i+1}, \ldots, j_m\}\right),\ i = 1, \ldots, m,$$

where $(j_1, \ldots, j_m)$ is a permutation of V' satisfying $f_{j_1} \leq f_{j_2} \leq \ldots \leq f_{j_m}$. The fact that the function $\mathrm{vol}_h(A_i)$ is monotonically decreasing in i (since $h \in \mathbb{R}^m_+$) simplifies the expression for $q_{(l,h)}$. □

Theorem 5.4 *The combinatorial optimization problem (5.14) is equivalent to the continuous problem*

$$\min_{f \in \mathbb{R}^m_+} \frac{R_1(f) - R_2(f)}{S_1(f) - S_2(f)} =: F_0^{(\gamma)}(f), \tag{5.17}$$

for any $\gamma \geq 0$, where

$$\begin{aligned}
R_1(f) &= \left\langle f, \gamma \bar{d} + (\rho)_{i=1}^m \right\rangle + \sigma \max_i \{f_i\},\ \rho = g + \gamma \sum_{j=1}^p M_j,\ \sigma = \mathrm{vol}_g(U) + \gamma \sum_{j=1}^p k_j, \\
R_2(f) &= \gamma \sum_{j=1}^p \left\langle f, q_{(l_j, M_j)}(f) \right\rangle + \gamma \sum_{j=1}^p \left\langle f, q_{(k_j, M_j)}(f) \right\rangle + \gamma \frac{1}{2} \sum_{i,j \in V'} D_{ij} \left| f_i - f_j \right|, \\
S_1(f) &= \left\langle f, (d_i)_{i=1}^m + (d_i^U)_{i=1}^m \right\rangle + \mathrm{assoc}(U) \max_i \{f_i\} \\
S_2(f) &= \frac{1}{2} \sum_{i,j \in V'} w'_{ij} \left| f_i - f_j \right|,
\end{aligned}$$

and $q_{(l,h)}$ is defined in (5.15). The functions R_1, R_2, S_1 and S_2 are convex, positively 1-homogeneous and $R_1 - R_2$, $S_1 - S_2$ are non-negative on the positive orthant $\mathbb{R}^m_+$. Moreover, it holds for any $f \in \mathbb{R}^m_+$,

$$\min_{i=0,\ldots m-1} \hat{F}_0^{(\gamma)}(A_i) \leq F_0^{(\gamma)}(f),$$

where the set A_i are defined as

$$A_0 = V',\ A_i = \{j \in V' | f_j > f_i\},\ i = 1, \ldots, m. \tag{5.18}$$

That is a minimizer A^* of the combinatorial optimization problem (5.14) *can be obtained by optimal thresholding of any minimizer f^* of the continuous problem* (5.17).

Proof: The essential idea is to replace the set functions in the objective of (5.14) by their Lovász extensions and invoke Theorem 5.3. Let $\hat{R}$ denote the numerator of the ratio $\hat{F}_0^{(\gamma)}$,

$$\hat{R}(A) = \mathrm{vol}_g(A) + \mathrm{vol}_g(U)\hat{P}_0(A) + \gamma\hat{T}_0(A).$$

We decompose $\hat{R}$ as the difference $\hat{R}_1 - \hat{R}_2$, where

$$\hat{R}_1(A) = \mathrm{vol}_g(A) + \mathrm{vol}_g(U)\hat{P}_0(A) + \gamma\sum_{j=1}^{p}\mathrm{vol}_{M_j}(A) + \gamma\sum_{j=1}^{p}k_j\hat{P}_0(A) + \gamma\,\mathrm{vol}_{\bar{d}}(A),$$

$$\hat{R}_2(A) = \gamma\sum_{j=1}^{p}\min\{l_j, \mathrm{vol}_{M_j}\} + \gamma\sum_{j=1}^{p}\min\{k_j, \mathrm{vol}_{M_j}\} + \gamma\mathrm{cut}_H(A, V'\backslash A).$$

Note that both $\hat{R}_1$ and $\hat{R}_2$ are submodular because of the non-negativity of $\gamma, k_j,\ j = 1, \ldots, p$ and $\mathrm{vol}_g(U)$. Their Lovász extensions are positively 1-homogeneous convex functions and are given by the functions R_1 and R_2 respectively (by Lemmas 2.4, 2.5 and 5.3). Moreover, by Proposition 2.1, $R_1 - R_2$ is the Lovász extension of the non-negative set function $\hat{R}$ and it follows from the definition (2.5) that it is also non-negative on the positive orthant.

Similarly denote by $\hat{S}$ the denominator of $\hat{F}_0^{(\gamma)}$ and decompose it as the difference $\hat{S}(A) = \hat{S}_1(A) - \hat{S}_2(A)$, where

$$\hat{S}_1(A) = \mathrm{vol}_d(A) + \mathrm{assoc}(U)\hat{P}_0(A) + \mathrm{vol}_{d^U}(A)$$
$$\hat{S}_2(A) = \mathrm{cut}_{G'}(A, V'\backslash A).$$

Again one sees that S_1 and S_2 are submodular and their Lovász extensions are given by S_1 and S_2 respectively. Moreover since $\hat{S}$ is non-negative, its Lovász extension $S_1 - S_2$ is also non-negative on the positive orthant.

Thus we replaced all the set functions by their Lovász extensions in $F_0^{(\gamma)}$. The result now follows from Theorem 5.3. □

Now we state the main result connecting the generalized densest subgraph problem (5.2) and the continuous relaxation (5.17).

Theorem 5.5 *Let $G(V, W)$ be the given graph and let $G'(V', W')$ be the subgraph induced by the vertex set $V' = V\backslash U$ where U is the seed set. Let $A_0 \subseteq V$ be any subset satisfying the constraints of the generalized densest subgraph problem (5.2) such that* $\mathrm{density}(A_0) > 0$. *Moreover let ϕ^* be the optimal value of the problem* (5.2). *If $\gamma > \frac{\mathrm{vol}_d(V)}{\theta\mathrm{density}(A_0)}$ where θ is the minimum value of infeasibility defined in* (5.11) *of any set $A \subseteq V'$, then*

$$\phi^* = \max\left\{\frac{1}{\hat{F}^{(\gamma)}(\emptyset)}, \frac{1}{\bar{\phi}^*}\right\}, \tag{5.19}$$

where

$$\bar{\phi}^* = \min_{f \in \mathbb{R}^m_+} F_0^{(\gamma)}(f). \tag{5.20}$$

Moreover if $\hat{F}^{(\gamma)}(\emptyset) < \bar{\phi}^*$ *then* U *is an optimal solution of the problem* (5.2)*; otherwise an optimal solution* C^* *of the problem* (5.2) *can be obtained from a minimizer of the problem* (5.20)*.*

Proof: For the given value of γ, it follows from Theorem 5.2 and Lemma 5.2 that

$$\phi^* = \frac{1}{\min\left\{\hat{F}^{(\gamma)}(\emptyset), \min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A)\right\}} = \max\left\{\frac{1}{\hat{F}^{(\gamma)}(\emptyset)}, \frac{1}{\min_{A \subseteq V'} \hat{F}_0^{(\gamma)}(A)}\right\}.$$

Theorem 5.4 then yields the result stated in (5.19). It is clear that if $\hat{F}^{(\gamma)}(\emptyset) < \bar{\phi}^*$, then the empty set is an optimal solution for (5.12) and consequently U is optimal for (5.2). Otherwise by Theorem 5.4 one can obtain an optimal solution C^* of (5.2) from optimal thresholding of any minimizer of $F_0^{(\gamma)}$. □

5.4 Algorithm for the generalized densest subgraph problem

5.4.1 Solution via RatioDCA

Recall that RatioDCA described in Section 2.1.7 minimizes a non-negative ratio of positively 1-homogeneous, difference of convex functions on $\mathbb{R}^n$. Since the continuous relaxation (5.17) requires minimization over the positive orthant $\mathbb{R}^m_+$, we use a slight variant of RatioDCA algorithm. More specifically, we add the positive orthant constraint to the inner problem solved in RatioDCA. The resulting algorithm specialized to the problem (5.17) is given in Algorithm 5.1. The key part in the

Algorithm 5.1 Minimization of the ratio $F_0^{(\gamma)}$ over $\mathbb{R}^m_+$ using RatioDCA

1: **Initialization:** $f^0 \in \mathbb{R}^m_+$, $\lambda^0 = F_0^{(\gamma)}(f^0)$
2: **repeat**
3: $f^{t+1} = \underset{f \in \mathbb{R}^m_+,\ \|f\|_2 \leq 1}{\arg\min} \frac{\lambda^t}{2} \sum_{i,j=1}^m w'_{ij} |f_i - f_j| + \sigma \max_i\{f_i\} + \langle f, c^t \rangle\},$
where $c^t = \gamma \bar{d} + (\rho)_{i=1}^m - r_2 - \lambda^t s_1$, $r_2 \in \partial R_2(f^t)$ and $s_1 \in \partial S_1(f^t)$.
4: $\lambda^{t+1} = F_0^{(\gamma)}(f^{t+1})$
5: **until** $\frac{|\lambda^{t+1} - \lambda^t|}{\lambda^t} < \epsilon$
6: **Output:** λ^t and f^t

algorithm is solving the inner convex problem, which in our case has the form

$$\min_{f \in \mathbb{R}^m_+,\ \|f\| \leq 1} \frac{\lambda^t}{2} \sum_{i,j=1}^m w'_{ij} |f_i - f_j| + \sigma \max_i\{f_i\} + \left\langle f, c^t \right\rangle, \tag{5.21}$$

where $c^t = \gamma\bar{d} + (\rho)_{i=1}^m - r_2 - \lambda^t s_1$, $\rho = g + \gamma\sum_{j=1}^p M_j$, $\sigma = \mathrm{vol}_g(U) + \gamma\sum_{j=1}^p k_j$. Moreover, $r_2(f^t), s_1(f^t)$ are respectively the elements of the subdifferentials of R_2 and S_1 at f^t. By Lemma 5.3, $r_2(f^t)$ is given by

$$r_2 = \gamma\textstyle\sum_{j=1}^p q_{(l_j,M_j)}(f) + \gamma\sum_{j=1}^p q_{(k_j,M_j)}(f) + \gamma\partial\mathrm{TV}_H(f^t),$$

with $\partial\left(\mathrm{TV}_H(f^t)\right)_i = \left\{\sum_{j=1}^m D_{ij}u_{ij} \,|\, u_{ij} = -u_{ji}, u_{ij} \in \mathrm{sign}(f_i^t - f_j^t)\right\}$, where $\mathrm{sign}(x) := +1$, if $x > 0$; -1 if $x < 0$; $[-1, 1]$, if $x = 0$ (see Lemma 2.6). It is easy check that

$$s_1 = (d_i)_{i=1}^m + (d_i^U)_{i=1}^m + \mathrm{assoc}(U)\frac{1}{|C_{f^t}|}\mathbf{1}_{C_f^t},$$

where C_{f^t} is the set of indices where f^t has the maximum value.

Note that after solving the continuous relaxation (5.17) (and hence its combinatorial counter part (5.14)) one has to derive the solution for the original problem (5.12) using Lemma 5.2 where we check if the empty set is in fact the solution of (5.12). In the following we assume that this step is included as a part of our overall method.

5.4.2 Quality guarantee for our method

Although the problem of minimizing $F_0^{(\gamma)}$ is non-convex and hence global convergence is not guaranteed, we have the following quality guarantee for our overall method.

Theorem 5.6 *Let $G(V, W)$ be the given graph and let $G'(V', W')$ be the subgraph induced by the vertex set $V' = V\backslash U$ where U is the seed set. Let $C_0 \subseteq V$ be any subset containing the seed set U such that $C_0\backslash U \neq \emptyset$ and one uses $\mathbf{1}_{C_0\backslash U}$ as an initialization, then our overall method either terminates in one iteration or outputs $\bar{f}$ which yields a subset $A \subseteq V'$ satisfying*

$$\hat{F}^{(\gamma)}(A) < \hat{F}^{(\gamma)}(C_0\backslash U),$$

for any $\gamma \geq 0$. Moreover assume that $C_0 \subseteq V$ is any subset that satisfies all the constraints of the generalized densest subgraph problem (5.2) such that $C_0\backslash U \neq \emptyset$ and density$(C_0) > 0$. *If one uses $\mathbf{1}_{C_0\backslash U}$ as initialization and $\gamma > \frac{\mathrm{vol}_d(V)}{\theta\,\mathrm{density}(C_0)}$ where θ is the minimum value of infeasibility defined in* (5.11) *of any set $A \subseteq V'$, then the output of our overall method $C \subseteq V$ satisfies all the constraints of* (5.2) *and has a strictly larger density,*

$$\mathrm{density}(C) > \mathrm{density}(C_0)$$

Proof: First note that $F_0^{(\gamma)}$ and $\hat{F}_0^{(\gamma)}$ satisfy $F_0^{(\gamma)}(\mathbf{1}_A) = \hat{F}_0^{(\gamma)}(A)$ for any $A \subseteq V'$. Let $f^0 = \mathbf{1}_{C_0\backslash U}$ be the initialization of the RatioDCA algorithm for some subset $C_0\backslash U \neq \emptyset$. It has been shown that RatioDCA either stops in one iteration or outputs a non-constant $\bar{f}$ with a strictly smaller objective (see Proposition 2.5). In our setting this means, if the algorithm does not terminate in one iteration, that $F_0^{(\gamma)}(\bar{f}) < F_0^{(\gamma)}(f^0) = \hat{F}_0^{(\gamma)}(C_0\backslash U)$. As shown in theorem 5.4, optimal thresholding of $\bar{f}$ results in a subset $A_{\bar{f}}$ satisfying $\hat{F}_0^{(\gamma)}(A_{\bar{f}}) \leq F_0^{(\gamma)}(\bar{f})$ and hence we have

$$\hat{F}_0^{(\gamma)}(A_{\bar{f}}) < \hat{F}_0^{(\gamma)}(C_0\backslash U).$$

Since we choose either the empty set or $A_{\bar{f}}$ depending on whichever has a smaller objective, it holds that our overall method returns A where

$$\hat{F}^{(\gamma)}(A) \leq \min\{\hat{F}^{(\gamma)}(\emptyset), \hat{F}_0^{(\gamma)}(A_{\bar{f}})\} \leq \hat{F}_0^{(\gamma)}(A_{\bar{f}}) < \hat{F}_0^{(\gamma)}(C_0 \backslash U) = \hat{F}^{(\gamma)}(C_0 \backslash U), \tag{5.22}$$

where the last equality follows because $\hat{F}_0^{(\gamma)}$ and $\hat{F}^{(\gamma)}$ agree on non-empty subsets. For the chosen value of γ, using a similar argument as in Theorem 5.1, one sees that for any subset $B \subseteq V'$ that violates at least one constraint of (5.5), it holds that $\hat{F}^{(\gamma)}(B) > \hat{F}^{(\gamma)}(C_0 \backslash U)$. Since our method, if not terminated in one iteration, always returns a subset A that has a strictly smaller objective value than $\hat{F}^{(\gamma)}(C_0 \backslash U)$, A has to be feasible. For any feasible A, it holds that

$$\hat{F}^{(\gamma)}(A) = \frac{1}{\hat{G}(A)} = \frac{1}{\text{density}(A \cup U)}.$$

Moreover since C_0 is feasible for (5.2), $C_0 \backslash U$ is feasible (5.5) and hence

$$\hat{F}^{(\gamma)}(C_0 \backslash U) = \frac{1}{\hat{G}(C_0 \backslash U)} = \frac{1}{\text{density}(C_0)}.$$

This together with (5.22), shows that the density of $C = A \cup U$ is strictly larger than that of C_0. □

Directly solving the non-convex problem (5.17) for the value of γ given in Theorem 5.1 often yields poor results. Hence in our implementation we adopt the following strategy. We first solve the unconstrained version of problem (5.17) (i.e., $\gamma = 0$) and then iteratively solve (5.17) for increasing values of γ until all constraints are satisfied. In each iteration, we increase γ only for those constraints which were infeasible in the previous iteration; in this way, each penalty term is regulated by different value of γ. Moreover, the solution obtained in the previous iteration of γ is used as the starting point for the current iteration.

5.4.3 Smooth minimization of the inner problem

We first show that the solution of the inner problem (5.21) can be obtained by solving the following slightly related problem where the norm constraint is eliminated,

$$\min_{f \in \mathbb{R}_+^m} \frac{1}{2} \|f\|_2^2 + \frac{\lambda^l}{2} \sum_{i,j=1}^{m} w'_{ij} |f_i - f_j| + \sigma \max_i\{f_i\} + \langle f, c^t \rangle . \tag{5.23}$$

The following result is shown in [22, 20].

Lemma 5.4 *([20]) Let* $\Phi(f) = \frac{\lambda^l}{2} \sum_{i,j=1}^{m} w'_{ij} |f_i - f_j| + \langle f, c \rangle + \sigma \max_i\{f_i\}$ *and*

$$f^* \in \arg\min_{f \in \mathbb{R}_+^m} \Phi(f) + \frac{1}{2} \|f\|_2^2 .$$

Then it holds that

$$f' \in \arg\min_{f \in \mathbb{R}_+^m,\ \|f\|_2 \leq 1} \Phi(f)$$

where

$$f' = \begin{cases} \frac{f^*}{\|f^*\|} & f^* \neq 0, \\ 0 & \textit{otherwise} \end{cases}$$

Proof: First assume that $f^* \neq 0$. Then we have

$$\begin{aligned}
\Phi\left(\frac{f^*}{\|f^*\|_2}\right) &= \frac{1}{\|f^*\|_2}\left(\Phi(f^*) + \frac{1}{2}\|f^*\|_2^2\right) - \frac{1}{2}\|f^*\|_2 \\
&= \frac{1}{\|f^*\|_2}\left(\min_{f\in\mathbb{R}^m_+} \Phi(f) + \frac{1}{2}\|f\|_2^2\right) - \frac{1}{2}\|f^*\|_2 \\
&= \frac{1}{\|f^*\|_2}\left(\min_{f\in\mathbb{R}^m_+,\ \|f\|_2=\|f^*\|_2} \Phi(f) + \frac{1}{2}\|f\|_2^2\right) - \frac{1}{2}\|f^*\|_2 \\
&= \min_{f\in\mathbb{R}^m_+,\ \|f\|_2=\|f^*\|_2} \Phi\left(\frac{f}{\|f^*\|_2}\right) + \frac{1}{2}\|f^*\|_2 - \frac{1}{2}\|f^*\|_2 \\
&= \min_{g\in\mathbb{R}^m_+,\ \|g\|_2=1} \Phi(g),
\end{aligned}$$

where in the last step we used the variable substitution $g = \frac{f}{\|f^*\|}$. Thus

$$\frac{f^*}{\|f^*\|_2} \in \operatorname*{arg\,min}_{g\in\mathbb{R}^m_+,\ \|g\|_2=1} \Phi(g). \tag{5.24}$$

Since Φ is a positively 1-homogeneous function, i.e., $\Phi(\alpha f) = \alpha\Phi(f)$ for any $\alpha \geq 0$, we have $\Phi(0) = 0$. Thus, we have $\min_{f\in\mathbb{R}^m_+} \Phi(f) + \frac{1}{2}\|f\|_2^2 \leq 0$. Moreover, since f^* is a minimizer of the problem $\min_{f\in\mathbb{R}^m_+} \Phi(f) + \frac{1}{2}\|f\|_2^2$ and $f^* \neq 0$, we have

$$\Phi(f^*) + \frac{1}{2}\|f^*\|_2^2 \leq 0. \implies \Phi\left(\frac{f^*}{\|f^*\|_2}\right) \leq -\frac{1}{2}\|f^*\|_2 < 0.$$

Thus, $\min_{g\in\mathbb{R}^m_+,\ \|g\|_2\leq 1} \Phi(g)$ is negative since $\frac{f^*}{\|f^*\|_2}$ is feasible for this problem. Then, by 1-homogeneity of Φ, any minimizer g of the problem $\min_{g\in\mathbb{R}^m_+,\ \|g\|_2\leq 1} \Phi(g)$ satisfies $\|g\| = 1$. Thus, under the assumption $f^* \neq 0$, we have using (5.24)

$$\operatorname*{arg\,min}_{g\in\mathbb{R}^m_+,\ \|g\|_2\leq 1} \Phi(g) = \operatorname*{arg\,min}_{g\in\mathbb{R}^m_+,\ \|g\|_2=1} \Phi(g) \ni \frac{f^*}{\|f^*\|_2}.$$

On the other hand, if $f^* = 0$, we claim that $\min_{f\in\mathbb{R}^m_+,\ \|f\|_2\leq 1} \Phi(f) = 0$. For the sake of contradiction, assume that $\Phi(f') < 0$ where $f' \in \arg\min_{f\in\mathbb{R}^m_+,\ \|f\|_2\leq 1} \Phi(f)$. By 1-homogeneity of Φ this implies that $\|f'\|_2 = 1$. Now define $g = \alpha f'$ for some $\alpha \in (0, -2\Phi(f'))$. Since $\alpha > 0$, we have $g \in \mathbb{R}^m_+$ and

$$\Phi(g) + \frac{1}{2}\|g\|_2^2 = \alpha\Phi(f') + \frac{1}{2}\alpha^2\|f'\|_2^2 = \alpha\left(\Phi(f') + \frac{1}{2}\alpha\right) < 0,$$

which is contradiction to the fact that $f^* = 0$ is an optimal solution of the problem $\arg\min_{f\in\mathbb{R}^m_+} \Phi(f) + \frac{1}{2}\|f\|_2^2$. Hence $\min_{f\in\mathbb{R}^m_+,\ \|f\|_2\leq 1} \Phi(f) = 0$ and $f' = 0$ is a minimizer of this problem. □

We will now rewrite the non-smooth convex problem (5.23) as an equivalent problem where the objective function has Lipschitz continuous gradient. In the following we use the linear operator introduced in Section 3.5.3.

Proposition 5.1 *Let $E' \subseteq V' \times V'$ denote the set of edges of the graph $G'(V', W')$, $\overrightarrow{E'} = \{(i,j) \in E',\ i < j\}$ denote the directed edges. Let $A : \mathbb{R}^{\overrightarrow{E'}} \to \mathbb{R}^{V'}$ be a linear operator defined as in (3.15). Moreover, let $\Delta_m = \{x \in \mathbb{R}^m : x_i \geq 0,\ \sum_{i=1}^m x_i = 1\}$. The inner problem (5.23) is equivalent to*

$$\min_{\alpha \in \mathbb{R}^{\overrightarrow{E'}},\ \|\alpha\|_\infty \leq 1} \min_{x \in \Delta_m} \frac{1}{2} \left\| P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right) \right\|_2^2 =: \Psi(\alpha, x), \tag{5.25}$$

where $P_{\mathbb{R}^m_+}(\cdot)$ denotes the projection on to the positive orthant $\mathbb{R}^m_+$. The gradient of the objective function Ψ at α is given by

$$\nabla \Psi(\alpha, x) = \begin{pmatrix} -\lambda^t A^T P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right) \\ -\sigma P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right) \end{pmatrix}$$

where the adjoint A^T is given in (3.16). Moreover, the Lipschitz constant of the gradient of Ψ is upper bounded by $\sqrt{2} \max\left\{\lambda^t \|A\|_2, \sigma\right\} \sqrt{(\lambda^t)^2 \|A\|_2^2 + \sigma^2}$.

Proof: Note that $\max_i\{f_i\} = \max_{x \in \Delta_n} \langle x, f \rangle$, where Δ_m is the simplex defined as $\Delta_n = \{x \in \mathbb{R}^m,\ x \geq 0, \sum_{i=1}^m x_i = 1\}$. Using Lemma 3.5 we rewrite the inner problem (5.23) as

$$\begin{aligned} &\min_{f \in \mathbb{R}^m_+} \frac{1}{2} \|f\|_2^2 + \max_{\{\alpha \in \mathbb{R}^{\overrightarrow{E'}},\ \|\alpha\|_\infty \leq 1\}} \lambda^t \langle f, A\alpha \rangle + \sigma \max_{x \in \Delta_n} \langle f, x \rangle + \left\langle f, c^t \right\rangle \\ = &\max_{\{\alpha \in \mathbb{R}^{\overrightarrow{E'}},\ \|\alpha\|_\infty \leq 1\}} \max_{x \in \Delta_n} \min_{f \in \mathbb{R}^m_+} \frac{1}{2} \|f\|_2^2 + \left\langle f, \lambda^t A\alpha + \sigma x + c^t \right\rangle, \end{aligned}$$

The optimization over f has the closed-form solution given by

$$f^* = P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right).$$

Plugging f^* into the objective and using that $\left\langle P_{\mathbb{R}^m_+}(v), v \right\rangle = \left\| P_{\mathbb{R}^m_+}(v) \right\|_2^2$ yields the problem (5.25).

It is easy to check that the function $\frac{1}{2} \max\{0, y\}^2,\ y \in \mathbb{R}$ is differentiable and the first derivative is given by $\max\{0, y\}$. Thus the partial derivatives of the function $h(v) := \frac{1}{2} \|P_{\mathbb{R}^m +}(v)\|_2^2,\ v \in \mathbb{R}^m$ are given by $\max\{0, v_k\},\ k = 1, \ldots, m$. Since the partial derivatives are also continuous, the function h is differentiable at any $v \in \mathbb{R}^m$ (see Theorem 17.3.8 in [113]). Moreover the function $-\frac{\lambda^l}{2} A\alpha - \sigma x - c^t$ is also differentiable with respect to α and x. Thus the objective function $\Psi(\alpha, x)$ of (5.25) is differentiable (see Theorem 17.4.1 in [113]) and one obtains its gradient from the chain rule as

$$\nabla \Psi(\alpha, x) = \begin{pmatrix} -\lambda^t A^T P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right) \\ -\sigma P_{\mathbb{R}^m_+} \left(-\lambda^t A\alpha - \sigma x - c^t \right) \end{pmatrix}$$

Finally, we derive the Lipschitz constant of the gradient as follows. For any $\alpha, \beta \in \mathbb{R}^{\overrightarrow{E'}}$ and $x, y \in \mathbb{R}^m$, we have

$$\begin{aligned} \|\nabla \Psi(\alpha, x) - \nabla \Psi(\beta, y)\|_2^2 = (\lambda^t)^2 &\left\| A^T \left(P_{\mathbb{R}^m_+}(z) - P_{\mathbb{R}^m_+}(z') \right) \right\|_2^2 \\ &+ \sigma^2 \left\| P_{\mathbb{R}^m_+}(z) - P_{\mathbb{R}^m_+}(z') \right\|_2^2, \end{aligned}$$

where we used the notation $z = -\lambda^t A\alpha - \sigma x - c^t$ and $z' = -\lambda^t A\beta - \sigma y - c^t$. Since $\left\| P_{\mathbb{R}^m_+}(z) - P_{\mathbb{R}^m_+}(z') \right\|_2 \leq \|(z - z')\|_2$ (because of the non-expansiveness property of the projection operator [99]), we have

$$\begin{aligned} \|\nabla\Psi(\alpha, x) - \nabla\Psi(\beta, y)\|_2^2 &\leq (\lambda^t)^2 \left\|A^T\right\|_2^2 \|(z - z')\|_2^2 + \sigma^2 \|z - z'\|_2^2 \\ &\leq \left((\lambda^t)^2 \left\|A^T\right\|_2^2 + \sigma^2\right) \|z - z'\|_2^2, \end{aligned} \tag{5.26}$$

where the first inequality follows from the definition of the norm. We simplify the final term as

$$\begin{aligned} \|z - z'\|_2^2 &= \left\|-\lambda^t A\alpha - \sigma x + \lambda^t A\beta + \sigma y\right\|_2^2 = \left\|\lambda^t A(\alpha - \beta) + \sigma(x - y)\right\|_2^2 \\ &\leq 2\left\|\lambda^t A(\alpha - \beta)\right\|_2^2 + 2\|\sigma(x - y)\|_2^2 \leq 2(\lambda^t)^2 \|A\|_2^2 \|\alpha - \beta\|_2^2 + 2\sigma^2 \|x - y\|_2^2 \\ &\leq 2\max\left\{(\lambda^t)^2 \|A\|_2^2, \sigma^2\right\} \left(\|\alpha - \beta\|_2^2 + \|x - y\|_2^2\right). \end{aligned}$$

Combining this with (5.26) and using $\|A\| = \left\|A^T\right\|$, we have

$$\begin{aligned} \|\nabla\Psi(\alpha, x) - \nabla\Psi(\beta, y)\|_2^2 \leq \left((\lambda^t)^2 \|A\|_2^2 + \sigma^2\right) 2\max\left\{(\lambda^t)^2 \|A\|_2^2, \sigma^2\right\} \\ \left(\|\alpha - \beta\|_2^2 + \|x - y\|_2^2\right). \end{aligned}$$

Thus the Lipschitz constant is given by $\sqrt{2}\max\left\{\lambda^t \|A\|_2, \sigma\right\} \sqrt{(\lambda^t)^2 \|A\|_2^2 + \sigma^2}$. □

The dual problem (5.25) can now be solved very efficiently using recent first order methods like FISTA [15] (see Section 3.5.4). Comparing with the general model (3.23) solved by FISTA, we identify for our problem (5.25)

$$F(\alpha, x) = \frac{1}{2}\left\|P_{R^m_+}\left(-\lambda^t A\alpha - \sigma x - c^t\right)\right\|_2^2, \quad G(\alpha, x) = \iota_{\mathcal{B}}(\alpha) + \iota_{\Delta_m}(x),$$

where $\mathcal{B} = \{\alpha \in \mathbb{R}^{\overrightarrow{E'}},\ -1 \leq \alpha_{ij} \leq 1,\ \forall (i,j) \in \overrightarrow{E'}\}$ and $\iota_{\mathcal{D}}$ for a convex set $\mathcal{D}$ is the indicator function defined as

$$\iota_{\mathcal{D}}(\beta) = \begin{cases} 0, & \beta \in \mathcal{D} \\ \infty & \text{otherwise} \end{cases} \tag{5.27}$$

Thus the proximal problem one has to solve in our setting is given by (see Step 4 of Algorithm 3.2)

$$\begin{aligned} \alpha^{p+1} &= \underset{\alpha \in \mathbb{R}^{\overrightarrow{E'}},\ \|\alpha\|_\infty \leq 1}{\arg\min} \frac{1}{2}\|\alpha - c^p\|_2^2 \\ x^{p+1} &= \underset{x \in \Delta_m}{\arg\min} \frac{1}{2}\|x - a^p\|_2^2 \end{aligned}$$

where $c^p = \bar{\alpha}^p + \frac{1}{L}\lambda^t A^T z^p$ and $a^p = \bar{x}^p + \frac{1}{L}\sigma z^p$ and $z^p = P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^p - \sigma x^p - c^t\right)$. The first problem on α can be solved in closed-form for each variable independently

$$\alpha_{ij}^{p+1} = P_{[-1,1]}(c_{ij}^p) = \max\{-1, \min\{1, c_{ij}^p\}\},\ \forall (i,j) \in \overrightarrow{E'},$$

Algorithm 5.2 Solution of the inner problem (5.25) using FISTA

1: **Input:** Lipschitz constant L of $\nabla\Psi$
2: **Initialization:** $\alpha^0 \in \mathbb{R}^{\overrightarrow{E'}}$, $\bar{\alpha}^0 = \alpha^0$, $x^0 \in \mathbb{R}^m$, $\bar{x}^0 = x^0$, $t_1 = 1,\ p = 0$.
3: **repeat**
4: $z^p = P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^p - \sigma x^p - c^t\right)$
5: $c^p = \bar{\alpha}^p + \frac{1}{L}\lambda^t A^T z^p$
6: $a^p = \bar{x}^p + \frac{1}{L}\sigma z^p$
7: $\alpha_{ij}^{p+1} = \max\{-1, \min\{1, c_{ij}^p\}\},\ \forall (i,j) \in \overrightarrow{E'}$
8: $x^{p+1} = P_{\Delta_m}(a^p)$
9: $t_{p+1} = \frac{1+\sqrt{1+4t_p^2}}{2}$
10: $\bar{\alpha}^{p+1} = \alpha^{p+1} + \frac{t_p - 1}{t_{p+1}}(\alpha^{p+1} - \alpha^p)$
11: $\bar{x}^{p+1} = x^{p+1} + \frac{t_p - 1}{t_{p+1}}(x^{p+1} - x^p)$
12: $p = p + 1$
13: **until** convergence
14: **Output:** α^p, x^p

where $P_{[-1,1]}(\cdot)$ denotes the projection onto the interval $[-1, 1]$. Note that the update of x is given by the projection on to the simplex. The complete details are given in Algorithm 5.2.

One can verify that each iteration takes linear time $O(|V'| + \left|\overrightarrow{E'}\right|)$ because the linear operators A and A^T take $O(\overrightarrow{E'})$ time and the projection on to the simplex (step 8) $O(V')$ time [69]. We can check convergence by examining the gap between the objective values of the dual problem (5.25) and the primal problem (5.23). The solution of (5.23) can be obtained from the solution α^* and x^* of (5.25) as (see Proposition 5.1)

$$f^* = P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^* - \sigma x^* - c^t\right).$$

Finally the solution of (5.21) can be obtained as (from Lemma 5.4)

$$f^* = \frac{P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^* - \sigma x^* - c^t\right)}{\left\|P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^* - \sigma x^* - c^t\right)\right\|_2}, \quad \text{if } \left\|P_{\mathbb{R}^m_+}\left(-\lambda^t A\alpha^* - \sigma x^* - c^t\right)\right\|_2 \neq 0,$$

otherwise $f^* = 0$.

5.5 LP relaxation of the generalized densest subgraph problem

In this section we show that there exists a linear programming (LP) relaxation for the generalized densest subgraph problem. Here, we directly consider the version

where the seed constraint has been integrated:

$$\begin{aligned} \max_{A \subseteq V'} \; & \frac{\mathrm{vol}_d(A) - \mathrm{cut}_{G'}(A, V' \backslash A) + \mathrm{assoc}(U) + \mathrm{vol}_{d^U}(A)}{\mathrm{vol}_g(A) + \mathrm{vol}_g(U)} \\ \text{subject to} : : \; & k_j \leq \mathrm{vol}_{M_j}(A) \leq l_j, \quad \forall j \in \{1, \ldots, p\} \\ & \mathrm{dist}(u, v) \leq d_0, \quad \forall u, v \in A \end{aligned} \tag{5.28}$$

The LP relaxation, derived in the following, can be solved optimally in polynomial time and provides an upper bound on the optimum value of the problem (5.28). In practice such an upper bound is useful in obtaining bounds on the quality of the solutions found by approximation algorithms. This is because the global optimum value is guaranteed to lie in the interval given by the objective value of the solution found by the approximation algorithm and the upper bound given by the relaxation.

Theorem 5.7 *Let $G(V, W)$ be the given graph and U be the seed set. The following LP is a relaxation of the generalized densest subgraph problem (5.28).*

$$\begin{aligned} \max_{t \in \mathbb{R},\ f \in \mathbb{R}^{V'},\ \alpha \in \mathbb{R}^{E'}} \; & \sum_{(i,j) \in E'} w'_{ij} \alpha_{ij} + 2 \left\langle d^U, f \right\rangle + t \ \mathrm{assoc}(U) \\ \text{subject to} : : \; & t k_j \leq \langle M_j, f \rangle \leq t l_j, \quad \forall j \in \{1, \ldots, p\} \\ & f_u + f_v \leq t, \quad \forall u, v : \mathrm{dist}(u, v) > d_0 \\ & t \geq 0, \quad \alpha_{ij} \leq f_i, \ \alpha_{ij} \leq f_j, \ \forall (i, j) \in E' \\ & 0 \leq f_i \leq t, \ \forall i \in V', \ \alpha_{ij} \geq 0, \ \forall (i, j) \in E' \\ & \langle g, f \rangle + t \, \mathrm{vol}_g(U) = 1. \end{aligned} \tag{5.29}$$

where $V' = V \backslash U$, E' is the set of edges induced by V' and $w'_{ij} = w_{ij}, i, j \in V'$.

Proof: First note that from Equation (5.6) shown in Lemma 5.1, the numerator of (5.28) can be rewritten as

$$\mathrm{assoc}_{G'}(A) + \mathrm{assoc}_G(U) + 2 \ \mathrm{cut}_G(A, U),$$

since $\mathrm{cut}_G(A, V' \backslash A) = \mathrm{cut}_{G'}(A, V' \backslash A)$ and $\mathrm{assoc}_{G'}(A) = \mathrm{assoc}_G(A)$ for any $A \subseteq V'$. The following problem is equivalent to (5.28), because (i) for every feasible set A of (5.28), there exist corresponding feasible y, X given by $y = \mathbf{1}_A$, $X_{ij} = \min\{y_i, y_j\}$, with the same objective value and (ii) an optimal solution of the following problem always satisfies $X^*_{ij} = \min\{y^*_i, y^*_j\}$:

$$\begin{aligned} \max_{y \in \{0,\ 1\}^{V'},\ X \in \{0,\ 1\}^{E'}} \; & \frac{2 \sum_{i<j} w'_{ij} X_{ij} + 2 \left\langle d^U, y \right\rangle + \mathrm{assoc}(U)}{\langle g, y \rangle + \mathrm{vol}_g(U)} \\ \text{subject to} : : \; & k_j \leq \langle M_j, y \rangle \leq l_j, \quad \forall j \in \{1, \ldots, p\} \\ & y_u + y_v \leq 1, \quad \forall u, v : \mathrm{dist}(u, v) > d_0 \\ & X_{ij} \leq y_i, \quad X_{ij} \leq y_j, \quad \forall (i, j) \in E' \end{aligned}$$

Relaxing the integrality constraints $y \in \{0, 1\}^{V'}, X \in \{0, 1\}^{E'}$ to interval constraints $y \in [0, 1]^{V'}, X \in [0, 1]^{E'}$ and using the variable substitution $X_{ij} = \frac{\alpha_{ij}}{t}$ and $y_i = \frac{f_i}{t}$,

for $t > 0$, we obtain the relaxation:

$$\begin{aligned}
\max_{t\in\mathbb{R},\ f\in\mathbb{R}^{V'},\ \alpha\in\mathbb{R}^{E'}} \quad & \frac{2\sum_{i<j} w'_{ij}\alpha_{ij} + 2\left\langle d^U, f\right\rangle + t\ \mathrm{assoc}(U)}{\langle g, f\rangle + t\,\mathrm{vol}_g(U)} \\
\text{subject to : : } \quad & tk_j \le \langle M_j, f\rangle \le tl_j, \quad \forall j \in \{1,\ldots,p\} \\
& f_u + f_v \le t, \quad \forall u, v : \mathrm{dist}(u,v) > d_0 \\
& t \ge 0, \quad \alpha_{ij} \le f_i,\ \alpha_{ij} \le f_j,\ \forall (i,j) \in E' \\
& 0 \le f_i \le t,\ \forall i \in V',\ \alpha_{ij} \ge 0,\ \forall (i,j) \in E'
\end{aligned}$$

Since this problem is invariant under scaling, we can fix the scale by setting the denominator to 1, which yields the equivalent LP stated in the theorem. □

Note that the solution f^* of the LP (5.29) is, in general, not integral, i.e., $f^* \notin \{0,1\}^{V'}$. One can use standard techniques of randomized rounding or optimal thresholding to derive an integral solution from f^*. However, the resulting integral solution may not necessarily give a subset that satisfies the constraints of (5.28). In the special case when there are only lower bound constraints, one can obtain a feasible set A by thresholding f^* (see (5.18)) according to the objective of (5.28). This is possible in this special case because there is always a threshold f_i^* which yields a non-empty subset A_i (in the worst case the full set V') satisfying all the lower bound constraints. In our experiments for this special case, we derived a feasible set from the solution of LP in this fashion by choosing the threshold that yields a subset that satisfies the constraints and has the highest objective value.

The LP relaxation (5.29), in the absence of seed and lower bound constraints, is vacuous in the sense that solution of the unconstrained problem (5.3) is still feasible for the LP. To see this, note that when there are no seed and lower bound constraints, the LP relaxation has the form,

$$\begin{aligned}
\max_{t\in\mathbb{R},\ f\in\mathbb{R}^{V'},\ \alpha\in\mathbb{R}^{E'}} \quad & \sum_{(i,j)\in E'} w'_{ij}\alpha_{ij} \\
\text{subject to : : } \quad & \langle M_j, f\rangle \le tl_j, \quad \forall j \in \{1,\ldots,p\} \\
& f_u + f_v \le t, \quad \forall u, v : \mathrm{dist}(u,v) > d_0 \\
& t \ge 0, \quad \alpha_{ij} \le f_i,\ \alpha_{ij} \le f_j,\ \forall (i,j) \in E' \\
& 0 \le f_i \le t,\ \forall i \in V',\ \alpha_{ij} \ge 0,\ \forall (i,j) \in E' \\
& \langle g, f\rangle = 1.
\end{aligned}$$

Since the variable t does not appear in the objective, one can set it to arbitrary large positive value in order to make the solution f^* of the unconstrained problem (5.3) feasible for this problem. However, in the presence of seed or lower bound constraints, t is not a free variable anymore and cannot be increased arbitrarily. Hence the LP relaxation is useful on the instances of (5.28) with at least one lower bound or a seed constraint (i.e., $\mathrm{vol}_g(U) > 0$).

5.6 Application: Team formation in social networks

In this section, we discuss team formation problem in social networks as an application of our model (5.2). Given a task, team formation is the problem of identifying a team of skilled experts based on the social network $G(V, W)$ reflecting their previous collaboration. Here V represents the set of n experts and $W \in \mathbb{R}_+^{n\times n}$ provides the compatibility between every pair of experts based on their previous collaboration. Each expert is assumed to possess one or more skills from a given skill set $\mathcal{A} = \{a_1, \ldots, a_p\}$. The task $\mathcal{T}$ is typically defined in terms of the type and the amount of skills required. The authors of [47] consider a variant of team formation problem where the task $\mathcal{T}$ is given by the set of pairs $\{(a_j, \kappa_j)\}_{j=1}^p$, where $a_j \in \mathcal{A}$, specifying that at least κ_j of skill a_j is required to finish the given task. They propose the following model based on the densest subgraph problem for building a team

$$\begin{aligned} &\max_{C\subseteq V} \frac{\text{assoc}(C)}{\text{vol}_g(C)} \\ &\text{subject to}: \text{vol}_{M_j}(C) \geq \kappa_j, \quad \forall j \in \{1, \ldots, p\} \end{aligned} \tag{5.30}$$

where $g = \mathbf{1}_n$ and $M \in \{0, 1\}^{n\times p}$ is the *binary* skill matrix with the entries M_{ij} indicating whether the expert i possesses the skill j or not. Here we used the notation M_j to denote the j^{th} column of the skill matrix M. The density based objective possesses useful properties like strict monotonicity and robustness unlike diameter based measures used in the literature. In case of the density based objective, if an edge gets added (because of a new collaboration) or deleted (because of newly found incompatibility) the density of the subgraphs involving this edge necessarily increases respectively decreases, which is not true for the diameter based objective. In contrast to density based objective, the impact of small changes in graph structure is more severe in the case of diameter objective [47].

By extending the greedy method of [68], it is shown in [47] that there exists a 3-approximation algorithm for some special cases of the problem (5.30). The time complexity of this greedy algorithm is $O(kn^3)$, where n is the number of experts and $k := \sum_{j=1}^m \kappa_j$ is the minimum number of experts required. We note here that this approximation guarantee has been improved to a factor of two in [23] for the problem (5.30).

The main drawback of the above model (5.30) is that it does not allow encoding more natural requirements such an upper bound on the team size or the cost of the team. Note that adding upper bound constraints to the densest subgraph problem makes the problem much harder. We consider the team formation problem in a more realistic setting and allow formulation of the following natural requirements.

- **Inclusion of a specified group:** We allow a group of experts $U \subseteq V$ around whom the team has to be formed, i.e., $U \subseteq C$. This constraint is useful because the team leader is usually fixed before forming the team.

- **Skill requirement:** We redefine the task $\mathcal{T}$ as the set of triples $\{(a_j, \kappa_j, \iota_j)\}_{j=1}^p$, where $a_j \in \mathcal{A}$, specifying that at least κ_j and at most ι_j of skill a_j is required

to finish the given task. This way we allow an upper bound l_j on each skill a_j.

- **Bound on the team size:** We allow an upper bound on the size of the team, i.e., $|C| \leq k$ for a given value of k.
- **Budget constraint:** We associate a cost c_i, $\forall i \in V$ to each expert and allow a budget constraint, i.e., $\sum_{i \in C} c_i \leq b$ for a given value of b.
- **Locality of the team:** Another important generalization of our formulation is the inclusion of *distance* constraints for any general distance function[1]. Let dist be any distance function between two experts measured according to some non-negative, symmetric function. One can specify in our formulation that the distance between any pair of experts in C should not be larger than a given value d_0, i.e., $\text{dist}(u, v) \leq d_0, \forall u, v \in C$. Such a constraint can be used to enforce locality of the team e.g. in a geographical sense (the distance could be travel time) or social sense (distance in the network). Another potential application is the modeling of mutual incompatibilities of team members e.g., on a personal level, which can be addressed by assigning a high distance to experts who are mutually incompatible and thus should not be put together in the same team.

Note that the upper bound constraints on the team size and the budget can be rewritten as skill constraints and can be incorporated into the skill matrix M accordingly. Thus the team formation problem can be modeled as a generalized densest subgraph problem (5.2). Moreover, the generalized density that we use in the objective of (5.2) leads to further modeling freedom as it enables us to give weights g to the experts according to their expertise. By giving smaller weight to those with high expertise, one can obtain solutions that not only satisfy the given skill requirements but also give preference to the more competent team members (i.e. the ones having smaller weights). Furthermore we allow the skill matrix to have any positive value.

5.7 Experiments

We now empirically show that our method (called here as **FORTE**) consistently produces high quality compact teams. We also show that the quality guarantee given by Theorem 5.6 is useful in practice as our method often improves a given sub-optimal solution.

5.7.1 Experimental Setup

Since we are not aware of any publicly available real world datasets for the team formation problem, we use, as in [47], a scientific collaboration network extracted from the DBLP database. Similar to [47], we restrict ourselves to four fields of computer science: Databases (DB), Theory (T), Data Mining (DM), Artificial Intelligence (AI). Conferences that we consider for each field are given as follows:

[1]The distance function need not satisfy the triangle inequality.

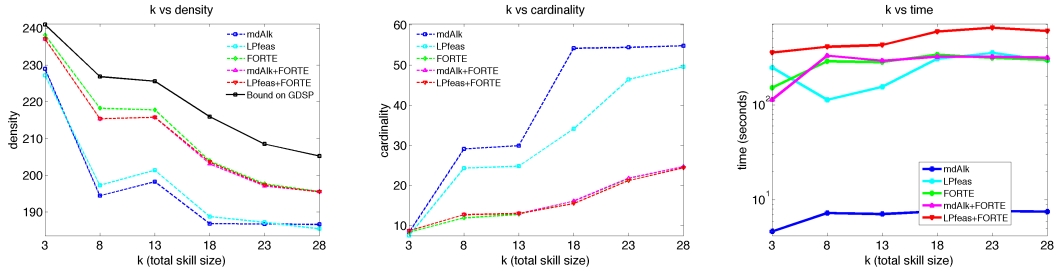


Figure 5.1: Densities and team sizes of **mdAlk**, our method (**FORTE**), a feasible point constructed from the LP (**LPfeas**), and **FORTE** initialized with **LPfeas** and **mdAlk**, averaged over 10 trials. All versions of (**FORTE**) significantly outperform **mdAlk**, and LPfeas both in terms of densities and sizes of the teams found. The densities of **FORTE** are close to the upper bound on the optimum of the generalized densest subgraph problem (GDSP, for short) given by the LP.

DB = {SIGMOD, VLDB, ICDE, ICDT, PODS}, T = {SODA, FOCS, STOC, STACS, ICALP, ESA}, DM = {WWW, KDD, SDM, PKDD, ICDM, WSDM}, AI = {IJCAI, NIPS, ICML, COLT, UAI, CVPR}.

For our team formation problem, the skill set is given by $\mathcal{A}$ ={DB, T, DM, AI}. Any author who has at least three publications in any of the above 23 conferences is considered to be an expert. In our DBLP co-author graph, a vertex corresponds to an expert and an edge between two experts indicates prior collaboration between them. The weight of the edge is the number of shared publications. Since the resulting co-author graph is disconnected, we take its largest connected component (of size 9264) for our experiments.

5.7.2 Quantitative Evaluation

In this section we perform a quantitative evaluation of our method in the special case of the team formation problem with lower bound constraints and $g_i = 1 \, \forall i$ (problem (5.30)). We evaluate the performance of our method against the greedy method proposed in [47], refered to as **mdAlk**. Similar to the experiments of [47], an expert is defined to have a skill level of 1 in skill j, if he/she has a publication in any of the conferences corresponding to the skill j. As done in [47], we create random tasks for different values of skill size, $k = \{3, 8, 13, 18, 23, 28\}$. For each value of k we sample k skills with replacement from the skill set $\mathcal{A}$ = {DB, T, DM, AI}. For example if $k = 3$, a sample might contain {DB, DB, T}, which means that the random task requires at least two experts from the skill DB and one expert from the skill T.

In Figure 1, we show for each method densities and sizes of the solutions found for different skill sizes k, averaged over 10 random runs. In the first plot, we also show the optimal values of the LP relaxation in (5.29). Note that this provides an upper bound on the optimal value of (5.2). We can obtain feasible solutions from the LP relaxation of (5.2) via thresholding (see Section 5.5), which are shown in the plot as **LPfeas**. Furthermore, the plots contain the results obtained when the solutions of **LPfeas** and **mdAlk** are used as the initializations for **FORTE**.

The plots show that **FORTE** always produces teams of higher densities and smaller sizes compared to **mdAlk** and **LPfeas**. Furthermore, **LPfeas** produces

better results than the greedy method in several cases in terms of densities and sizes of the obtained teams. The results of **mdAlk+FORTE** and **LPfeas+FORTE** further show that our method is able improve the sub-optimal solutions of **mdAlk** and **LPfeas** significantly and achieves almost similar results as that of **FORTE** which was started with the unconstrained solution of (5.17). Under the worst-case assumption that the upper bound on the problem (5.2) (abbreviated as GDSP in the plot) computed using the LP is the optimal value, the solution of **FORTE** is $94\% - 99\%$ optimal (depending on k).

5.7.3 Qualitative Evaluation

In this experiment, we assess the quality of the teams obtained for several tasks with different skill requirements. Here we consider the team formation problem (5.2) in its more general setting. We use the generalized density objective of (5.1) where each vertex is given a rank r_i, which we define based on the number of publications of the corresponding expert. For each skill, we rank the experts according to the number of his/her publications in the conferences corresponding to the skill. In this way each expert gets four different rankings; the total rank of an expert is then the minimum of these four ranks. The main advantage of such a ranking is that the experts that have higher skill are given preference, thus producing more competent teams. Note that we choose a relative measure like rank as the vertex weights instead of an absolute quantity like number of publications, since the distribution of the number of publications varies between different fields. In practice such a ranking is always available and hence, in our opinion, should be incorporated.

Furthermore, in order to identify the main area of expertise of each expert, we consider his/her relative number of publications. Each expert is defined to have a skill level of 1 in skill j if he has more than 25% of his/her publications in the conferences corresponding to skill j. As a distance function between authors, we use the shortest path on the *unweighted version* of the DBLP graph, i.e. two experts are at a distance of two, if the shortest path between the corresponding vertices in the unweighted DBLP graph contains two edges. Note that in general the distance function can come from other general sources beyond the input graph, but here we had to rely on the graph distance because of lack of other information.

In order to assess the *competence* of the found teams, we use the list of the 10000 most cited authors of Citeseer [1]. Note that in contrast to the skill-based ranking discussed above, this list is only used in the evaluation and *not* in the construction of the graph. We compute the average inverse rank as in [47] as $AIR := 1000 \cdot \sum_{i=1}^{k} \frac{1}{R_i}$, where k is the size of the team and R_i is the rank of expert i on the Citeseer list of 10000 most cited authors. For authors not contained on the list we set $R_i = 10001$. We also report the densities of the teams found in order to assess their *compatibility*.

We create several tasks with various constraints and compare the teams produced by **FORTE**, **mdAlk** and **LPfeas** (feasible solution derived from the LP relaxation). Note that in our implementation we extended the **mdAlk** algorithm of [47] to incorporate general vertex weights, using Dinkelbach's method from fractional programming [39]. The results for these tasks are shown in Table 1. We report the upper bound given by the LP relaxation, density value, AIR as well as number and sizes of the connected components. Furthermore, we give the names and the Citeseer ranks of the team members who have rank at most 1000. Note

that **mdAlk** could only be applied to some of the tasks and it was not possible to find a feasible team in several cases from the LP relaxation.

As a first task we show the unconstrained solution where we maximize density without any constraints. Note that this problem is optimally solvable in polynomial time and all methods find the optimal solution. The second task asks for at least three experts with the skill DB. Here again all methods return the same team, which is indeed optimal since the LP bound agrees with the density of the obtained team.

Next we illustrate the usefulness of the additional modeling freedom of our formulation by giving an example task where obtaining meaningful, connected teams is not possible with the lower bound constraints alone. Consider a task where we need at least four experts having the skill AI (Task 3). For this, all methods return the same disconnected team of size seven where only four members have the skill AI. The other three experts possess skills DB and DM and are densely connected among themselves. One can see from the LP bound that this team is again optimal. This example illustrates the major drawback of the density based objective which while preferring higher density subgraphs compromises on the connectivity of the solution. Our further experiments revealed that the subgraph corresponding to the skill AI is less densely connected (relative to the other skills) and forming coherent teams in this case is difficult without specifying additional requirements. With the help of subset and distance based constraints supported by **FORTE**, we can now impose the team requirements more precisely and obtain meaningful teams. In Task 4, we require that Andrew Y. Ng is the team leader and that all experts of the team should be within a distance of two from each other in terms of the underlying co-author graph. The result of our method is a densely connected and highly ranked team of size four with a density of 3.89. Note that this is very close to the LP bound of 3.91. The feasible solution obtained by **LPfeas** is worse than our result both in terms of density and AIR. The greedy method **mdAlk** cannot be applied to this task because of the distance constraint. In Task 5 we choose Bernhard Schoelkopf as the team leader while keeping the constraints from the previous task. Out of the three methods, only **FORTE** can solve this problem. It produces a large disconnected team, many members of which are highly skilled experts from the skill DM and have strong connections among themselves. To filter these densely connected members of high expertise, we introduce a budget constraint in Task 6, where we define the cost of the team as the total number of publications of its members. Again this task can be solved only by **FORTE** which produces a compact team of four well-known AI experts. A slightly better solution is obtained when **FORTE** is initialized with the infeasible solution of the LP relaxation as shown (only in this task). This is an indication that on more difficult instances of (5.2), it pays off to run **FORTE** with more than one starting point to get the best results. The solution of the LP, possibly infeasible, is a good starting point apart from the unconstrained solution of (5.17).

Tasks 7, 8 and 9 provide some additional teams found by **FORTE** for other tasks involving upper and lower bound constraints on different skills. As noted in Section 5.5 the LP bound is loose in the presence of upper bound constraints and this is also the reason why it was not possible to derive a feasible solution from the LP relaxation in these cases. In fact the LP bounds for these tasks remain the same even if the upper bound constraints are dropped from these tasks.

5.8 Conclusions

In this chapter we developed a novel method for a generic version of densest subgraph problem based on exact continuous relaxation. We showed how our model based on densest subgraphs allows realistic formulation of the team formation problem in social networks. The main feature of our method is the practical guarantee that our solution always satisfies the given constraints. Furthermore, we derived a linear programming relaxation that allows us to check the quality of solutions found and also provides a good starting point for our non-convex method. Finally we showed in the experiments that our method found qualitatively better teams that are more compact and have higher densities than those found by the greedy method [47].

<table>
<tr><th>Task</th><th>FORTE</th><th>mdAlk</th><th>LPfeas</th></tr>
<tr><td>Task 1: Unconstrained (LP bound: 32.7)</td><td>#Comps: 1 (2) Density: 32.7 AIR: 11.1
Jiawei Han (54), Philip S. Yu (279)</td><td>#Comps: 1 (2) Density: 32.7 AIR: 11.1
Jiawei Han (54), Philip S. Yu (279)</td><td>#Comps: 1 (2) Density: 32.7 AIR: 11.1
Jiawei Han (54), Philip S. Yu (279)</td></tr>
<tr><td>Task 2: DB≥3 (LP bound: 29.8)</td><td>#Comps: 1 (3) Density: 29.8 AIR: 7.56
Jiawei Han (54), Philip S. Yu (279) **(+1)**</td><td>#Comps: 1 (3) Density: 29.8 AIR: 7.56
Jiawei Han (54), Philip S. Yu (279) **(+1)**</td><td>#Comps: 1 (3) Density: 29.8 AIR: 7.56
Jiawei Han (54), Philip S. Yu (279) **(+1)**</td></tr>
<tr><td>Task 3: AI≥4 (LP bound: 16.6)</td><td>#Comps: 3 (1,3,3) Density: 16.6 AIR: 10.3
Michael I. Jordan (28), *Jiawei Han (54)*, Daphne Koller (127), *Philip S. Yu (279)*, Andrew Y. Ng (345), Bernhard Schoelkopf (364) **(+1)**</td><td>#Comps: 3 (1,3,3) Density: 16.6 AIR: 10.3
Michael I. Jordan (28), *Jiawei Han (54)*, Daphne Koller (127), *Philip S. Yu (279)*, Andrew Y. Ng (345), Bernhard Schoelkopf (364) **(+1)**</td><td>#Comps: 3 (1,3,3) Density: 16.6 AIR: 10.3
Michael I. Jordan (28), *Jiawei Han (54)*, Daphne Koller (127), *Philip S. Yu (279)*, Andrew Y. Ng (345), Bernhard Schoelkopf (364) **(+1)**</td></tr>
<tr><td>Task 4: AI≥4, $\text{dist}_G(u,v) \leq 2$, S={Andrew Ng} (LP bound: 3.91)</td><td colspan="2">#Comps: 1 (4) Density: 3.89 AIR: 14.2
Michael I. Jordan (28), Sebastian Thrun (97), Daphne Koller (127), Andrew Y. Ng (345)</td><td>#Comps: 1 (6) Density: 3.5 AIR: 12.5
Michael I. Jordan (28), Geoffrey E. Hinton (61), Sebastian Thrun (97), Daphne Koller (127), Andrew Y. Ng (345), Zoubin Ghahramani (577)</td></tr>
<tr><td>Task 5: AI≥4, $\text{dist}_G(u,v) \leq 2$, S={B.Schölkopf} (LP bound: 6.11)</td><td colspan="3">#Comps: 2 (11,1) Density: 3.54 AIR: 3.94
Jiawei Han (54), *Christos Faloutsos (140)*, Thomas S. Huang (146), *Philip S. Yu (279)*, *Zheng Chen (308)*, Bernhard Schoelkopf (364), *Wei-Ying Ma (523)*, *Ke Wang (580)* **(+4)**</td></tr>
<tr><td>Task 6: AI≥4, $\text{dist}_G(u,v) \leq 2$, S={B.Schölkopf}, $\sum_i c_i \leq 255$ (LP bound: 2.06)</td><td colspan="3">#Comps: 1 (4) Density: 1.24 AIR: 1.82
Alex J. Smola (335), Bernhard Schoelkopf (364) **(+2)**
LP+FORTE: #Comps: 2 (2,2) Density: 1.77 AIR: 2.73
Robert E. Schapire (293), Alex J. Smola (335), Bernhard Schoelkopf (364), Yoram Singer (568)</td></tr>
<tr><td>Task 7: 3≤DB≤6, DM≥10, (LP bound: 11.3)</td><td colspan="3">#Comps: 1 (10) Density: 9.52 AIR: 4.96
Haixun Wang (50), Jiawei Han (54), Philip S. Yu (279), Zheng Chen (308), Ke Wang (580) **(+5)**</td></tr>
<tr><td>Task 8: 2≤DB≤5, 10≤DM≤15, 5≤AI≤10 (LP bound: 10.7)</td><td colspan="3">#Comps: 3 (1,12,3) Density: 7.4 AIR: 5.06
Michael I. Jordan (28), Jiawei Han (54), Daphne Koller (127), Philip S. Yu (279), Zheng Chen (308), Andrew Y. Ng (345), Bernhard Schoelkopf (364), Wei-Ying Ma (523), Divyakant Agrawal (591) **(+7)**</td></tr>
<tr><td>Task 9: AI≤2, T≥2, C≤6 (LP bound: 19)</td><td colspan="3">#Comps: 3 (2,2,2) Density: 6.17 AIR: 1.53
Didier Dubois (426), Micha Sharir (447), *Divyakant Agrawal (591)*, Henri Prade (713), Pankaj K. Agarwal (770) **(+1)**</td></tr>
</table>

Table 5.1: Teams formed by **FORTE**, **mdAlk** and **LPfeas** for various tasks. We list the number and sizes of the found components, the (generalized) maximum density as well as the average inverse rank (AIR) based on the Citeseer list. Finally, we give name and rank of each team member with rank at most 1000. Experts who do not have the skill required by the task but are still included in the team are shown in *italic font*.

Chapter 6

Conclusions

6.1 Summary

In this thesis we presented novel graph-based methods for several problems arising in unsupervised and semi-supervised data analysis. The main contribution of the thesis is the derivation of exact continuous relaxation results for the constrained clustering problem in two-class setting (Chapter 3) and a generic version of the densest subgraph problem (Chapter 5) which has applications in bioinformatics [102] and social network analysis [47]. We showed that the solutions obtained from the exact relaxations are far better than those of the existing methods [64, 79, 121, 117, 47]. Moreover, the exact continuous relaxations allowed us to find a solution that provably satisfies the given prior information, a requirement in some applications [47, 95]. We also showed how to handle soft and hard enforcement settings in a single framework.

We also derived novel methods for unconstrained and constrained clustering problems in the multi-class setting (Chapter 4). The multi-class clustering problem is much more difficult than its two-class counterpart mainly because of the partitioning constraint. In contrast to the existing methods which often fail to produce k-way clustering, our method solves a more generic balanced cut problem and always yields k clusters. We have empirically shown that our method performs better than a wide variety of clustering methods [34, 115, 31, 33, 124, 9, 56, 123, 19]. More importantly, our method, allows easy integration of priors in the multi-class setting unlike the existing work, which fails to incorporate even the simple label constraints. We further showed that our method produces a solution satisfying all the given constraints under the condition that *a* consistent partition can be found efficiently. This includes special cases such as constraints arising from labels or constraints that are mutually exclusive.

On the algorithmic front, we presented efficient methods for solving the continuous relaxations. All our algorithms have monotonic descent guarantee; i.e., the objective values of the iterates produced by our method are monotonically decreasing. Because of this property, our method often improves the solution found by other methods. Moreover, given an initialization consistent with the constraints, our method often produces a solution with strictly better objective while still being consistent. Moreover, we also developed a preconditioning method in Chapter 3 for a generic graph-based convex problem and showed that it drastically improves the

performance of the first-order method FISTA [15].

6.2 Future work

The ideas developed in this thesis can be used to solve related problems such as hypergraph clustering. Hypergraphs are a flexible modeling tool as they encode higher order relationships. In a joint work [57] we already developed a method for hypergraph clustering for the two-class setting based on exact continuous relaxation. Now, one can use the techniques presented in this thesis to incorporate prior knowledge in hypergraph clustering. Moreover, one can similarly develop a direct multi-class method for hypergraph clustering.

In some application problems, e.g., semantic segmentation from image tags [120], the prior information can be formulated as higher order constraints. Thus a potential future direction is designing constrained clustering methods for incorporating higher order constraints.

Alternative clustering [13, 61] is another related problem where one is interested to find a clustering different from the given clustering. One can model the requirement of alternative clustering using must-link and cannot-link constraints [61]. One can now use our constrained clustering methods to solve the alternative clustering problem more effectively since the priors here are hard constraints.

Finally, we would like to point to some open issues. Our algorithms for continuous relaxations although possess monotonic descent guarantees and yield much better results than the convex or spectral relaxations, it is not clear how far they are from the global optimum. Since these problems are NP-hard, it may not be possible to guarantee global optimality in general. However, it maybe possible to show global optimality or bound the gap between the optimum and the objective value of our solution in restricted settings. We have already shown in Chapter 5 that for the generalized densest subgraph problem there exists a linear programming relaxation whose solution provides upper bound on the optimum value (for the maximization problem). This is useful in checking the quality of the solution found by our method. One possible future direction would be deriving such continuous relaxations yielding non-trivial bounds for other problems as well.

There are several future directions possible for the multi-class clustering. Our method developed in Chapter 4 is based on solving a linear programming (LP) problem in each iteration. Although one need not solve it to global optimality, a faster method for solving the LP would speed up the overall algorithm. It would be worthwhile to derive better preconditioning methods, similarly to those developed in Chapter 3, for the special form of LP problems considered here.

As shown in Chapter 3, preconditioning heavily influences the convergence of the first-order optimization method FISTA [15]. One promising direction is generalizing the preconditioning technique developed in Chapter 3 to other first order methods such as PDHG [24, 94].

Bibliography

[1] Citeseer statistics – Most cited authors in computer science. *http://citeseerx.ist.psu.edu/stats/authors?all=true*.

[2] D. Aloise, A. Deshpande, P. Hansen, and P. Popat. NP-hardness of Euclidean sum-of-squares clustering. *Mach. Lear.*, 75(2):245–248, 2009.

[3] S. Amghibech. Eigenvalues of the discrete p-laplacian for graphs. *Ars Combin.*, 67:283–302, 2003.

[4] R. Andersen. Finding large and small dense subgraphs. *CoRR*, abs/cs/0702032, 2007.

[5] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *Ann. IEEE Symp. Found. Comp. Sci. (FOCS)*, pages 475–486, 2006.

[6] R. Andersen and K. Lang. Communities from seed sets. In *Proc. Int. Conf. on World Wide Web (WWW)*, pages 223–232, 2006.

[7] R. Andersen and Y. Peres. Finding sparse cuts locally using evolving sets. In *Proc. Ann. ACM Symp. Theor. Comput. (STOC)*, pages 235–244, 2009.

[8] A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil. A DC-programming algorithm for kernel selection. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 41–48, 2006.

[9] R. Arora, M. R. Gupta, A. Kapila, and M. Fazel. Clustering by left-stochastic matrix factorization. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 761–768, 2011.

[10] F. Bach. Learning with submodular functions: A convex optimization perspective. *Found. Trends Mach. Learn.*, 6(2-3):145–373, 2013.

[11] F. R. Bach and M. I. Jordan. Learning spectral clustering. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 305–312, 2003.

[12] G. D. Bader and C. W. V. Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinform.*, 4, 2003.

[13] E. Bae and J. Bailey. COALA: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *Proc. IEEE Int. Conf. Data Mining (ICDM)*, pages 53–62, 2006.

[14] S. Basu, I. Davidson, and K. Wagstaff. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall, 2008.

[15] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2(1):183–202, 2009.

[16] C.E Bichot and P. Siarry. *Graph Partitioning*. ISTE-Wiley, 2011.

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

[18] Y. Boykov and M.P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *IEEE Int. Conf. on Comp. Vis. (ICCV)*, 2001.

[19] X. Bresson, T. Laurent, D. Uminsky, and J. H. von Brecht. Multiclass total variation clustering. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 1421–1429, 2013.

[20] T. Bühler. *A flexible framework for solving constrained ratio problems in machine learning*. PhD thesis, Saarland University, 2015.

[21] T. Bühler and M. Hein. Spectral clustering based on the graph p-Laplacian. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 81–88, 2009.

[22] T. Bühler, S. S. Rangapuram, S. Setzer, and M. Hein. Constrained fractional set programs and their application in local clustering and community detection. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 624–632, 2013.

[23] V. T. Chakaravarthy, N. Modani, S. R. Natarajan, S. Roy, and Y. Sabharwal. Density functions subject to a co-matroid constraint. In *IARCS Ann. Conf. on Found. of Soft. Tech. and Theor. Comp. Sci. (FSTTCS)*, pages 236–248, 2012.

[24] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. of Math. Imaging and Vision*, 40:120–145, 2011.

[25] K.-C Chang. Variational methods for non-differentiable functionals and their applications to partial differential equations. *J. Math. Anal. Appl.*, 80:102–129, 1981.

[26] F. Chung. A local graph partitioning algorithm using heat kernel pagerank. In *Proc. Int. Work. Alg. Models Web Graph (WAW)*, pages 62–75, 2009.

[27] F.H. Clarke. *Optimization and Nonsmooth Analysis*. Wiley New York, 1983.

[28] I. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: Spectral clustering and normalized cuts. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, pages 551–556, 2004.

[29] I. Dhillon, Y. Guan, and B. Kulis. A unified view of kernel k-means, spectral clustering and graph cuts. Technical Report TR-04-25, University of Texas at Austin, 2004.

[30] I. Dhillon, Y. Guan, and B. Kulis. A fast kernel-based multilevel algorithm for graph clustering. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, pages 629–634, 2005.

[31] I. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, pages 1944–1957, 2007.

[32] C. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *SIAM Int. Conf. Data Mining (SDM)*, pages 606–610, 2005.

[33] C. Ding, T. Li, and M. I. Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *Proc. IEEE Int. Conf. Data Mining (ICDM)*, pages 183–192, 2008.

[34] C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, pages 126–135, 2006.

[35] T. Pham Dinh and H. A. Le Thi. Convex analysis approach to DC programming: Theory, algorithm and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

[36] T. Pham Dinh and H. A. Le Thi. A DC optimization algorithm for solving the trust region subproblem. *SIAM J. Optim.*, 8(2):476–505, 1998.

[37] T. Pham Dinh and H. A. Le Thi. Recent advances in DC programming and DCA. *Trans. Comput. Intell. XIII*, 8342:1–37, 2014.

[38] T. Pham Dinh and E. B. Souad. Duality in DC. (difference of convex functions) optimization. Subgradient methods. *Trends Math. Opt.*, 84(1):277–293, 1988.

[39] W. Dinkelbach. On nonlinear fractional programming. *Manag. Sci.*, 13(7):492–498, 1967.

[40] Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. In *Proc. Int. Conf. on World Wide Web (WWW)*, pages 461–470, 2007.

[41] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[42] A.P. Eriksson, C. Olsson, and F. Kahl. Normalized cuts revisited: A reformulation for segmentation with linear grouping constraints. In *IEEE Int. Conf. on Comp. Vis. (ICCV)*, pages 1–8, 2007.

[43] E. Esser, X. Zhang, and T. F. Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM J. on Imaging Sciences*, 3(4):1015–1046, 2010.

[44] K. Fan. On a theorem of Weyl concerning eigenvalues of linear transformations, I. In *Proc. Nation. Acad. of Sci.*, pages 652–655, 1949.

[45] U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[46] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. of Comp. Vis.*, 59:167–181, 2004.

[47] A. Gajewar and A. D. Sarma. Multi-skill collaborative teams based on densest subgraphs. In *SIAM Int. Conf. Data Mining (SDM)*, pages 165–176, 2012.

[48] M. Girolami. Mercer kernel-based clustering in feature space. *IEEE Trans. on Neural. Netw.*, 13(3):780–784, 2002.

[49] A. V. Goldberg. Finding a maximum density subgraph. Technical Report UCB/CSD-84-171, EECS Department, University of California, Berkeley, 1984.

[50] T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. *SIAM J. on Imag. Sci.*, 2(2):323–343, 2009.

[51] S. Guattery and G. Miller. On the quality of spectral separators. *SIAM J. Matrix Anal. Appl.*, 19:701–719, 1998.

[52] L. Hagen and A. B. Kahng. Fast spectral methods for ratio cut partitioning and clustering. In *Int. Conf. Comput. Aided Design (ICCAD)*, pages 10–13, 1991.

[53] L. Hagen and A. B. Kahng. New spectral methods for ratio cut partition and clustering. *IEEE Trans. Comput. Aided Des.*, 9(11):1074–1085, 1992.

[54] T. Hansen and M. Mahoney. Semi-supervised eigenvectors for locally-biased learning. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 2537–2545, 2012.

[55] M. Hein and T. Bühler. An inverse power method for nonlinear eigenproblems with applications in 1-spectral clustering and sparse PCA. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 847–855, 2010.

[56] M. Hein and S. Setzer. Beyond spectral clustering - tight relaxations of balanced graph cuts. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 2366–2374, 2011.

[57] M. Hein, S. Setzer, L. Jost, and S. S. Rangapuram. The total variation on hypergraphs - learning on hypergraphs revisited. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 2427–2435, 2013.

[58] J.-B. Hiriart-Urruty. Generalized differentiability, duality and optimization for problems dealing with differences of convex functions. In *Convexity and duality in optimization*, pages 37–70, 1985.

[59] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

[60] H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21(1):213–221, 2005.

[61] D. Ian and Q. Zijie. Finding alternative clusterings using constraints. In *Proc. IEEE Int. Conf. Data Mining (ICDM)*, pages 773–778, 2008.

[62] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[63] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 2^{nd} edition, 2002.

[64] S.D. Kamvar, D. Klein, and C.D. Manning. Spectral learning. In *Int. Joint. Conf. on Arti. Intell.*, pages 561–566, 2003.

[65] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103, 1972.

[66] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. on Sci. Comput.*, 20(1):359–392, 1998.

[67] S. Khot. Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4), 2006.

[68] S. Khuller and B. Saha. On finding dense subgraphs. In *Int. Colloq. Autom., Lang. and Programm. (ICALP)*, pages 597–608, 2009.

[69] K. Kiwiel. On linear-time algorithms for the continuous quadratic knapsack problem. *J. Opt. Theor. Appl.*, 134(3):549–554, 2007.

[70] A. Krause and V. Cevher. Submodular dictionary selection for sparse representation. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 567–574, 2010.

[71] A. Krause, B. McMahan, C. Guestrin, and A. Gupta. Robust submodular observation selection. *J. of Mach. Lear. Res. (JMLR)*, 9:2761–2801, 2008.

[72] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *Proc. of the Int. Conf. on World Wide Web (WWW)*, pages 1481–1493, 1999.

[73] H. A. Le Thi and T. Pham Dinh. The DC (difference of convex function) programming and DCA revisited with DC models of real-world nonconvex optimization problems. *Ann. Oper. Res.*, 133(1):23–46, 2005.

[74] H. A. Le Thi, V. N. Huynh, and T. Pham Dinh. DC programming and DCA for general DC programs. *Adv. Comput. Meth. for Knowl. Eng.*, 282:15–35, 2014.

[75] H. A. Le Thi, H.M. Le, V.V. Nguyen, and T. Pham Dinh. A DC programming approach for feature selection in support vector machines learning. *Adv. Data Anal. Classif.*, 2(3):259–278, 2008.

[76] D. D. Lee and H. S. Seung. Unsupervised learning by convex and conic coding. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 515–521, 1996.

[77] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[78] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 556–562, 2000.

[79] Z. Li, J. Liu, and X. Tang. Constrained clustering via spectral regularization. In *IEEE Conf. Comput. Vis. Patt. Recogn. (CVPR)*, pages 421–428, 2009.

[80] H. Lin and J. Bilmes. A class of submodular functions for document summarization. In *Proc. of the Assoc. Comput. Ling.*, pages 510–520, 2011.

[81] S. Lloyd. Least squares quantization in PCM. *IEEE Trans. on Info. Theory*, 28(2):129–137, 1982.

[82] L. Lovász. Submodular functions and convexity. *Math. Program.: The State of the Art*, pages 235–257, 1983.

[83] J. Macqueen. Some methods for classification and analysis of multivariate observations. In 5^{th} *Berkeley Symp. on Math. Stat. and Prob.*, pages 281–297, 1967.

[84] M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k-means problem is NP-hard. In *Proc. of the Int. Work. on Algo. and Comput.*, pages 274–285, 2009.

[85] M. W. Mahoney, L. Orecchia, and N. K. Vishnoi. A local spectral method for graphs: With applications to improving graph partitions and exploring data graphs locally. *J. of Mach. Lear. Res.*, 13(1):2339–2365, 2012.

[86] S. Maji, N. Vishnoi, and J. Malik. Biased normalized cuts. In *IEEE Conf. Comput. Vis. Patt. Recogn. (CVPR)*, pages 2057–2064, 2011.

[87] S. B. Maurer and A. Ralston. *Discrete Algorithmic Mathematics*. CRC Press, 3^{rd} edition, 2005.

[88] M. Narasimhan and J. Bilmes. PAC-learning bounded tree-width graphical models. In *Proc. Conf. Uncert. Art. Intell. (UAI)*, pages 410–417, 2004.

[89] Y. Nesterov. A method of solving a convex programming problem with convergence rate o $(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[90] Y. Nesterov. *Introductory lectures on convex optimization : a basic course*. Kluwer Academic Publishers, 2004.

[91] Y. Nesterov. Gradient methods for minimizing composite objective function. CORE discussion paper, Universit catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2007.

[92] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 849–856, 2001.

[93] K. Pearson. On lines and planes of closest fit to systems of points in space. *Phil. Mag.*, 2(11):559–572, 1901.

[94] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *IEEE Int. Conf. on Comp. Vis. (ICCV)*, pages 1762–1769, 2011.

[95] S. S. Rangapuram, T. Bühler, and M. Hein. Towards realistic team formation in social networks based on densest subgraphs. In *Proc. Int. Conf. on World Wide Web (WWW)*, pages 1077–1088, 2013.

[96] S. S. Rangapuram and M. Hein. Constrained 1-spectral clustering. In *Proc. Int. Conf. Art. Intell. Stat. (AISTATS)*, pages 1143–1151, 2012.

[97] S. S. Rangapuram, P. K. Mudrakarta, and M. Hein. Tight continuous relaxation of the balanced k-cut problem. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 3131–3139, 2014.

[98] R. T. Rockafellar. *Convex analysis*. Princeton University Press, 1970.

[99] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM J. on Control and Opt.*, 14(5):877–898, 1976.

[100] R. T. Rockafellar, R. J.-B Wets, and M. Wets. *Variational analysis*. Springer, 1998.

[101] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268, 1992.

[102] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang. Dense subgraphs with restrictions and applications to gene annotation graphs. In *Int. Conf. on Res. in Comp. Mol. Bio.*, pages 456–472, 2010.

[103] S. Schaible and J. Shi. Fractional programming: Applications and algorithms. *Europ. J. Operat. Res.*, 7(2):111–120, 1981.

[104] S. Schaible and J. Shi. Fractional programming: the sum-of-ratios case. *Optimization Methods and Software*, 18:219–229, 2003.

[105] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

[106] R. Sharan and R. Shamir. Click: A clustering algorithm with applications to gene expression analysis. In *Proc. of the AAAI Conf. on Art. Intell.*, pages 307–316, 2000.

[107] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

[108] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:888–905, 2000.

[109] A. J. Soper, C. Walshaw, and M. Cross. A combined evolutionary search and multilevel optimisation approach to graph-partitioning. *J. of Glob. Opt.*, 29(2):225–241, 2004.

[110] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proc. Ann. ACM Symp. Theor. Comput. (STOC)*, pages 81–90, 2004.

[111] V. Spirin and LA. Mirny. Protein complexes and functional modules in molecular networks. *Proc. of the Nation. Acad. of Sci. USA*, 100:12123–12128, 2003.

[112] A. Szlam and X. Bresson. Total variation and Cheeger cuts. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 1039–1046, 2010.

[113] T. Tao. *Analysis II.* Hindustan Book Agency, 2006.

[114] S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM J. on Opt.*, 20(3):1364–1377, 2009.

[115] U. von Luxburg. A tutorial on spectral clustering. *Stat. and Comput.*, 17:395–416, 2007.

[116] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *Proc. Int. Conf. Mach. Learn. (ICML)*, pages 577–584, 2001.

[117] X. Wang and I. Davidson. Flexible Constrained Spectral Clustering. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining (KDD)*, pages 563–572, 2010.

[118] D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Comput. J.*, 10(1):85–86, 1967.

[119] E. P. Xing, M. I. Jordan, S. J. Russell, and A. Y. Ng. Distance metric learning with application to clustering with side-information. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 521–528, 2003.

[120] J. Xu, A. G. Schwing, and R. Urtasun. Tell me what you see and I will show you where it is. In *IEEE Conf. Comput. Vis. Patt. Recogn. (CVPR)*, 2014.

[121] L. Xu, W. Li, and D. Schuurmans. Fast normalized cut with linear constraints. In *IEEE Conf. Comput. Vis. Patt. Recogn. (CVPR)*, pages 421–428, 2009.

[122] F. Yang and Z. Wei. Generalized Euler identity for subdifferentials of homogeneous functions and applications. *J. Math. Anal. Appl.*, 337:516–523, 2008.

[123] Z. Yang, T. Hao, O. Dikmen, X. Chen, and E. Oja. Clustering by nonnegative matrix factorization using graph random walk. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 1088–1096, 2012.

[124] Z. Yang and E. Oja. Linear and nonlinear projective nonnegative matrix factorization. *IEEE Tran. on Neural. Netw.*, 21(5):734–749, 2010.

[125] S. X. Yu and J. Shi. Grouping with bias. In *Adv. in Neur. Inf. Proc. Syst. (NIPS)*, pages 1327–1334, 2001.

[126] S. X. Yu and J. Shi. Segmentation given partial grouping constraints. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 26(2):173–183, 2004.