Perspectives of Graph Diffusion: Computation, Local Partitioning, Statistical Recovery, and Applications

by

Shenghao Yang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2025

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Satish Rao Professor, Dept. of Electrical Engineering and Computer Sciences, University of California, Berkeley
Supervisor:	Kimon Fountoulakis Associate Professor, School of Computer Science
Internal Member:	Lap Chi Lau Professor, School of Computer Science
Internal Member:	Yaoliang Yu Associate Professor, School of Computer Science
Internal-External Member:	Stephen A. Vavasis Professor, Dept. of Combinatorics and Optimization

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I am the sole author of Chapters 1, 2, 3, and 8 which were not written for publication.

This thesis consists in part of four published conference papers. Exceptions to sole authorship of material are as follows:

Research presented in Chapter 4: Dr. Kimon Fountoulakis and Dr. Di Wang are co-authors on the publication related to Chapter 4 [51]. The authors made equal contributions in the published work. My primary contribution was the design and analysis of the optimization algorithm, conducting empirical experiments, and writing relevant sections of the manuscript. Chapter 4 introduces a new optimization algorithm, its complexity analysis, and an analysis of local graph clustering using approximate solutions, of which I am the sole author.

Research presented in Chapter 5: Dr. Kimon Fountoulakis and Dr. Pan Li are co-authors on the publication related to Chapter 5 [48]. As the lead author of the publication, I was responsible for problem formulation, analysis of local graph clustering, design and analysis of optimization algorithms, conducting empirical experiments, and writing the manuscript.

Research presented in Chapter 6: Dr. Kimon Fountoulakis is an co-author on the publication related to Chapter 6 [136]. As the lead author of the publication, I was responsible for problem formulation, design and analysis of the algorithm, conducting empirical experiments, and writing the manuscript.

Research presented in Chapter 7: Dr. Kimon Fountoulakis and Artur Back de Luca are co-authors on the publication related to Chapter 7 [38]. As the lead author of the publication, I was responsible for problem formulation, design and analysis of the algorithm, conducting empirical experiments on synthetic data, and writing relevant sections of the manuscript.

Abstract

Diffusion describes the process of mass moving from one region to another. In the context of graph, the diffusing mass spreads from nodes to nodes along the edges of the graph. Broadly speaking, this includes a number of stochastic and deterministic processes such as random walk, heat diffusion, network flow and electrical flow on graphs. Graph diffusion is a highly important primitive, and has been shown to have a variety of surprising properties both theoretically and practically. In this thesis, we present several new perspectives of graph diffusion, with an emphasis on how diffusion algorithms uncover the local clustering structure of the input data without necessarily exploring the entire graph.

In the first two parts of the thesis, we introduce a new class of graph diffusion methods that are provably better at extracting the local clustering structure of a graph or a hypergraph. Here, diffusion is formulated as a pair of primal and dual convex optimization problems, based on the idea of spreading mass in the graph while minimizing a p-norm network flow cost. The primal solution of the diffusion problem provides an intuitive physical interpretation where paint (i.e. mass) spills from the source nodes, spreads over the graph, and there is a sink at each node where up to a certain amount of paint can settle. The dual solution embeds the nodes on the non-negative real line and is considered as the output of diffusion. We will show that the dual variables nicely encode the local clustering structure around a given set of seed nodes. In particular, assume the existence of a cluster C of low conductance $\Phi(C)$, the sweep cut procedure on the dual variables returns a cluster $\tilde{C}$ with $\Phi(\tilde{C}) \leq \tilde{O}(\Phi(C)^{1-1/p})$, where $p \geq 2$ is the p-norm objective of the diffusion problem.

In the next two parts of the thesis, we introduce a weighted diffusion mechanism which allows any existing diffusion method to take into account additional node information such as node attributes and labels. The method weighs the edges of the graph based on the attributes or the labels of each node. Depending on the nature and availability of additional node information, two simple yet effective edge-weighting schemes are introduced and analyzed. Over contextual random graphs generated by a local variant of the stochastic block model with noisy node information, we will show that, if the additional information contains enough signal about the ground-truth cluster, then employing existing diffusion algorithms in the weighted graph can more accurately recover the ground-truth cluster than employing diffusion in the original graph without edge weights. In particular, statistical recovery guarantees in terms of precision and F1 score will be derived and compared.

All of the results are supplemented with extensive experiments on both synthetic and real-world data to illustrate the technical results and the effectiveness of the new methods in practice. The code is open-source on GitHub.

Acknowledgements

I owe many thanks to my advisor Kimon Fountoulakis, who always had my best interest at heart and provided me with invaluable guidance through every possible aspect of a successful PhD study. Thank you for being always available to chat and for your time and patience during our countless hours of meetings. I am also very grateful to Lap Chi Lau, Yaoliang Yu, Stephen Vavasis, and Satish Rao for serving as examiners for my thesis.

I would like to thank my collaborators Di Wang, Pan Li, Aseem Baranwal, Amit Levi, Artur Back de Luca, Peihao Wang, and Yunyu Liu for the great time and discussions we had together. Pan Li has also been a great mentor to me. Thank you for your support and trust.

I had a great pleasure spending time with other PhD students, now my friends, from Kimon's group. Aseem and I started around the same time, and we had many chats during the highs and lows of our PhD journey. I really liked his funny jokes. Artur and I shared many rides between Toronto and Waterloo when we had to attend class on campus every Monday and Wednesday. Traveling with Artur had made the monotonous drive in the typical Canadian winter much more pleasant.

Last but not least, I want to thank my dear wife, Xiaoyu, for the love and support, for bringing meaning to all that I do, for feeding me with good food before conference deadlines when I didn't have time to cook, for taking care of our daughter when I was writing this thesis. I couldn't imagine attending my PhD graduation ceremony without her presence.

Dedication

To my wife, Xiaoyu, and our daughter, Lillian. You are the wonders of my world.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Graph diffusion is a generic process in which mass spreads from some nodes to nearby nodes along the edges of the graph. This includes a number of stochastic or deterministic processes such as random walk, heat diffusion, and electrical flow on graphs. Due to its ability to extract useful structural properties of a graph and scalability to work with massive data sets, graph diffusion has a variety of useful applications both theoretically and practically. For example, existing diffusion methods have been explored by theoretical computer scientists for designing efficient graph algorithms [8, 113, 102, 81, 3], by statisticians for statistical network analysis and clustering [13, 60, 59], by machine learning researchers for node embedding, semi-supervised learning, and graph neural networks [118, 101, 70, 63, 53, 126]. In the industry, companies such as Google, Twitter and Pinterest deploy a number of diffusion-based graph clustering and semi-supervised learning algorithms for fraud detection, content discovery, and recommendation applications [130, 42, 43].

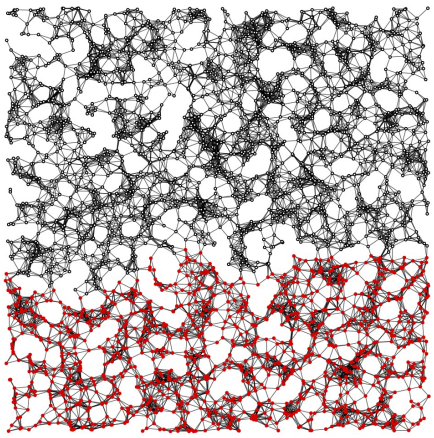
The advancement of modern computing and storage facilities has led to an increasing availability of graph data that are more complex than ever. Below are 3 ways in which graph data are becoming more complex.

1. *Structural noise.* Large-scale graphs in the real world usually do not exhibit a clear high-level structure, while at the same time they contain small-scale sets of nodes that are more meaningful than global sets. An illustrative example is shown in Figure 1.1. Accurately capturing such local graph structures requires algorithms that maintain strong performance against noisy edge connections prevalent in real-world graphs.

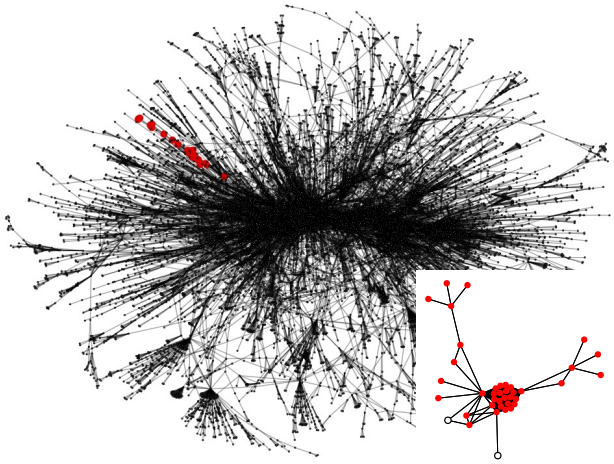
2. *Higher-order interdependence.* In addition to the simple pairwise relationship modeled by the edges in a graph, in many applications, extracting useful structural prop-

erties requires exploiting higher-order interactions among subsets of nodes in the graph. This in turn requires algorithms that can work with higher-order connectivity patterns that model such interactions.

3. *Presence of node attributes and related additional node information.* Heterogeneous data sources consisting of a graph and additional node information in the form of texts, images, or ground-truth node identities are becoming more common. Identifying local structures in such data typically requires leveraging both edge connectivity information and node information. In this case, we need algorithms that can adapt and take into account additional node information.



(a) A random geometric graph partitions into two globally well-balanced pieces.



(b) A typical real-world graph has a classic hairball layout that lacks global structure. (Inset: A zoomed view of a cluster of nodes where the two unfilled nodes are the border with the rest of the graph.)

Figure 1.1: (Adapted from [46].) A geometric graph has a pleasing and intuitive layout in the two-dimensional plane. While most real-world graphs demonstrate little high level structure, they have rich local structures.

In this thesis, new graph diffusion algorithms that apply to each of the above scenarios will be introduced and studied. To characterize the quality of a diffusion method, we will theoretically analyze its ability to extract local clustering structures within large graphs. This is done by analyzing how the output of diffusion correlates with well-defined local

clusters. More precisely, we will analyze how well a diffusion algorithm can be applied to solve the following local graph clustering problem.

Local Graph Clustering: *Given a graph and a set of seed node(s) from that graph, identify a good small cluster of nodes which contains the seed.*

Here, a metric is needed to quantify to what extent a set of nodes constitutes a "good" cluster. Depending on the problem context, reasonable ones include combinatorial metrics such as the conductance (e.g. a lower conductance implies a tightly-knit set of nodes) or statistical metrics such as precision and recall (e.g. when one assumes an underlying statistical model which generates the ground-truth cluster). When a combinatorial metric is used, the resulting problem is often referred to as the local graph partitioning problem in the literature. When a statistical metric is used, the resulting problem is sometimes called a cluster recovery problem. In this thesis, both combinatrial and statistical metrics are considered for the local graph clustering problem, and therefore the terms "local graph partitioning", "local graph clustering" and "local cluster recovery" will be used interchangeably. Therefore, both worst-case and average-case results will be provided. In addition, since the output of a graph diffusion process can often be represented as a very sparse vector, special attention will be paid to analyze the computational properties of diffusion algorithms. In particular, the computational complexity of diffusion algorithms should ideally depend on the size of the output (e.g. number of nonzero entries in the output vector) rather than the size of the input graph. On the application side, for each new diffusion method, an extensive set of empirical experiments on solving semi-supervised learning tasks on graphs will be provided. Compared with the previous start-of-the-art methods, the new methods offer significant performance improvement in most cases.

1.1 Summary of Contributions

The main results of this thesis are presented in four parts. All of the content in this thesis is based on the papers published during my doctoral study with my supervisor and coauthors [51, 48, 136, 38].

1.1.1 p-Norm Flow Diffusion on Graphs

The first part introduces a new class of graph diffusion algorithms based on the idea of diffusing mass while minimizing a network flow cost. In contrast to most existing diffusion algorithms which are defined by the dynamics of the underlying diffusion procedure, i.e.

the step-by-step rules of how to send mass from one node to another, we take a distinct approach and formulate diffusion as an optimization problem over the underlying graph. In this setting, the properties of diffusion are governed by the optimization objective, the dynamics of diffusion is given by the optimization algorithm (e.g. gradient descent), and the output of diffusion is given by the optimal solution of the optimization problem. This decoupling of objective and algorithm provides a natural way to apply the idea of employing the p-norm metric in diffusion, namely adopting the p-norm in the optimization objective. For example, when $p = 2$ the optimization problem includes electrical flows as a special case. For the general p-norm the optimization problem induces different diffusion dynamics which can be shown to lead to improved local graph partitioning guarantees. We will refer to the optimization problem associated with general p-norm as the p-norm flow diffusion problem. Theoretically, two important properties of the p-norm flow diffusion problem are derived. Informally, the properties are summarized as follows:

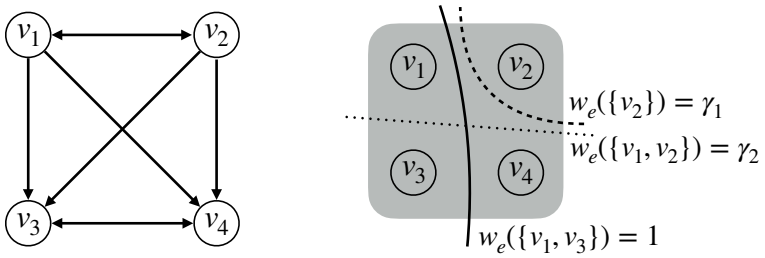
1. Suppose there exists a cluster C with conductance $\Phi(C)$, and we are given a set of seed nodes which overlaps reasonably with C. Then the optimal solution of the p-norm flow diffusion problem can be used to find a cluster $\tilde{C}$ with conductance at most $\tilde{O}(\Phi(C)^{1-1/p})$. A formal statement is provided in Theorem 4.4.

2. A good approximate solution of the p-norm flow diffusion problem can be computed in time that does not depend on the size of the input graph (i.e. number of nodes or edges of the graph). In addition, this approximate solution can be used to achieve the same local clustering performance as above. A formal statement is provided in Theorem 4.8.

Unlike flow-based methods, as a graph diffusion method, p-norm flow diffusion works very well even if we start diffusion from a single seed node. Empirically, we focus on the single seed node setting and demonstrate that p-norm flow diffusion outperforms existing state-of-the-art local diffusion methods for semi-supervised learning tasks over several real-world social and biological networks.

1.1.2 p-Norm Flow Diffusion on Hypergraphs

The second part extends the p-norm flow diffusion problem to hypergraphs which, in addition to consisting of a set of nodes and a set of hyperedges, are associated with submodular cut-cost functions. Hypergraphs generalize graphs by allowing a hyperedge to contain multiple nodes that capture higher-order relationships in complex systems and datasets. In order to explore and understand higher-order relationships in hypergraphs, recent work has



(a) Network motif (b) Hyperedge and cut-cost w_e

Figure 1.2: A food network can be mapped into a hypergraph by taking each network pattern in (a) as a hyperedge [77]. This network pattern captures carbon flow from two preys (v_1, v_2) to two predators (v_3, v_4). (b) is a hyperedge associated with cut-cost w_e that models their relations: w_e is a set function defined over the node set e s.t. $w_e(\{v_i\}) = \gamma_1$ for $i = 1, 2, 3, 4$, $w_e(\{v_1, v_2\}) = \gamma_2$, $w_e(\{v_1, v_3\}) = w_e(\{v_1, v_4\}) = 1$ and $w_e(S) = w_e(e \backslash S)$ for $S \subseteq e$. w_e becomes the unit cut-cost when $\gamma_1 = \gamma_2 = 1$; w_e is cardinality-based if $\gamma_1 = 1/2$ and $\gamma_2 = 1$; more generally, w_e is submodular if $\gamma_1 = 1/2$ and $0 \leq \gamma_2 \leq 1$. The specific choices depend on the application.

made use of cut-cost functions which are defined by associating each hyperedge with a specific set function. These functions assign specific penalties for separating the nodes within individual hyperedges. They generalize the notion of edge cut and have been shown to be crucial for determining small-scale community structure in hypergraphs. The most popular cut-cost functions with increasing capability to model complex multiway relationships are the unit cut-cost, cardinality-based cut-cost, and general submodular cut-cost. An illustration of a hyperedge and the associated cut-cost function is given in Figure 1.2. In the simplest setting, all cut-costs take value either 0 or 1 (e.g., the case where $\gamma_1 = \gamma_2 = 1$ in Figure 1.2b), we obtain the unit cut-cost. In a slightly more general setting, the cut-costs are determined solely by the number of nodes in either side of the hyperedge cut (e.g., the case when $\gamma_1 = 1/2$ and $\gamma_2 = 1$ in Figure 1.2b), we obtain a cardinality-based cut-cost function. We will refer to hypergraphs associated with arbitrary submodular cut-cost functions (e.g., the case where $\gamma_1 = 1/2$ and $0 \leq \gamma_2 \leq 1$ in Figure 1.2b) as general submodular hypergraphs. In this thesis, we introduce the first local diffusion method that can work with general submodular hypergraphs. The contributions are two-fold:

1. Theoretically, a local partitioning result similar to the graph setting will be derived, but for the more complex setting when one deals with general submodular hypergraphs. A formal statement of the result is provided in Theorem 5.6.

2. Empirically, experiments using both synthetic and real-world data show that the new hypergraph diffusion method significantly improves accuracy local clustering and semi-supervised learning problems over hypergraphs with unit, cardinarlity-based and general submodular cut-costs.

1.1.3 Weighted Diffusion in the Presence of Node Attributes

The third part provides an analysis of local diffusion in weighted graphs whose edge weights are defined by the Gaussian kernel of node attributes. Traditionally, local graph diffusion methods are studied primarily in contexts where node attributes are not available. Consequently, the analyses of these algorithms are often concerned with combinatorial properties such as how well the output of diffusion encodes structural properties of a graph. With the increasing availability of multi-modal datasets, it is now very common that a graph dataset contains additional sources of information such as node attributes. In many such cases, edge connections and node attributes jointly determine the underlying clustering structure of the data. So far, there has been no principled local diffusion method that takes into account node attributes. To fill in this gap, we propose a simple graph diffusion algorithm that simultaneously considers both graph structural and node attribute information. It is based on the idea of diffusing mass in a weighted graph which shares the same set of edges as the input graph, but additionally has the edges weighted by the Gaussian kernel of node attributes. An illustration of how edge weights can help the diffusion algorithm uncover the unknown ground-truth target is provided in Figure 1.3.

An analysis on the performance of the weighted diffusion algorithm from a statistical perspective will be provided, under the assumption that the target cluster and the node attributes have been generated from a random data model, akin to a one-sided Contextual Stochastic Block Model (CSBM). Conditions under which the algorithm is guaranteed to fully recover the target cluster with bounded false positives are also derived. Informally, the theoretical results are as follows.

1. *Diffusion with sufficiently good node attributes*: If the node attributes contain sufficiently strong signal about the target cluster, then weighted diffusion can exactly recover the target cluster while incurring zero false positions, as long as the target cluster is connected. A formal statement is provided in Theorem 6.8.

2. *Diffusion with moderately good node attributes*: If the node attributes contain a good amount of signal about the target cluster, then weighted diffusion can fully recover the target cluster, while the number of false positives is bounded by a quantity that

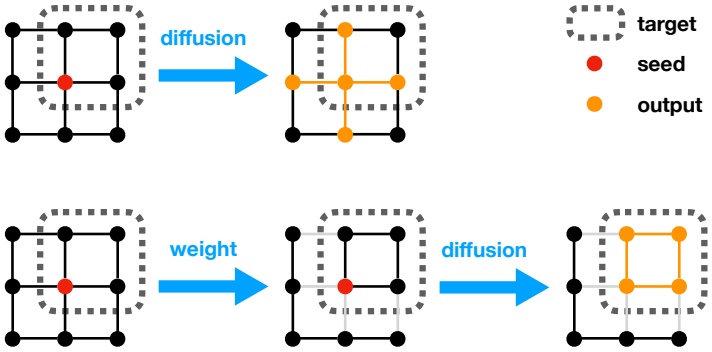


Figure 1.3: Illustration of the difference between weighted and unweighted diffusion over a toy grid graph, where the top right corner of the grid is an unknown target cluster we aim to identify. Observe that the edge weights create a clustering structure in the grid and help diffusion uncover the target. When the boundary edges have small weights, weighted diffusion is forced to spread mass within the target cluster. Because we compute edge weight as Gaussian kernel of node attributes, a small edge weight on the boundary is a direct result of having informative node attributes.

is jointly controlled by the signal from the node attributes and signal from the local graph structure. A formal statement is provided in Theorem 6.9.

1.1.4 Weighted Diffusion in the Presence of Noisy Node Labels

The forth part contains the design and analysis of local diffusion in graphs with corrupted or noisy node labels which indicate whether or not a node belongs to an unknown target cluster. This provides the first rigorous study on how to effectively incorporate noisy node labels in local graph diffusion and improve its performance for cluster recovery. More precisely, we study how local diffusion can be employed to solve the following variant of the local graph clustering problem:

Local Graph Clustering with Noisy Node Labels: Given a graph and a set of seed nodes, the goal is to recover an unknown target cluster around the seed nodes. Suppose that we additionally have access to noisy node labels (not to be confused with ground-truth labels), which are initially set to 1 if a node belongs to the target cluster and 0 otherwise, and then a fraction of them is flipped. How and when can these labels be used to improve clustering performance?

Local graph diffusion with noisy node labels has strong practical motivation. We briefly

discuss the context in the next. Traditionally, local graph diffusion has been studied under a simple, homogeneous context where the only available source of information is the connectivity of nodes, i.e. edges of the graph. There is often little hope that local diffusion can capture a well-connected ground-truth target cluster which also has many external connections. Meanwhile, the emergence of heterogeneous data sources consisting of a graph and any additional node information like texts, images, or ground-truth labels offers new possibilities for improving existing local diffusion algorithms. For example, this additional information can significantly benefit local graph clustering, especially when the graph structure itself does not manifest a tightly knit cluster of nodes. However, little effort has been made to formally investigate the benefits of combining multiple data sources into a diffusion algorithm. While ground-truth label information has been extensively used in (semi-)supervised learning contexts and proved vital in numerous applications, it is neither exploited by existing local diffusion methods, nor analyzed in proper theoretical settings with regard to how and why they can become helpful. In this context, node labels may be viewed as an abstract aggregation of all additional sources of information. The level of label noise, i.e. the fraction of flipped labels, controls the quality of the additional data sources to which we might have access. From a practical point of view, noisy labels may be seen as the result of applying an oracle (e.g. a classifier) which predicts cluster affiliation of a node based on its attributes. The quality of both the oracle and the data we have at hand is thus represented by the label noise in the abstract setting described above.

The weighted diffusion algorithm introduced in this thesis is very simple, yet surprisingly effective in practice. The high-level idea consists of the following two steps. Given a graph G and noisy node labels, first, construct a weighted graph G^w based on the labels, and second, employ local graph diffusion in the weighted graph. The contributions are two-fold:

1. Theoretically, assuming the existence of a ground-truth target cluster in a localized stochastic block model, a recovery guarantee of the weighted diffusion method is provided. In addition, we derive sufficient conditions on the label noise under which, with high probability, employing diffusion over the weighted graph G^w leads to a more accurate recovery of the target cluster than employing diffusion over the original graph G. A formal statement is provided in Theorem 7.3.

2. Empirically, an extensive set of experiments over six attributed real-world graphs indicates that the proposed method, which combines multiple sources of additional information, consistently leads to better local clustering results than existing local methods. More specifically, it will be shown that: (1) reasonably good node labels can be obtained as outputs of a classifier that takes as input the node attributes; (2)

the classifier can be obtained with or without ground-truth label information, and it does not require access to the entire graph; (3) employing diffusion in the weighted graph G^w outperforms both the classifier and diffusion in the original graph G.

Chapter 2

Literature Review

2.1 Graph and Hypergraph Diffusion

Broadly speaking, diffusion on a graph or hypergraph refers to the process in which the diffusing mass spreads from the source nodes to nearby nodes along the edges. Some well-known graph diffusion processes include the random walk process [85, 36] and the related PageRank process [96, 57], where the probability mass spreads from the seed nodes to the rest of the graph, and the heat diffusion process [86, 37], where the heat diffuses from the source nodes through the edges. Usually, a graph diffusion process can be equivalently defined in multiple different ways. For example, the diffusion process that underlies PageRank can be equivalently defined as computing the stationary distribution of a Markov chain [57], finding the solution to a system of linear equations [57], solving an eigenvalue problem [57], and minimizing a convex quadratic function [58, 50]. Of course, the most straightforward way to define a graph diffusion process is to specify the step-by-step rule that instructs how mass should spread from one node to another, e.g., as in [124].

Most existing graph diffusion methods are linear in the sense that their underlying diffusion dynamics is captured by a linear operator in the form of a matrix, such as the transition probability matrix of a random walk, or the Laplacian matrix, or, more generally, the generalized graph diffusion matrix [69, 53] which includes the former two as special cases. Only recently, nonlinear diffusion methods have been proposed and shown to achieve state-of-the-art performance for community detection and semi-supervised learning tasks. Among these, the capacity releasing diffusion [124] is a combinatorially inspired nonlinear diffusion method based on a carefully maintained iterative push-relabel procedure, the nonlinear diffusion method proposed by [63] applies a nonlinear transformation within a

Laplacian matrix-based linear diffusion process and thus turns it into a nonlinear process. The p-norm flow diffusion process that we introduce in Chapter 4 is based on the idea of p-norm network flow and therefore is inherently nonlinear.

In practice, graph diffusion methods are often used for very large graphs consisting of millions of nodes and billions of edges [114]. There is a large body of work on fast computation or approximation of graph diffusion, in particular, PageRank, under different problem settings and computing environments. This includes randomized algorithms [45, 14, 84, 83, 128], local algorithms [8, 7, 23, 125], accelerated and localized optimization algorithms [50, 89], and distributed and parallel algorithms [16, 103, 82, 67]. In this thesis, we focus on the locality property of the diffusion algorithm, similar to [8, 50, 89]. In particular, since we formulate diffusion as an optimization problem, we will require that the corresponding optimization algorithm be able to find a highly accurate solution without having to explore the entire graph, and consequently, the time complexity does not necessarily depend on the size of the graph. This is a crucial property that enables scaling to large graphs with billions of edges.

In contrast to graph diffusion methods, diffusion on hypergraphs has not received a lot of attention until recently. Several recent works focus on generalizing random walk and the PageRank process from graphs to hypergraphs [32, 30, 62, 116]. On the other hand, [76] and [6] study generic variational formulations of hypergraph diffusion, where the output of a diffusion process such as PageRank is the solution of an optimization problem defined over the underlying hypergraph. All these methods require processing the entire hypergraph structure and thus are not scalable. The local diffusion methods proposed in [64] and in [80] do not require access to the entire hypergraph. However, they either cannot work with nontrivial higher-order node relationships, or rely on a reduction to a directed graph and thus can result in poor clustering performance in the worst case. In Chapter 5, we will introduce the first hypergraph diffusion method which not only allows modeling nontrivial higher-order node interactions, but also enjoys superior performance guarantee for the local graph clustering problem.

2.2 Local Graph Clustering

The local graph clustering problem originates from the theoretical computer science community. Given a seed node from a graph, the problem aims to find a small set of nodes that has a small edge conductance [113]. [112, 113] first introduced the problem and proposed a local partitioning algorithm based on truncated random walk processes in the graph. Subsequent to [112], [71] derived an improved approximation guarantee for the random

walk-based local partitioning algorithm. Algorithms for local graph partitioning based on closely related diffusion processes or, more generally, Markov processes over nodes of the graph have also been studied. [8] proposed a local partitioning algorithm using the Approximate Personalized PageRank (APPR) vectors, [37] studied the same problem using heat diffusion, and [100, 9, 54] designed and analyzed evolving set-based algorithms, which remain the currently best known algorithms for the local graph partitioning problem in terms of conductance approximation and computational complexity. Early developments of local graph partitioning algorithms have been mostly motivated by the problem of finding small sets of tightly knit community-like nodes without having to examine the entire graph. The problem has many important applications in the processing of massive graphs. More recently, [9] conjectured that an evolving set-based local graph partitioning algorithm can be used to disprove the small set expansion hypothesis [99], which has important implications for the computational hardness of several problems. The conjecture of [9] was later refuted by [27]. The authors of [27] provided new analyses for both random walk-based and evolving set-based local partitioning algorithms, and described a particularly hard example in which none of the existing local partitioning algorithms would work. In all of these works, a commonly used metric for measuring the quality of a local partitioning algorithm is the edge conductance of the output cluster (or metrics closely related to conductance). These works also focus on a common problem setting for local graph clustering, where one is given a single seed node within the graph and there is no additional assumption on the structural properties of the graph. In this context, all existing algorithms are subject to the quadratic approximation error with respect to conductance, informally known as the Cheeger barrier [72, 27].

On the other hand, if one makes additional assumptions on the structural properties of the graph, then it is possible to obtain improved approximation guarantees (with respect to the edge conductance of the output) for the local graph partitioning problem. For example, assuming good internal connectivity of a target cluster that contains the seed node, [4] improved the approximation guarantee for the PageRank-based local clustering algorithm, which goes beyond the Cheeger inequality. Based on the result of [4], the authors of [95] made a significant further improvement in the approximation guarantee (under the same assumption on the internal connectivity of the target cluster), by combining the PageRank-based algorithm with a flow-based local partitioning algorithm. A similar approximation guarantee was also achieved by [124] under similar structural assumptions on the internal connectivity of the target cluster. The algorithm of [124] relied on a carefully designed iterative diffusion procedure to control the maximum amount of flow over any edge of the graph.

A variant of the local graph partitioning problem is the cut improvement problem [10].

In this problem, instead of a given single seed node, a small set of nodes which already has a certain amount of overlap with an unknown low-conductance cluster is provided as input, and the goal is to find the unknown low-conductance cluster, or a slightly different set of nodes which also has low conductance. In this problem setting, flow-based local algorithms are known to return an output whose conductance is at most $1/\alpha$ times greater than the best possible result, where α is the overlap ratio [10, 95].

Due to its wide range of applications in practice, the local graph clustering problem has received a lot of interest from the machine learning and data mining communities. Unlike in theoretical computer science where the problem setting is often fixed and the focus is theoretical improvement, research work on local graph clustering in these communities is typically motivated by practical applications, and the focus is more on the side of improving empirical performance rather than improving worst-case guarantee. Consequently, local graph clustering algorithms have been proposed and analyzed under different problem settings and assumptions. Some works assume a good initial set of nodes is given, and thus the resulting methods solve (variants of) the cut improvement problem. These include the seed expansion methods [70] and flow-based methods [121, 122, 49], which introduce different objectives to achieve certain desired algorithmic properties for real-world applications. Some works introduce new methods to achieve empirically fast computation time or improved practical performance in terms of clustering accuracy. These include higher-order clustering methods [140], numerical optimization-based methods [46, 50], spectral methods based on heat kernel [69], Lanczos iteration [108], and nonlinear diffusion [63]. The local graph clustering problem has also been generalized to hypergraphs. Existing local clustering methods for hypergraphs include diffusion-based methods such as hypergraph PageRank [116], nonlinear hypergraph diffusion [80], hypergraph capacity-releasing diffusion [64], which can work with a single seed input, and cut-improvement methods such as [119], which require a reference set of nodes as input. Overall, local clustering methods proposed by research work in the machine learning and data mining communities typically achieve state-of-the-art performance in practice for various tasks related to local graph clustering. However, in many cases a rigorous analysis (e.g. without making additional, hard-to-verify assumptions) on how well the conductance of the output of the method can approximate the best conductance is missing.

In Chapter 4 and Chapter 5 when we design and analyze new diffusion algorithms for the local graph clustering problem, we try to find a good balance between theoretical rigor (e.g. from the perspective of a theoretical computer scientist) and practical performance (e.g. from the perspective of a machine learning and data mining researcher). Our overall objective is slightly leaned towards achieving better empirical performances. On the theoretical side, we try to make as few assumptions as possible in the analysis of algorithms.

In particular, our assumptions align with what is typically assumed in the theoretical computer science literature for the cut improvement problem. On the practical side, we provide extensive experiments to empirically demonstrate the superior performance of our methods in real-world applications. Our diffusion algorithms can take either a single seed node or multiple seed nodes as input. However, our theoretical analysis assumes that we are given a set of seed nodes that has good overlap with a low-conductance cluster. In this theoretical setting, flow-based local algorithms [10, 95, 49] achieve a stronger worst-case guarantee than the result we obtain in Chapter 4. However, unlike our method, which is based on diffusion, flow-based methods in practice require a descent amount of initial overlap in order to produce a good output, that is, they cannot work well with a small initial overlap. While, on the other hand, empirically we show that our method outperforms state-of-the-art methods in the single-seed setting. Therefore, there is a gap between what we can prove theoretically and what we can achieve empirically. Developing a rigorous theoretical justification for the good empirical performance of our methods in Chapter 4 and Chapter 5 is an open problem.

2.3 Graph Clustering with Side Information

Real-world graph datasets often contain two types of information. The first is the structure of the graph, that is, edges of the graph which provide relational information between pairs of nodes of the graph. The second, sometimes referred to as side information, is information concerning individual nodes of the graph, such as node attributes and node labels. When multiple sources of information are available, various methods have been proposed to solve problems closely related to (global) graph clustering, where every node of the graph is assigned to one or more subsets of nodes (e.g. communities consisting of nodes that share a common characteristic). These include community detection algorithms for attributed networks [135, 111, 44, 141, 115] and semi-supervised learning methods for graph datasets which contain ground-truth node label information [142, 68, 61]. In the statistics literature, the problem of community detection in attributed random graphs has been studied quite extensively. We will discuss the statistical perspective separately in the next subsection. However, all existing methods require processing the entire graph and all data points, and hence they are not suitable in the context of locally exploring the graph structure from some given seed nodes. So far, the problem of local graph clustering in the presence of side information has received little attention. There is no principled local diffusion algorithm that takes into account additional node information. To fill in this gap, in Chapter 6 and Chapter 7, we present two local diffusion algorithms, each of which will work with a

different common type of node information.

2.4 Statistical Analysis of Clustering on Random Graphs

Recently, contextual random graph models have been used in the machine learning and statistics literature for the analysis of methods and algorithms for attributed graphs. [22, 39, 134, 24, 2, 40] study algorithms for community detection in the (variants of) Contextual Stochastic Block Model (CSBM). [17, 47, 18, 127] analyze the separability of nodes in the CSBM by functions representable by graph neural networks. [131] characterizes the effect of applying multiple graph convolutions on data generated from the CSBM. [129, 19] study optimal node classifiers for the CSBM from Bayesian inference perspectives. There is also a large body of work on the statistical analysis of clustering algorithms in the stochastic block model without the presence of node attributes, and we refer to the nice survey [1] and references therein. The random graph model that we consider in Chapter 6 can be seen as a localized version of the CSBM, but it is more general and we are the first to study statistical performance of a local graph clustering algorithm in contextual random models. On the other hand, unlike prior methods which require processing the entire graph and all data points, since our methods do not examine the entire graph, our theoretical results require stronger assumptions on the connectivity of the graph and the signal strength from node attributes.

The problem of local graph clustering in attributed graphs is related to the statistical problem of anomaly detection [12, 11, 105, 98] and estimation [31]. Anomaly detection aims to decide whether or not there exists an anomalous cluster of nodes whose associated random variables follow a different distribution than those of the rest of the graph. It does not aim to identify the anomalous cluster, and therefore, anomaly detection methods do not apply to local graph clustering. Anomaly estimation aims to locate the anomalous cluster and is more related to problem setting of local graph clustering. However, existing analyses for both anomaly detection and anomaly estimation are restricted to scalar-valued random variables, and the methods rely on computing test statistics or estimators which require processing the entire graph [98, 31]. Our methods differ from anomaly estimation methods in that they do not require processing the entire graph.

Chapter 3

Preliminaries and Notations

This chapter provides technical terminology and notation that we use throughout.

3.1 Sets, Vectors, Functions

For a set S we denote $|S|$ the cardinality of the set. We write 2^S to represent the power set of S. For a subset $S \subseteq P$, the complement of S in P is denoted as $S^{\mathsf{c}} = P \backslash S$. $\mathbb{N}$ denotes the set of natural numbers and $\mathbb{R}$ denotes the set of real numbers. Given a natural number $n \in \mathbb{N}$, we write $[n] = \{1, 2, \ldots, n\}$ the set of integers from 1 to n.

The n-dimensional Euclidean space is denoted by $\mathbb{R}^n$. For a vector $x \in \mathbb{R}^n$, we use x_i to denote its i^{th} element. Given a finite set S and a real-valued function $f : S \to \mathbb{R}$, sometimes we represent f explicitly as a vector $f \in \mathbb{R}^{|S|}$, with the understanding that there is an implicit mapping $S \to [|S|]$, and hence we abuse notation and write $f_i = f(i)$ for $i \in S$. For a subset $T \subseteq S$, we write

$$f(T) = \sum_{i \in T} f(i).$$

Similarly, sometimes we treat a vector $x \in \mathbb{R}^n$ equivalently as a function $x : [n] \to \mathbb{R}$, and we write $x(T) = \sum_{i \in T} x_i$ for $x \in \mathbb{R}^n$ and $T \subseteq [n]$.

The **support** of a vector $x \in \mathbb{R}^n$ or equivalently a function $x : [n] \to \mathbb{R}$ is defined as

$$\text{supp}(x) = \{i \in [n] : x(i) \neq 0\}.$$

Given a subset $S \in [n]$, we use $\mathbf{1}_S$ to denote the **indicator vector** of S, that is

$$\mathbf{1}_S(j) = \begin{cases} 1, & \text{if } j \in S, \\ 0, & \text{otherwise.} \end{cases}$$

With slight abuse of notation, we reserve $e_i = \mathbf{1}_{\{i\}}$ for the i^{th} **canonical basis vector** of n-dimensional Euclidean space $\mathbb{R}^n$, that is,

$$e_i(j) = \begin{cases} 1, & \text{if } j = i, \\ 0, & \text{otherwise.} \end{cases}$$

3.2 Graphs, Matrices, Volume, Cut and Conductance

We consider undirected, connected and weighted **graph** $G = (V, E, w)$, where $V = \{1, 2, \ldots, n\}$ is a set of nodes, $E \subseteq V \times V$ is a set of edges, and $w : E \to \mathbb{R}_+$ assigns each edge $(i, j) \in E$ with a positive weight. If $w(i, j) = 1$ for every $(i, j) \in E$, then G reduces to an unweighted graph, and in this case we simply write $G = (V, E)$. For simplicity we focus on undirected graphs throughout, although the algorithms and results extend to the strongly connected directed case in a straightforward manner.

Given a graph $G = (V, E, w)$, we write $i \sim j$ if $(i, j) \in E$. Given two subsets $S, T \subseteq V$, we denote

$$E(S, T) = \{(i, j) \in E \mid i \in S, j \in T\}$$

the set of edges between S and T. The **degree** of a node $i \in V$ is defined as

$$\deg_G(i) = \sum_{j \sim i} w(i, j).$$

For a subset $S \subseteq V$, the **volume** of S is defined as

$$\mathrm{vol}_G(S) = \sum_{i \in S} \deg_G(i),$$

the **cut-set** and **cut-size** of S are defined as

$$\partial_G(S) = E(S, V \backslash S) = \{(i, j) \in E : i \in S, j \notin S\}, \quad \mathrm{vol}_G(\partial_G(S)) = \sum_{(i,j) \in \partial_G(S)} w(i, j),$$

respectively. The **conductance** of S in G is defined as

$$\Phi_G(S) = \frac{\mathrm{vol}_G(\partial_G(S))}{\min\{\mathrm{vol}_G(S), \mathrm{vol}_G(V \backslash S)\}}.$$

We use subscripts to indicate the graph we are working with, and we omit them when the graph is clear from context.

Given a graph $G = (V, E, w)$, sometimes we ignore the edge weights and work with purely combinatorial notation of degree and volume. In that case, we write

$$\deg^{\circ}_G(i) = |\{j \in V \mid j \sim i\}|$$

to denote the number of neighbors of node i, and correspondingly we define

$$\mathrm{vol}^{\circ}_G(S) = \sum_{i \in S} \deg^{\circ}_G(i).$$

For a matrix $X \in \mathbb{R}^{m \times n}$, we use X_{ij} to denote the element on the i^{th} row and j^{th} column. For a vector $x \in \mathbb{R}^n$, we use $\mathrm{Diag}(x)$ to denote the $n \times n$ diagonal matrix whose diagonal is x. Given a graph $G = (V, E, w)$, we reserve the upper-case letter A to denote the **adjacency matrix** of G, that is, $A_{ij} = w(i, j)$ if $(i, j) \in E$ and $A_{ij} = 0$ otherwise. If the graph is unweighted, then we simply have that $A_{ij} = 1$ if $(i, j) \in E$ and $A_{ij} = 0$ otherwise. We use $B \in \mathbb{R}^{m \times n}$ to denote the unweighted **signed incidence matrix** of G, where $m = |E|$ and the edges are oriented arbitrarily, the row that corresponds to the oriented edge (i, j) has two nonzero entries, with -1 at column i and 1 at column j. Unless otherwise stated, we reserve the lower-case letter d to denote the vector of node degrees, that is, $d_i = \deg(i)$ for each $i \in V$. We use the upper-case letter D to denote the diagonal matrix of node degrees, that is, $D = \mathrm{Diag}(d)$.

We refer to a **function over nodes** $x : V \to \mathbb{R}$ and its explicit representation as an n-dimensional vector $x \in \mathbb{R}^n$ interchangeably. Similarly, we refer to a **function over edges** $y : E \to \mathbb{R}$ and its explicit representation as an m-dimensional vector $y \in \mathbb{R}^m$ interchangeably, where $m = |E|$. Therefore, the edge weight function $w : E \to \mathbb{R}_+$ is treated equivalently as a vector $w \in \mathbb{R}^m$, and we write $w_{ij} = w(i, j)$ for an edge $(i, j) \in E$.

3.3 Hypergraphs, Submodular Functions, Volume, Cut and Conductance

A **hypergraph** $H = (V, E, \mathcal{W})$ is defined by a set of nodes V, a set of hyperedges $E \subseteq 2^V$, i.e. each hyperedge $e \in E$ is a subset of V, and a set of **generalized edge weights** $\mathcal{W} = \{(w_e, \vartheta_e)\}_{e \in E}$. Here, w_e is a function $w_e : 2^e \to \mathbb{R}_+$, and ϑ_e is a positive scalar which plays the role as a normalization factor. For every $e \in E$ and every subset $S \subseteq e$,

the weight $w_e(S)$ indicates the cost of splitting e into to subsets, S and $e \setminus S$. A **proper weight function** w_e should satisfy

$$w_e(\emptyset) = w_e(e) = 0$$

which we will assume throughout. In addition, by defining the normalization factor $\vartheta_e = \max_{S \subseteq e} w_e(S)$ for any function w_e where $w_e(S) > 1$ for some set S, we will assume without loss of generality that we work with normalized w_e such that $w_e(S) \in [0, 1]$ for all $e \in E$ and $S \subseteq e$. To simplify notation we extend the domain of w_e from 2^e to 2^V by identifying $w_e(S) = w_e(S \cap e)$ for every $S \subseteq V$. Because the weight function w_e specifies the cost of splitting the edge e, we will also refer to w_e as the **cut-cost function** associated with e.

Given a set S and a real-valued set function $F : 2^S \to \mathbb{R}$. We say that F is **submodular** if $F(A) + F(B) \geq F(A \cup B) + F(A \cap B)$ for any $A, B \subseteq S$. To ensure computational tractability while allowing to model a rich class of edge weight functions, we require that the function w_e in our definition of hypergraphs to be submodular. This includes nearly all hypergraphs which have been studied in prior work. For example, when $w_e(S) = 1$ for every $e \in E$ and for every $S \in 2^e \setminus \{\emptyset, e\}$, the hypergraph $H = (V, E, \mathcal{W})$ reduces to a unit cut-cost hypergraph [116]. If additionally we have that $\vartheta_e = 1$ for every $e \in E$, then the hypergraph H reduces to an unweighted hypergraph which is completely characterized by its nodes and hyperedges. In this case, we simply write $H = (V, E)$. On the other hand, when $w_e(S)$ only depends on $|S|$, it reduces to a cardinality-based cut-cost hypergraph [120].

Given a hypergraph $H = (V, E, \mathcal{W})$, the **degree** of a node $i \in V$ is defined as

$$\deg_H(i) = \sum_{e \in E : i \in e} \vartheta_e.$$

For $S \subseteq V$, the **volume** of S is

$$\mathrm{vol}_H(S) = \sum_{i \in S} \deg_H(i).$$

The **cut-set** of S is defined as

$$\partial_H(S) = \{e \in E \mid e \cap S \neq \emptyset, e \cap \bar{S} \neq \emptyset\}, \quad \text{where } \bar{S} = V \setminus S.$$

The **cut-size** of S in H is defined as

$$\mathrm{vol}_H(\partial_H(S)) = \sum_{e \in \partial_H(S)} \vartheta_e w_e(S) = \sum_{e \in E} \vartheta_e w_e(S).$$

The **conductance** of S in H is

$$\Phi_H(S) = \frac{\mathrm{vol}_H(\partial_H(S))}{\min\{\mathrm{vol}_H(S), \mathrm{vol}_H(V \setminus S)\}}.$$

Note that our definitions for degree, volume, cut and conductance over hypergraphs are straightforward generalizations of the same terms defined over graphs. For example, given an undirected and weighted graph $G = (V, E, w)$, one can equivalently write it as a hypergraph $H = (V, E, \mathcal{W})$ which has the same sets of nodes V and edges E. To specify the set of edge weight functions $\mathcal{W} = \{(w_e, \vartheta_e)\}_{e \in E}$, for every $e = (i, j) \in E$, one can set $w_e(S) = 1$ is $S = \{i\}$ or $S = \{j\}$ and 0 otherwise, and set $\vartheta_e = w(i, j)$. Then it is easy to see that $H = (V, E, \mathcal{W})$ defines the same graph as $G = (V, E, w)$, and that the definitions for degree, volume, cut and conductance remain the same. In general, our definition of a hypergraph is much broader than our definition of a graph. Even in the case where every hyperedge $e \in E$ contains exactly 2 nodes, our definition of a hypergraph allows to model a directed graph by defining appropriate edge weight functions $\{w_e\}_{e \in E}$. For example, to model the scenario where there is a directed edge (i, j) but no directed edge (j, i), one can simply define $w_e(\{i\}) = 1$ and $w_e(S) = 0$ for any other $S \subseteq e = \{i, j\}$. The resulting w_e is a submodular function and hence satisfies our requirement for w_e.

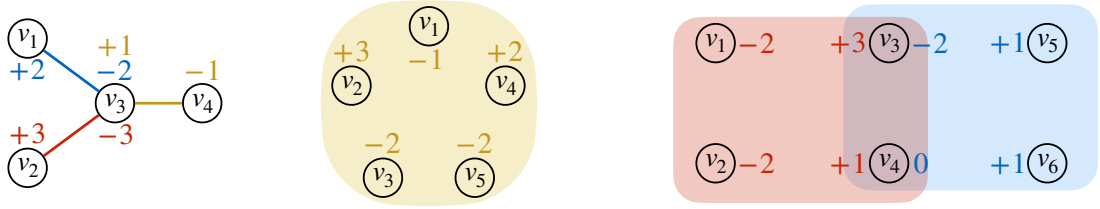
3.4 Flows

Given a graph $G = (V, E, w)$ or a hypergraph $H = (V, E, \mathcal{W})$, a **flow routing** over an edge $e \in E$ is a function $r_e : e \to \mathbb{R}$, where for node $i \in e$, the quantity $r_e(i)$ specifies the amount of mass that flows from $\{i\}$ to $e \backslash \{i\}$ over the edge e. To simplify notation we extend the domain of r_e to V by identifying $r_e(i) = 0$ for $i \notin e$, so r_e is treated as a function $r_e : V \to \mathbb{R}$ or equivalently an n-dimensional vector where $n = |V|$. The net outflow at a node $i \in V$ is given by $\sum_{e \in E} r_e(i)$. Given a routing function r_e and a set of nodes $S \subseteq V$, a directional flow routing on e with direction $S \to e \backslash S$ is represented by

$$r_e(S) = \sum_{i \in e} r_e(i),$$

which specifies the net amount of mass that flows from S to $e \backslash S$ over the edge e. A flow routing r_e is called **proper** if it obeys **flow conservation**, i.e.,

$$r_e(e) = \sum_{i \in e} r_e(i) = 0.$$



(a) Flows on graph (b) Hyperedge routing (c) Flows on hypergraph

Figure 3.1: Illustration of proper flow routings. The numbers next to each node correspond to entries in the flow routing vector r_e over an edge e. We assign the same color to an edge and its associated flow values. Our definition of flow routing is a natural generalization of network flow where $r_e(i) = \pm f$ if and only if $i \in e$, i.e., i is incident to e, where f and the sign determine the amplitude and direction of the flow over e. In Figure 3.1a, the net outflow at node v_3 is given by $\sum_{e \in E} r_e(v_3) = 1 - 2 - 3 = -4$. In Figure 3.1b, the directional flow from $\{v_1\}$ to $\{v_2, v_3, v_4, v_5\}$ over this hyperedge equals -1; similarly, the directional flow from $\{v_1, v_2, v_4\}$ to $\{v_3, v_5\}$ equals $3 + 2 - 1 = 4$, etc. In Figure 3.1c, the net outflow at node v_3 is given by $\sum_{e \in E} r_e(v_3) = 3 - 2 = 1$.

For graphs, because the flow of mass over an edge $e = (i, j)$ always happens between nodes i and j, we always have that $r_e(i) = -r_e(j)$. Therefore, we may simplify the notation further and write $f_e = r_e(i)$ where $|f_e|$ defines the magnitude of flow and $\text{sign}(f_e)$ defines the direction of flow. Consequently, we may use a function $f : E \to \mathbb{R}$ to completely specify a flow routing over the entire set of edges, where $f(i, j)$ specifies the amount of mass that flows from i to j. We abuse the notation to also use $f(j, i) = -f(i, j)$ for an edge $e = (i, j)$. For hypergraphs our definition of flow routing generalizes the notion of network flows to hypergraphs. We provide concrete illustrations in Figure 3.1.

Suppose that each node $i \in V$ is assigned $\Delta(i)$ amount of source mass initially, and we route the mass along edges according to flow routing $\{r_e\}_{e \in E}$. Then by our definition of r_e, the net mass at node i after routing is given by

$$\text{mass}(i) = \Delta(i) - \sum_{e : i \in e} r_e(i) = \Delta(i) - \sum_{e \in E} r_e(i),$$

where the second equality follows from our definition that $r_e(i) = 0$ if $i \notin e$. For graphs, one can write is succinctly as

$$\text{mass}(i) = \Delta(i) - \sum_{(i,j) \in E} f(i, j) = \Delta(i) + \sum_{(i,j) \in E} f(j, i) = [\Delta + B^T f](i)$$

where we abuse the notation and assume the signs in B are absorbed into f via $f(i,j) = -f(j,i)$.

3.5 Limiting Behavior of Sequences and Functions

We use standard notations $O_n, \Omega_n, \Theta_n, o_n, \omega_n$ for the limiting behaviors of a function with respect to n, and we omit the subscript when it is clear from the context. For completeness we provide the definitions below.

For functions $f, g : \mathbb{N} \to \mathbb{R}$, we write $f(n) = O(g(n))$ if there is $n_0 \in \mathbb{N}$ and $M \in \mathbb{R}$ such that $|f(n)| \leq M|g(n)|$ for all $n \geq n_0$. We write $f(n) = \Omega(f(n))$ if there is $n_0 \in \mathbb{N}$ and $m \in \mathbb{R}$ such that $|f(n)| \geq m|g(n)|$ for all $n \geq n_0$. We write $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

For functions $f, g : \mathbb{N} \to \mathbb{R}$, we write $f(n) = o(g(n))$ if for every positive constant $\epsilon > 0$, there is $n_0 \in \mathbb{N}$ such that $|f(n)| \leq \epsilon|g(n)|$ for all $n \geq n_0$. We write $f(n) = \omega(g(n))$ if for every positive constant $c > 0$, there is $n_0 \in \mathbb{N}$ such that $|f(n)| \geq c|g(n)|$ for all $n \geq n_0$.

For functions $f, g : \mathbb{N} \to \mathbb{R}$, we write $f(n) = \text{poly}(g(n))$ if $f(n) = g(n)^{O(1)} = O(g(n)^d)$ for some $d = O(1)$.

3.6 Accuracy Metrics

Depending on the context we may use **false positives**, **false negatives**, **precision**, **recall**, or the **F1 score** as metrics of clustering accuracy. Given a ground-truth set C and an output set (e.g. output of some algorithm) $\tilde{C}$, the number of false positives is $|\tilde{C} \setminus C|$, and the number of false negatives is $|C \setminus \tilde{C}|$. The precision is defined as

$$\text{Precision}(\tilde{C}) = \frac{|\tilde{C} \cap C|}{|\tilde{C}|}.$$

The recall is defined as

$$\text{Recall}(\tilde{C}) = \frac{|\tilde{C} \cap C|}{|C|}.$$

The F1 score is the harmonic mean of precision and recall:

$$\text{F1}(\tilde{C}) = \frac{2}{\text{Precision}(\tilde{C})^{-1} + \text{Recall}(\tilde{C})^{-1}} = \frac{2|\tilde{C} \cap C|}{|\tilde{C}| + |C|}.$$

3.7 Local Graph Diffusion From a Seed Set

We will refer to **local graph diffusion** or simply **graph diffusion** as a generic process in which mass speards from some nodes to nearby nodes along the edges of a graph or a hypergraph. Naturally, to start diffusing mass there has to be some mass in the system. Therefore, to initialize a diffusion process, a set of nodes will receive some source mass to that the mass will then propagate to the rest of the graph. A node is called a **seed node** if it receives a positive amount of source mass. The set of seed node is called a seed set. A local diffusion process consists of multiple diffusion steps, and each diffusion step generally involve the following two steps:

- **Step 1.** Select a node from which mass will spread to its neighbors, according to some selection rule.
- **Step 2.** Send mass from the node to some or all of its neighbors, according to some diffusion rule.

Here, different selection rules in Step 1 and different diffusion rules in Step 2 give rise to different local graph diffusion processes. These rules can affect computational, combinatorial and statistical properties of diffusion. In general, the selection rule in Step 1 can greatly affect the computational efficiency of a local diffusion algorithm, and the diffusion rule in Step 2 can greatly affect how well a diffusion algorithm can extract the local clustering structure around the seed nodes. Given a graph $G = (V, E, w)$ or a hypergraph $H = (V, E, \mathcal{W})$, a set of seed nodes $S \subseteq V$, and optionally information about individual node properties (such as node attributes and node labels), our goal is to design and analyze these rules so that the resulting diffusion processes have provably good computational, combinatorial or statistical properties.

Given a graph $G = (V, E, w)$ or a hypergraph $H = (V, E, \mathcal{W})$, sometimes we call a vector $x \in \mathbb{R}^{|V|}$ **local** is $\mathrm{supp}(x)$ corresponds to a small subgraph in G or H, in which case x itself is a sparse vector. For an algorithm that operates on the graph G or the hypergraph H, we say that the algorithm is **strongly local** or simply **local** if the running time complexity of the algorithm does not depend on the size of G or H.

Chapter 4

p-Norm Flow Diffusion on Graphs

4.1 Diffusion as an Optimization Problem

Given a graph $G = (V, E)$ where $V = \{1, 2, \ldots, n\}$, $E \subseteq V \times V$, and $m = |E|$. Recall that $B \in \mathbb{R}^{m \times n}$ denotes the signed incidence matrix under an arbitrary orientation of the graph G, where the row that corresponds to the oriented edge (i, j) has two nonzero entries, with -1 at column i and 1 at column j. Given two functions $\Delta, T : V \to \mathbb{R}_+$ which specify the source mass and sink capacity of each node, respectively, we propose the following pair of convex optimization problems and refer to them as the p-norm flow diffusion problem:

$$\begin{aligned} &\min_{f} \ \|f\|_p \\ &\text{s.t. } B^T f + \Delta \leq T \end{aligned} \tag{4.1}$$

and its dual formulation with q such that $1/q + 1/p = 1$,

$$\begin{aligned} &\max_{x \geq 0} \ (\Delta - T)^T x \\ &\text{s.t. } \|Bx\|_q \leq 1. \end{aligned} \tag{4.2}$$

The solution to the dual problem $x \in \mathbb{R}^n_+$ gives an embedding of the nodes on the non-negative real line. This embedding is what we actually compute in the context of local clustering. On the other hand, the primal problem and its flow solution $f \in \mathbb{R}^m$ models the physical process of spreading mass while minimizing a p-norm network flow cost. We use the primal problem mostly for insights and analysis purposes. We discuss the interpretation of the primal problem as a diffusion next.

The Primal Problem. As mentioned earlier, we consider a diffusion on a graph $G = (V, E)$ as the task of spreading mass from a small set of nodes to a larger set of nodes. The function Δ will specify the amount of source mass starting at each node, and the function T will give the sink capacity of each node, i.e. the most amount of mass we allow at a node after the spreading. We sometimes refer to the **density** (of mass) at a node i as the ratio of the amount of mass at i over $\deg_G(i)$, and when we use density without any specific node, we mean the maximum density at any node. Naturally in a diffusion, we start with Δ having small support and high density, and the goal is to reach a state with bounded density enforced by the sink function. This gives a clean physical interpretation where paint (i.e. mass) spills from the source nodes and spreads over the graph and there is a sink at each node where up to a certain amount of paint can settle. We note that there is also a "paint spilling" interpretation for the personalized PageRank process where instead of sinks holding paint, the paint dries (and settles) at a fixed rate when it passes through nodes. These are two very different mechanisms on how the mass settles.

For the rest of this section, we will always use the particular sink capacity function, given as

$$T(i) = \deg_G(i), \text{ for all } i \in V.$$

For this particular choice of sink capacity we have

$$T(S) = \sum_{i \in S} \deg_G(i) = \mathrm{vol}_G(S).$$

The total amount of mass in the graph at any time is $\|\Delta\|_1 = \sum_{i \in V} \Delta(i)$, which remains constant throughout the diffusion as flow routing conserves mass. Given initial mass $\Delta \in \mathbb{R}^n$ and flow routing $f \in \mathbb{R}^m$, the vector $m = B^T f + \Delta$ gives the amount of mass $m(i)$ at each node i after spread mass according to the flow routing f, i.e. $m(i) = \Delta(i) + \sum_j f(j, i)$ is the sum of initial source mass and the net amount of mass routed to i. We say f is a **feasible flow routing** for a diffusion problem when $m(i) \leq T(i) = \deg_G(i)$ for all nodes, i.e. the mass obeys the sink capacity at each node. We say i's sink is **saturated** if $m(i) \geq T(i)$ and $\mathrm{excess}(i) = \max(m(i) - T(i), 0)$ the **excess mass** at i. Note there always exists some feasible routing as long as the total amount of mass $\|\Delta\|_1$ is at most $\mathrm{vol}_G(V)$, i.e. there is enough sink capacity over the entire graph to settle all the mass. This will be the case for local diffusion, and we will assume this implicitly through our discussion.

The goal of the p-norm flow diffusion problem is to find a feasible routing with low cost. We consider the p-norm of a routing $\|f\|_p = (\sum_e f_e^p)^{1/p}$ as its cost. For example, when $p = 2$ we can view the flow as electrical current, then the cost is the square root of the energy of the electrical flow; when $p = \infty$ the cost corresponds to the most congested edge's

congestion of the routing. For $p < \infty$, the cost will naturally encourage the diffusion to send mass to saturate nearby sinks before expanding further, and thus our model inherently looks for local solutions.

For reader familiar with the network flow literature, in the canonical p-norm flow problem, we are given the exact amount of mass required at each sink, i.e. the inequality constraint in is replaced by equality, and the high level question is *how* to route mass efficiently from given source(s) to destination(s). For example, if one sets $p = 2$, $\Delta = \mathbf{1}_s$ and $T = \mathbf{1}_t$ for two distinct nodes s, t, then the diffusion problem (4.1) becomes an s-t electrical flow problem. In the p-norm flow diffusion problem, as we have the freedom to choose the destination of mass as long as we obey the sink capacities, the essential question becomes *where* to route the mass so the spreading can be of low cost. In addition, the freedom to choose Δ allows for specialized algorithms which compute optimal solutions much faster, for example, with a total running time that depends only on $\|\Delta\|_1$ rather than $|V| = n$ or $|E| = m$. This is of particular interest in practice for large-scale applications where one cares about computational efficiency. Despite their similarity and close connection, the distinct properties of the p-norm flow diffusion problem poses a novel and meaningful direction to the classic problem of network flow.

The Dual Problem. It is straightforward to check problem (4.2) is the dual of problem (4.1), and strong duality holds for our problems so they have the same optimal value. For the dual problem, we view a solution x as assigning heights to nodes, and the goal is to separate the nodes with source mass (i.e. seed nodes) from the rest of the graph. This is reflected in the objective where we gain by raising nodes with large source mass higher but loss by raising nodes in general. If we consider the height difference between an edge's two endpoints as the *length* of an edge, i.e. $|x(i) - x(j)|$, we constrain the separation of nodes with a budget for how much we can stretch the edges. More accurately, the q-norm of the vector of edge lengths (i.e. $\|Bx\|_q$) is at most 1. This naturally encourages stretching edges along a bottleneck (i.e. cut with small number of edges crossing it) around the seed nodes, since we can stretch each edge more when the number of edges is smaller (and thus raise seed nodes higher). The dual problem also inherently looks for local solutions as raising nodes far away from the source mass only hurts the objective.

Sparsity. As mentioned in the above, both the primal and the dual problems inherently look for local solutions. Consequently, a distinct property of the p-norm flow diffusion problem is that the optimal solutions for both the primal and the dual problems are sparse. More precisely, the number of nonzero entries in the solutions $x \in \mathbb{R}^n$ and $f \in \mathbb{R}^m$, where $n = |V|$ and $m = |E|$, does not depend on either n or m. Lemma 4.1 says that the size of the support of the optimal solutions is bounded by the total amount of source mass $\|\Delta\|_1$. This is an important property as later we will exploit the locality of solution and devise

an algorithm which computes the solution in time $O(\text{ploy}(\|\Delta\|_1))$.

Lemma 4.1 (Sparsity of solutions for (4.1) and (4.2))**.** *Let f^* and x^* be optimal solutions of (4.1) and (4.2) respectively, where $T(i) = \deg(i)$ for all i. We have that*

1. $|\text{supp}(f^*)| \leq \|\Delta\|_1$;
2. $\text{vol}(\text{supp}(x^*)) \leq \|\Delta\|_1$;
3. $x^*(i) > 0$ *only if* $(B^T f^* + \Delta)(i) = \deg(i)$.

Proof. For $p < \infty$, the optimal routing will never push mass out of a node i unless i's sink is saturated, i.e. $f^*(i,j) > 0$ for i, j only if $(B^T f^* + \Delta)(i) = \deg(i)$. To see this, take the optimal primal solution f^*, and consider the decomposition of f^* into flow paths, i.e., the path that the diffusion solution used to send each unit of mass from its source node to the sink at which it settled. If any node other than the last node on this path has remaining sink capacity, we can truncate the path at that node, and strictly reduce the total cost of the diffusion solution. As each unit of mass is associated with the sink of one node, the total amount of mass $\|\Delta\|_1$ upper-bounds the total volume of the saturated nodes since it takes $\deg(i)$ amount of mass to saturate the node i. This observation proves the first claim.

The dual variable $x^*(i)$ corresponds to the primal constraint $(B^T f^* + \Delta)(i) \leq \deg(i)$, and it is easy the check the third claim of the lemma is just complementary slackness. The second claim follows from the first and the third claim. □

4.2 Local Graph Clustering

In this section we discuss the optimal solutions of our diffusion problem (and its dual) in the context of local graph clustering. At a high level, we are given a set of seed nodes S and want to find a cut of low conductance close to these nodes. In practice, minimizing conductance or other similar ratio cut measures often serve as the proxy for finding interesting local clusters when the structure is community-like, i.e. tightly-knit cluster of nodes. Following prior work in local clustering, we formulate the goal as a promise problem, where we assume the existence of an unknown good cluster $C \subset V$ with $\text{vol}(C) \leq \text{vol}(V)/2$ and conductance $\Phi(C)$. We consider a generic fixed $G = (V, E)$ and $p \in (1, \infty)$ throughout our discussion.

4.2.1 Diffusion Setup

To specify a particular diffusion problem and its dual, we need to provide the source mass Δ, and recall we always set the sink capacity $T(i) = \deg(i)$. Given a set of seed nodes S, we pick a scalar δ and let

$$\Delta(i) = \begin{cases} \delta \cdot \deg(i), & \text{if } i \in S, \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

Note this gives the total amount of mass $\|\Delta\|_1 = \delta \cdot \mathrm{vol}(S)$.

Now we discuss how to set δ. The intuition is that we want the total amount of source mass starting in C to be a constant factor larger than the volume of C, say $\Delta(C) \geq 2\mathrm{vol}(C)$ (any constant reasonably larger than 1 would work). The reason is that in such scenario, at least $\Delta(C) - \mathrm{vol}(C) \geq \mathrm{vol}(C)$ amount of mass has to be routed out of C since the nodes in C have total sink capacity $\mathrm{vol}(C)$. When C is a cut of low conductance, any feasible routing must incur a large cost since $\mathrm{vol}(C)$ amount of mass has to get out of C using a relatively small number of discharging edges. In this case, the optimal dual solution x^* will certify the high cost of any feasible primal solution. Naturally, the appropriate value of δ to get $\Delta(C) \geq 2\mathrm{vol}(C)$ depends on how well the seed set S overlaps with C. Suppose $\mathrm{vol}(S \cap C) \geq \alpha \mathrm{vol}(C)$, then we can set $\delta = 2/\alpha$. Without loss of generality, we assume the right value of δ is known since otherwise we can employ binary search to find a good value of δ.

More formally, we derive a low conductance cut from x^* using the standard sweep cut procedure. According to Lemma 4.1, because x^* has bounded support, i.e. $\mathrm{vol}(\mathrm{supp}(x^*)) \leq \|\Delta\|_1 = \delta \cdot \mathrm{vol}(S)$, the sweep cut procedure can be implemented in time $O(\delta \cdot \mathrm{vol}(S))$.

The Sweep Cut Procedure

1. Sort the nodes in decreasing order by their values in x^*.
2. For each $i \geq 1$ such that the i-th node still has strictly positive dual value, consider the cluster C_i containing the first i nodes, i.e., $C_i = \{j : x^*_j \geq x^*_i\}$. Among all these cuts (also called *level cuts*) output the one with the smallest conductance.

4.2.2 Local Clustering Guarantee

Our analysis assumes that the seed set S has a reasonable overlap with the target C.

Assumption 4.2. The seed set $S \subset V$ satisfies

1. $\text{vol}(S \cap C) \geq \alpha \text{vol}(C)$ for some $\alpha \in (0, 1]$;
2. $\text{vol}(S \cap C) \geq \beta \text{vol}(S)$ for some $\beta \in (0, 1]$.

Note that the sweep cut computation only requires the dual solution x^*, while the primal solution f^* and the values of α, β are only for analysis. Recall we want to set $\delta = 2/\alpha$ in (4.3) to formulate the dual problem, but we assume δ is known via binary search. We also assume the entire graph is larger than the total amount of source mass so the primal is feasible and the dual is bounded.

Assumption 4.3. The source mass Δ as specified in (4.3) satisfies $\delta = 2/\alpha$, which gives $\Delta(C) \geq 2\text{vol}(C)$ and $\|\Delta\|_1 = \Delta(S) \leq 2\text{vol}(C)/\beta < \text{vol}(V)$.

Theorem 4.4. *Under Assumption 4.2 and Assumption 4.3, the cluster C^* returned by the sweep cut procedure on the optimal solution x^* of Problem (4.2) satisfies*

$$\Phi(C^*) \leq \frac{6\Phi(C)^{1/q}}{\alpha\beta}.$$

Not surprisingly, our theorem is more meaningful when the given seed set S has a good overlap with some low conductance cut C, i.e. when α, β are bounded away from 0. In particular, suppose α, β are both at least $\frac{1}{\log^t(\text{vol}(C))}$ for some constant t, then the bound in our theorem becomes $\tilde{O}(\Phi(C)^{1/q})$, where we follow the tradition of using $\tilde{O}$ to hide poly-logarithmic factors. In particular, for 2-norm diffusion (i.e. $p = q = 2$) this matches the bound achieved by spectral and random walk based methods in this setting, and for p-norm diffusion with p approaches ∞ (i.e. q tends to 1), this matches the guarantees of previous flow based methods in this setting, e.g. [95, 124]. For any finite p, the bound in Theorem 4.4 is weaker than what is achievable by flow based methods. However, those methods either require a large initial overlap to produce any meaningful output in practice [95, 49], or require additional assumption on the internal connectivity of the target cluster [124]. On the contrary, as we will show in Section 4.4, our diffusion method works surprisingly well even with a single seed node. Therefore, we believe that Assumption 4.2 is an artifact required by our analysis. Empirically, our results indicate

that an $O(\Phi(C)^{1/q})$ approximation might be achievable by starting the diffusion from a good, single seed node. Proving such a result is an interesting open problem.

The proof of Theorem 4.4 is relatively short, and we provide the details in the rest of the section. We start with a simple lemma stating that the objective value of the optimal dual (and primal) solution must be large. Recall that $\partial(C) = \{(i,j) \in E : i \in C, j \notin C\}$, and $d \in \mathbb{R}^n$ is the vector of node degrees, i.e., $d(i) = \deg(i)$ for all i.

Lemma 4.5. *We have* $(\Delta - d)^T x^* = \|f^*\|_p \geq \mathrm{vol}(C)/|\partial(C)|^{1/q}$.

Proof. The equality simply follows from the strong duality between Problem (4.1) and Problem (4.2). To see the lower bound, note that by Assumption 4.3 there is least $\Delta(C) \geq 2\mathrm{vol}(C)$ amount of source mass trapped in C at the beginning. Since the sinks of nodes in C can settle $\mathrm{vol}(C)$ amount of mass, the remaining at least $\mathrm{vol}(C)$ amount of excess needs to get out of C using the $|\partial(C)|$ cut edges. We focus on the cost of f^* restricted to these edges alone. Since $p > 1$, the cost is the smallest if we distribute the routing load evenly on the $|\partial(C)|$ edges, and it is simple to see this incurs cost

$$\left(|\partial(C)| \cdot \left(\frac{\mathrm{vol}(C)}{|\partial(C)|}\right)^p\right)^{1/p} = \mathrm{vol}(C)/|\partial(C)|^{(p-1)/p}.$$

The total cost of f^* must be at least the cost incurred just by routing the excess out of C. □

Recall that we define the length of an edge $e = (i,j)$ to be $l(e) = |x^*(i) - x^*(j)|$. The actual dual solution may incur edges with tiny non-zero length which causes difficulties in the analysis. Thus, we define the following perturbed edge length so that any non-zero edge length is at least $1/\mathrm{vol}(C)^{1/q}$. Note this is only for analysis purpose and doesn't require changing x^*.

$$\tilde{l}(e) = \begin{cases} \max\left(\frac{1}{\mathrm{vol}(C)^{1/q}}, l(e)\right) & \text{if } l(e) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{4.4}$$

Note the constraint in the dual problem gives

$$\sum_{e=(i,j)} |x^*(i) - x^*(j)| \cdot l(e)^{q-1} = \sum_{e=(i,j)} |x^*(i) - x^*(j)|^q = \|Bx^*\|_q^q \leq 1.$$

The next lemma states that the perturbations on edges lengths are small enough so the above quantity remains small.

Lemma 4.6. $\sum_{e=(i,j)} |x^*(i) - x^*(j)| \cdot \tilde{l}(e)^{q-1} \leq 1 + \frac{2}{\beta}$.

Proof.

$$\sum_{e=(i,j)} |x^*(i) - x^*(j)| \cdot \tilde{l}(e)^{q-1} \leq \sum_{e=(i,j)} \tilde{l}(e)^q = \sum_{e:l(e)=\tilde{l}(e)} l(e)^q + \sum_{e:l(e)<\tilde{l}(e)} \frac{1}{\text{vol}(C)} \leq 1 + \frac{2}{\beta}.$$

The second to last equality follows from the fact that our perturbation only increases the lengths to $1/\text{vol}(C)^{1/q}$. The last inequality follows from that we only increase the length of an edge when its original edge is positive, which means at at least one of its endpoints has positive dual variable value. From Assumption 4.3 that $\delta = 2/\alpha$ we know that the total amount of mass is at most $\frac{2}{\alpha}\text{vol}(S)$. Together with the conditions in Assumption 4.2 we get $\frac{2}{\alpha}\text{vol}(S) \leq \frac{2}{\beta}\text{vol}(C)$. This upper-bounds $\text{vol}(\text{supp}(x^*))$ by Lemma 4.1. Thus, the number of edges with positive $l(e)$ is also at most $\frac{2}{\beta}\text{vol}(C)$. □

Consider the sweep cut procedure where we order the nodes by their dual values in x^*, and for any $h > 0$ denote the level cut

$$S_h = \{i \in V : x^*(i) \geq h\}$$

to be the set of nodes with dual value larger than h. We only need to consider S_h when h equals to the strictly positive dual value of some node in the support of x^*, and the sweep cut procedure will examine all such cuts. We proceed to argue that among these level cuts, there must exists some h where $\Phi(S_h)$ satisfies the bound in Theorem 4.4, and thus proving Theorem 4.4.

We start with rewriting the dual objective and constraint using the level cuts.

Claim 4.7. *With level cuts S_h as defined above, we have*

$$(\Delta - d)^T x^* = \int_{h=0}^{\infty} (\Delta(S_h) - \text{vol}(S_h))\, dh,$$

and

$$\sum_{e=(i,j)} |x^*(i) - x^*(j)| \cdot \tilde{l}(e)^{q-1} = \int_{h=0}^{\infty} \sum_{e \in \partial(S_h)} \tilde{l}(e)^{q-1} dh.$$

Proof. Both equations follow from changing the order of summation. To see first equation, pick any node $i \in V$. Node i contributes $(\Delta(i) - \deg(i)) \cdot x^*(i)$ to the left hand side, and the same amount to the right hand side as i is in the level cuts for all $h \in (0, x^*(i)]$. For the

second equation, pick any edge $e = (i, j)$. The edge belongs to $\partial(S_h)$ for all $h \in (x^*(j), x^*(i)]$ (assuming without loss of generality that $x^*(i) \geq x^*(j)$), so the contribution from any edge will be the same to both sides of the equation. □

Using Claim 4.7 and invoke Lemma 4.5 and Lemma 4.6, we take the ratio to get

$$\frac{\int_{h=0}^{\infty} \sum_{e \in \partial(S_h)} \tilde{l}(e)^{q-1} dh}{\int_{h=0}^{\infty} (\Delta(S_h) - \mathrm{vol}(S_h))\, dh} \leq \frac{3|\partial(C)|^{1/q}}{\beta \mathrm{vol}(C)},$$

which means there must exist some h with S_h non-empty and

$$\frac{\sum_{e \in \partial(S_h)} \tilde{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)} \leq \frac{3|\partial(C)|^{1/q}}{\beta \mathrm{vol}(C)}. \tag{4.5}$$

All it remains is to connect the left hand side in the above inequality to the conductance of S_h. For the denominator, since the source mass has density at most $\delta = 2/\alpha$ at any node, we get

$$\Delta(S_h) - \mathrm{vol}(S_h) \leq \frac{2}{\alpha} \mathrm{vol}(S_h). \tag{4.6}$$

For the numerator, any edge $e = (i, j)$ crossing a level cut S_h must have dual values $x^*(i), x^*(j)$ on different sides of h, thus having non-zero length $l(e)$, which means $\tilde{l}(e)$ is at least $1/\mathrm{vol}(C)^{1/q}$. This gives

$$\sum_{e \in \partial(S_h)} \tilde{l}(e)^{q-1} \geq \frac{|\partial(S_h)|}{\mathrm{vol}(C)^{(q-1)/q}}. \tag{4.7}$$

Put (4.5), (4.6) and (4.7) together we get

$$\Phi(S_h) = \frac{|\partial(S_h)|}{\mathrm{vol}(S_h)} \leq \frac{6|\partial(C)|^{1/q}}{\alpha\beta \mathrm{vol}(C)^{1/q}} = \frac{6\Phi(C)^{1/q}}{\alpha\beta},$$

which proves Theorem 4.4.

4.3 Optimization Algorithm and Complexity

So far we have only talked about how the dual solution x^* can be used for local clustering. To actually perform local clustering in a given graph, we still need an algorithm to compute x^*, either exactly (i.e. analytically) or approximately. It turns out that an analytic

solution for Problem (4.2) does not exist in general, and thus we need to devise an optimization algorithm and solve Problem (4.2) within some desired approximation error. In this section we provide a local algorithm which outputs an approximate solution $\tilde{x}$ in time $O(\text{poly}(\|\Delta\|_1))$. In addition, we show that if one uses the same diffusion setup as described in Theorem 4.4, then the sweep cut procedure on the approximate solution $\tilde{x}$ returns a cluster $\tilde{C}$ whose conductance $\Phi(\tilde{C})$ satisfies the same bound (up to a constant factor) as in Theorem 4.4.

The constrained formulation of the dual problem (4.2) prohibits direct application of efficient first-order optimization methods. Therefore, we consider the following equivalent formation which moves the q-norm constraint into the objective function:

$$\min_{x \geq 0} \ \frac{1}{q}\|Bx\|_q^q + x^T(d - \Delta). \tag{4.8}$$

It is straightforward to verify that the above dual formulation corresponds to replacing the primal objective function $\|f\|_p$ with $\frac{1}{p}\|f\|_p^p$:

$$\begin{aligned} \min_{f} \ & \frac{1}{p}\|f\|_p^p \\ \text{s.t.} \ & B^T f + \Delta \leq d. \end{aligned} \tag{4.9}$$

In addition, up to scaling the solutions of Problem (4.8) and Problem (4.2) are the same. Since the sweep cut procedure is invariant under scaling, we consider Problem (4.8) and Problem (4.2) as equivalent.

Our algorithm is obtained by applying a projected random permutation coordinate descent method. Due to d_i may not be the same as d_j if $i \neq j$, the coordinate-wise Hölder smoothness property of the objective function of Problem (4.8) is not uniform. Therefore, we apply the coordinate descent method to the following function $F(z)$ over $z \geq 0$:

$$F(z) := \frac{1}{q}\|BD^{-1/q}z\|_q^q + z^T D^{-1/q}(d - \Delta). \tag{4.10}$$

The resulting algorithm is provided in Algorithm 1. As we have intentionally written the algorithm in a way to represent what one should implement in practice in order to maintain locality, the reader may not immediately recognize Algorithm 1 as a "projected" and "random permutation" coordinate descent method. Let us briefly explain this. In Algorithm 1, maintaining the active set A^k and updating only the coordinates in A^k is equivalent to updating all coordinates and then applying a projection step to ensure the iterate $z^{k+1} \geq 0$. In addition, we do not require any specific order based on which the coordinates in A^k

Algorithm 1 Projected Random Permutation Coordinate Descent for Problem (4.8)

Initialize: $z^0 = 0$
For $k = 0, 1, 2, \ldots, K$ **do**
  Set $A^k = \{i \in V \mid \nabla_i F(z^k) < 0\}$.
  Set $y^k = z^k$.
  For each $i \in A^k$ **do**
    Update $y^k = y^k + \left(\frac{|\nabla_i F(y^k)|}{2}\right)^{p-1} e_i$.
  Set $z^{k+1} = y^k$.
Return $D^{-1/q} z^K$.

should be updated, it turns out that the ordering of coordinates in A^k can be arbitrary, and thus the term "random permutation". Finally, we note that the gradient vector $\nabla F(y^k)$ in Algorithm 1 can be updated incrementally after every coordinate update. In particular, using Equation (4.11), it is straightforward to see that, after updating the i^{th} coordinate value in y^k, we need to update the j^{th} coordinate value in $\nabla F(y^k)$ accordingly only if $j \sim i$ or $j = i$. Therefore, Algorithm 1 can be implemented to maintain locality. That is, that total number of coordinates that are ever required by Algorithm 1 for computation is upper bounded by $\text{vol}(z^K)$. In Theorem 4.10, we provide a natural and easy-to-check termination condition for Algorithm 1, which is related to the ℓ_1-norm of the gradient vector $\nabla F(z^k)$.

For the rest of this section we denote

$$z^* = \operatorname*{argmin}_{z \geq 0} F(z)$$

where the definition of $F(z)$ is provided in (4.10), and we denote

$$x^* = \operatorname*{argmin}_{x \geq 0} \frac{1}{q} \|Bx\|_q^q + x^T(d - \Delta).$$

It is easy to see that $x^* = D^{-1/q} z^*$. We use z^k to denote the k^{th} iterate generated by Algorithm 1, and correspondingly write $x^k = D^{-1/q} z^k$.

Theorem 4.8. *Assume the same diffusion setup as considered in Theorem 4.4. Then for $K = O(\text{vol}(C)^p / \beta^{2p})$, the cluster $\tilde{C}$ returned by the sweep cut procedure on the output of Algorithm 1 (i.e. $x^K = D^{-1/q} z^K$) satisfies (recall that $1/p + 1/q = 1$)*

$$\Phi(\tilde{C}) \leq O\left(\frac{\Phi(C)^{1/q}}{\alpha\beta}\right).$$

Note that Algorithm 1 updates a node i only if $z^*(i) > 0$. Moreover, for every $k \geq 0$, constructing the active set A^k and computing all the required coordinate-wise gradient $\nabla_i F(z^k)$ only need access to a node j if $j \sim i'$ for some $i' \in \text{supp}(z^*)$. The means that each iteration of Algorithm 1 can be executed in time

$$O(\text{vol}(\text{supp}(z^*))) = O(\|\Delta\|_1) = O(\text{vol}(C)/\beta),$$

where the last equality follows from our diffusion setup and Assumption 4.3 that $\|\Delta\|_1 \leq 2\text{vol}(C)/\beta$. Therefore, we obtain the following result as a direct consequence of Theorem 4.8.

Corollary 4.9. *Assume the same diffusion setup as considered in Theorem 4.4. With total running time $O(\text{vol}(C)^{p+1}/\beta^{2p+1})$, Algorithm 1 returns an approximate solution $\tilde{x}$ of Problem (4.8). The cluster $\tilde{C}$ returned by the sweep cut procedure on $\tilde{x}$ satisfies*

$$\Phi(\tilde{C}) \leq O\left(\frac{\Phi(C)^{1/q}}{\alpha\beta}\right).$$

Theorem 4.8 and Corollary 4.9 shows a natural tradeoff between the computational complexity for finding an approximate dual solution and the performance guarantee for local graph clustering. We leave the details of proof to Section 4.5.2. Essentially, Theorem 4.8 follows from combining a local clustering argument similar to the one discussed in Section 4.2.2 and the following rate of convergence in gradient norm for Algorithm 1.

Theorem 4.10. *Let $\epsilon > 0$. Let k be such that*

$$k \geq 2^{p-1}\left(\frac{\|\Delta\|_1}{\epsilon}\right)^p.$$

Then the k^{th} iterate z^k generated by Algorithm 1 satisfies

$$\|D^{1/q}[-\nabla F(z^k)]_+\|_1 \leq \epsilon\|\Delta\|_1,$$

where $[a]_+$ denotes $\max\{a, 0\}$ where the maximum is taken entry-wise.

Rather than using standard metrics such as distance to the optimal function value or distance to the optimal solution to measure convergence, in Theorem 4.10 we show convergence in terms of the ℓ_1-norm of the negative gradients scaled by $D^{1/q}$. It turns out that this is the appropriate quantity in our context, as it is closely related to diffusion and we require this quantity to be small in order to certify existence of a small cut. We leave

the proof of Theorem 4.10 to Section 4.5.1. Below we provide a physical interpretation of Algorithm 1 as diffusing excess mass iteratively from one node to another. In this context, the ℓ_1 norm used in Theorem 4.10 measures the sum of excess mass from every node in the graph. Recall that the excess mass on node i is defined to be the net mass at i minus its sink capacity. If the net mass is less than the sink capacity, then excess is 0. In our diffusion, the goal is to remove excess mass from the source nodes, and thus we naturally expect that the total amount of excess mass converge to 0 iteratively throughout the diffusion process. This is precisely what Theorem 4.10 guarantees (and it provides rate of convergence). In other words, Theorem 4.10 can be interpreted as lower bounding the number of required diffusion iterations such that at least $(1-\epsilon)$ fraction of the total amount of source mass has been absorbed by the sinks in the graph. The next section explains this interpretation in detail.

4.3.1 Interpreting Algorithm 1 as an Iterative Diffusion Process

Recall that

$$F(z) = \frac{1}{q}\|BD^{-1/q}z\|_q^q + z^T D^{-1/q}(d-\Delta).$$

We can expand the terms in $F(z)$ and write it equivalently as

$$F(z) = \frac{1}{q}\sum_{(i,j)\in E}\left|\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}}\right|^q + \sum_{i\in V} z_i\left(d_i^{1/p} - \frac{\Delta_i}{d_i^{1/q}}\right),$$

from which it is straightforward to see that the i^{th} coordinate gradient of $F(z)$ is

$$\nabla_i F(z) = d_i^{1/p} - \frac{\Delta_i}{d_i^{1/q}} + \sum_{j\sim i} d_i^{-1/q}\left|\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}}\right|^{q-1}\operatorname{sign}\left(\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}}\right). \tag{4.11}$$

Hence

$$d_i^{1/q}\nabla_i F(z) = d_i - \Delta_i + \sum_{j\sim i}\left|\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}}\right|^{q-1}\operatorname{sign}\left(\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}}\right).$$

One may easily verify that the flow solution f^* of Problem (4.9) and the dual solution x^* of Problem (4.8) are related by

$$|f_e^*| = |x_i^* - x_j^*|^{q-1}\operatorname{sign}(x_i^* - x_j^*).$$

We may extend this primal-dual relationship to any given $z \geq 0$, and define the flow of mass from node i to node j as

$$\tilde{f}(i,j) = \left| \frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}} \right|^{q-1} \operatorname{sign}\left(\frac{z_i}{d_i^{1/q}} - \frac{z_j}{d_j^{1/q}} \right). \tag{4.12}$$

Therefore, every $z \geq 0$ corresponds to a diffusion where the source mass is routed according to the flow variables $\tilde{f}$ defined in the above. Note that when $\tilde{f}(i,j)$ is defined based on the optimal dual solution z^*, we have $\tilde{f}(i,j) = f^*(i,j)$ where f^* is the optimal flow that minimizes the primal p-norm objective. For any other z, we still have a diffusion based on $\tilde{f}$, but the excess on some nodes may be strictly positive, or the flow $\tilde{f}$ incur a higher cost than f^*.

For any given $z \geq 0$, the quantity $d_i^{1/q} \nabla_i F(z)$ measures the difference between i's sink capacity d_i and i's net mass $\Delta_i - \sum_{j \sim i} \tilde{f}(i,j)$. If $d_i^{1/q} \nabla_i F(z) < 0$, then there is excess mass on node i, otherwise, the net mass on i is not larger than its sink capacity. Therefore, for given any $z \geq 0$, the quantity

$$\|D^{1/q}[-\nabla_i F(z)]_+\|_1 = \sum_{i \in V} \max\{0, -d_i^{1/q} \nabla_i F(z)\} = \sum_{i \in V} \operatorname{excess}(i) \tag{4.13}$$

measures the total amount of excess mass in the graph, if we route the source mass according to the flow variables $\tilde{f}$ defined in Equation (4.12). In our p-norm flow diffusion, we require that the excess mass on each node is 0. This intuitively explains why in Theorem 4.10 we measure progress in terms of how small $\|D^{1/q}[-\nabla_i F(z^k)]_+\|_1$ can be after k iterations.

Using the above interpretation, in Algorithm 1, if we define a diffusion based on z^k, then the active set A^k corresponds precisely to the set of nodes having strictly positive excess mass. A coordinate-wise update on $i \in A^k$ corresponds to pushing the excess mass from node i to its neighbors. As we will show in Proposition 4.16, the coordinate-wise gradient value remains non-positive after the update. This means that when a node i has some positive excess mass, the amount of mass that Algorithm 1 moves from i to its neighbors is never larger than the amount of excess. Consequently, if a node is saturated, it remains saturated throughout the execution of Algorithm 1. Starting from the set of seed nodes, Algorithm 1 literately expands the set of saturated nodes, and this set eventually converges to what it should be at optimality. Therefore, the entire execution of Algorithm 1 induces an iteratively expanding diffusion process, where the total amount of excess mass in the graph decreases as more and more nodes receive mass from their neighbors and become saturated. This intuitively explains why Algorithm 1 is inherently a local algorithm.

4.3.2 The Special Case $p = q = 2$

Even though our diffusion problem is more interesting when $p > 2$ or equivalently when $q < 2$, we discuss briefly the special case when $p = q = 2$. In this case, the coordinate-wise gradient of the objective function in Problem (4.8) has a slightly better smoothness property and hence permits using a larger step-size in the coordinate-wise updates. Since we will revisit this special case later in Chapter 6 and Chapter 7 for applications in statistical settings, we provide a variant of Algorithm 1 in Algorithm 2 for solving Problem (4.8) when $p = q = 2$. Apart from fixing $p = q = 2$, one can easily verify that the only difference between Algorithm 2 and Algorithm 1 is that the step-size used Algorithm 2 is two times the step-size used in Algorithm 1. As discussed in the previous section, there is an intuitive interpretation of the coordinate descent algorithm as an iterative diffusion process. We provide a concrete example in Algorithm 3, where we rewrite the algorithmic steps in Algorithm 2 using their combinatorial interpretation as diffusing mass in the graph.

Algorithm 2 Variant of Algorithm 1 for $p = q = 2$

Initialize: $z^0 = 0$
For $k = 0, 1, 2, \ldots, K$ **do**
  Set $A^k = \{i \in V \mid \nabla_i F(z^k) < 0\}$.
  Set $y^k = z^k$.
  For each $i \in A^k$ **do**
    Update $y^k = y^k + |\nabla_i F(y^k)| \cdot e_i$.
  Set $z^{k+1} = y^k$.
Return $D^{-1/2} z^K$.

Let z^* denote the optimal solution of Problem (4.8) and let $\bar{S} = \text{supp}(z^*)$. Define

$$d_{\max}(\bar{S}) = \max_{i \in \bar{S}} \deg(i), \quad \Phi^*(\bar{S}) = \min_{S' \subseteq \bar{S}} \Phi(S').$$

Theorem 4.11 gives the required number of iterations for Algorithm 2 and equivalently Algorithm 3 such that at least $(1 - \epsilon)$ fraction of the total amount of source mass has been absorbed by the sinks in the graph. We leave the details of proof to Section 4.5.3.

Theorem 4.11. *Let $\epsilon > 0$. Let K be such that*

$$K \geq \frac{16 d_{\max}(\bar{S})}{\Phi^*(\bar{S})^2} \log\left(\frac{2\|\Delta\|_1}{\epsilon}\right).$$

Then the $(K+1)^{th}$ iterate z^{K+1} generated by Algorithm 2 satisfies

$$\|D^{-1/2}[-\nabla F(z^{K+1})]\|_1 \leq \epsilon \|\Delta\|_1.$$

Algorithm 3 Combinatorial interpretation of Algorithm 2

Input: graph $G = (V, E)$, source Δ.

1. Initially, $z(i) = 0$, $\text{mass}(i) = \Delta(i)$ and $\text{excess}(i) = \max\{0, \text{mass}(i) - \deg(i)\}$, $\forall i \in V$.
2. For $k = 1, 2, \ldots, K$ do
   (a) Define $A^k = \{i \in V \mid \text{excess}(i) > 0\}$.
   (b) For each $i \in A^k$, apply `push`(i).
3. Return x where $x(i) = \dfrac{z(i)}{\sqrt{\deg(i)}}$ for $i \in V$.

`push`(i):

Make the following updates:

1. $z(i) = z(i) + \dfrac{\text{excess}(i)}{\sqrt{\deg(i)}}$.
2. $\text{mass}(i) = \deg(i)$.
3. For each node $j \sim i$:

$$\text{mass}(j) = \text{mass}(j) + \frac{\text{excess}(i)}{\deg(i)},$$
$$\text{excess}(j) = \max\{0, \text{mass}(j) - \deg(j)\}.$$

For Algorithm 3, this means that, after $K + 1$ iterations,

$$\sum_{i \in V} \text{excess}(i) \leq \epsilon \sum_{i \in V} \Delta(i).$$

4.4 Empirical Results

We implemented Algorithm 1 (for general $p > 2$ and $1 < q < 2$) and Algorithm 3 (for $p = q = 2$) in Julia.[1] For Algorithm 1, our implementation employs additional line-search for each coordinate update, so that the coordinate-wise gradient value equals 0 after the update. That is, we use line-search to find step-size $\alpha_{k,i}$ such that

$$\nabla_i F(z^k - \alpha_{k,i} \nabla_i F(z^k) e_i) = 0.$$

This leads to exact coordinate minimization steps which can speed up convergence in practice. Computing the required step-size $\alpha_{k,i}$ through binary line-search is possible because

[1] Our code is available at http://github.com/s-h-yang/pNormFlowDiffusion.

the coordinate gradients are monotone, see Fact 4.12. From a combinatorial perspective, exact coordinate minimization on the i^{th} coordinate corresponds to removing all excess mass from node i by diffusing it to the neighbors of i.

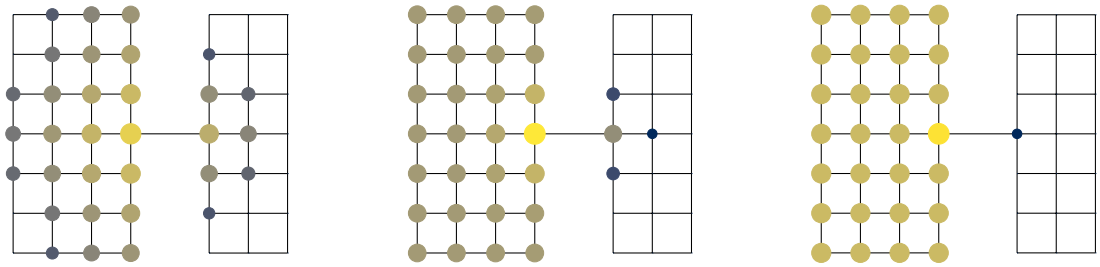
We start with a toy example in Section 4.4.1 to demonstrate the distinct behaviors of the optimal dual variables x^* under different p-norms. In Section 4.4.2, we show local clustering performance on synthetic graphs under different p-norms and we compare it with existing state-of-the-art local diffusion methods. In Section 4.4.3, we apply p-norm diffusion for the local community detection problem in real-world networks and show that a small $p = 4$ already offers a significant improvement in terms of accuracy.

4.4.1 Toy Example: Diffusion on a Dumbbell

The best way to visualize p-norm flow diffusion for various p values is to start the diffusion process on the same graph and the same set of seed nodes, with equal amount of source mass. To this end, we run p-norm flow diffusion for $p \in \{2, 4, 8\}$ on a synthetic tiny "dumbbell" graph obtained by removing edges from a 7×7 grid graph. We pick a single seed node s which locates on the left side of the "bridge", and assign source mass $\Delta(s) = 121$ to that node. For each p, we plot optimal dual variables x^* in Figure 4.1, where we use color intensities and circle sizes to indicate the relative magnitude of dual values, i.e., brighter colors and larger circles represent higher dual values for a fixed p, and no circle means the corresponding node has a zero (or nearly zero) dual value.

Recall that a dual variable is nonzero only if the corresponding node is saturated, i.e., the mass it holds equals its degree). Observe that when $p = 2$ the diffusion leaks a lot of mass to the other side of dumbbell, whereas for $p = 4$ and $p = 8$ the diffusion saturates the entire left-hand side part of the dumbbell, without leaking much mass to the right. The reason that this happens is because $p = 4$ and $p = 8$ put significantly larger penalty on the flow that passes through the "bridge", making it difficult to send mass over to the other side.

Note that, if we were to perform the sweep cut procedure described in Section 4.2.1, then diffusion with $p = 2$ would fail to recover the "correct" cluster, while diffusion with either $p = 4$ or $p = 8$ would return the entire left-hand side of dumbbell, which is the best possible result in terms of low conductance. Although this example is overly simplistic, it does demonstrate that p-norm flow diffusion with higher p values are more sensitive to bottlenecks when routing mass around the seed node(s), and, on the other hand, it is possible to overcome such bottleneck even when p is just slightly larger than 2, e.g. $p = 4$



(a) $p = 2$ (b) $p = 4$ (c) $p = 8$

Figure 4.1: p-norm flow diffusion on a dumbbell: color intensities and circle sizes are chosen to reflect relative magnitude of the optimal dual variables x^*. The seed node is the boundary node on the left side of the "bridge" (it has the brightest yellow color). Observe that when $p = 2$, the dual values spread towards all directions, while on the contrary, larger p forces dual values to concentrate on one side of the dumbbell.

and $p = 8$. Indeed, our subsequent experiments show that there is already a significant improvement in the context of local graph clustering, when raising $p = 2$ to $p = 4$.

4.4.2 Experiments on Synthetic Graphs

The LFR model [73] is a widely used generative model for creating synthetic benchmark instances for evaluating community detection algorithms. It is essentially a stochastic block model with the additional property that nodes' degrees follow the power law distribution, and there is a parameter μ controlling what fraction of a node's neighbours is outside the node's block. We use the LFR model to generate synthetic graphs with the following parameter setting: number of nodes is 1000, average degree is 10, maximum degree is 50, minimum community size is 20, maximum community size is 100, power law exponent for degree distribution is 2, power law exponent for community size distribution is 1. We consider different μ values between 0.1 and 0.4 with step 0.02. This range means that the synthetic graphs contain reasonable noisy (but not completely noise) clusters.

We compare the performance of p-norm flow diffusion with the ℓ_1-regularized PageRank [50] and a nonlinear diffusion method introduced in [63]. Given a starting node s, teleportation probability α, and tolerance parameter ρ, the ℓ_1-regularized PageRank is an optimization problem whose optimal solution is closely related to the Approximate Personalized PageRank (APPR) vector [8]. This ℓ_1-regularized optimization formulation

allows us to apply coordinate method and obtain an APPR vector in linear and strongly local running time. The nonlinear diffusion method from [63] iteratively applies a pointwise nonlinear transformation to node values after each matrix-vector product between the Laplacian matrix and the incumbent node vector. Since it is demonstrated in [63] that a nonlinear transformation defined by the power function $u \mapsto u^{0.5}$ has the best overall performance when compared to other nonlinear functions like tanh or the heat kernel, in our experiments we simply use the nonlinear diffusion defined by this power function. We choose $p \in \{2, 4, 8\}$ for p-norm diffusion and demonstrate the advantage of our method even when p is a small constant.

Our goal here is to compare the behaviour of different algorithms under a fixed setting, and not to fine tune any particular method. Therefore, besides an additional experiment on the synthetic graphs to examine the effect of having a larger overlap between the set of seed nodes and the target cluster, we always start diffusion from a **single seed node**, and set $\|\Delta\|_1 = t \cdot \mathrm{vol}(C)$ for some constant factor t and some target cluster C (recall from Assumption 4.3 this is without loss of generality). In particular, for our experiments with the synthetic graphs we set $t = 5$. We made sure the choice of t is such that $\|\Delta\|_1$ is less than the volume of the graph. For the nonlinear diffusion method, we use the same parameter setting as what the authors suggested in [63]. The ℓ_1-regularized PageRank is the only linear diffusion method among all methods that we compare, so we allow it to use ground truth information to choose the teleportation parameter α giving the best conductance result. We tune α by picking $\alpha \in \{\lambda/8, \lambda/4, \lambda/2, \lambda, 2\lambda\}$, where λ is the smallest nonzero eigenvalue of the normalized Laplacian for the subgraph that corresponds to the target cluster. Since the support size of APPR vector is bounded by $1/\rho$ [50], and the support size of dual variables for p-norm diffusion is bounded by $\|\Delta\|_1$, for comparison purposes, we set the tolerance parameter ρ for ℓ_1-regularized PageRank so that $\rho = 1/\|\Delta\|_1$.

Our theory indicates (not surprisingly) that better overlap of the input seed set and a target cluster will result in output cluster having better conductance and consequently better F1 if the target cluster has low conductance. We demonstrate this empirically in our first experiment on a synthetic graph generated by the LFR model with $\mu = 0.3$. Note that the parameter $\mu = 0.3$ means that 30% of all edges of a node from the ground truth cluster links to the outside of that cluster. Because of this noise level, the goal is not to recover the ground truth exactly, but to obtain a cluster that overlaps well with the target (e.g., has a good F1 measure). We have chosen $\mu = 0.3$ because it represents reasonably noisy clusters that are not completely noise. For this experiment, we randomly pick a set of seed nodes S from some target cluster C in the graph and vary the percentage overlap of S in C, i.e., $|S|/|C|$. Then for each $p \in \{2, 4, 8\}$, we run p-norm flow diffusion and use the sweep cut procedure to find the smallest conductance cluster among the level cuts.

Figure 4.2 shows the mean and the variance for the conductance and the F1 measure while varying the ratio $|S|/|C|$. As expected, as the percentage overlap $|S|/|C|$ increases, which also means that the tightest α in Theorem 4.4 decreases, we recover clusters with lower conductance and higher F1 score. Observe that while there is a large gap in both the conductance and the F1 between the results for $p = 2$ and $p = 4$, the gain in raising $p = 4$ to $p = 8$ is comparatively marginal.

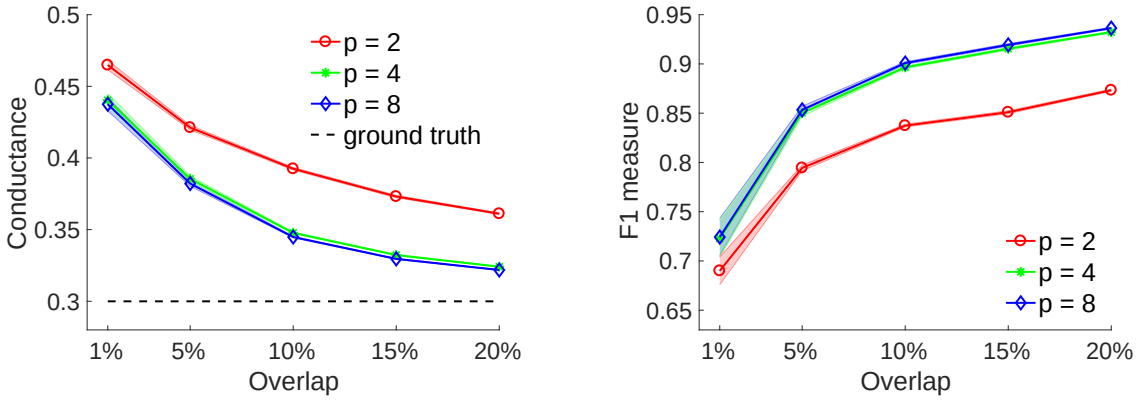


Figure 4.2: Conductance and F1 of the output cluster for the local clustering task when the seed set has different levels of overlap with the target cluster. The underlying graph is generated by the LFR model with $\mu = 0.3$. The black dashed line show ground truth conductance. The bands show the variation over 100 trials.

In the subsequent experiment we start the diffusion process from **single seed node**, as this is the most common practice for semi-supervised local clustering tasks. Our second experiment considers a set of synthetic graphs generated by the LFR model with varying μ. For each graph, we start from a random seed node and we repeat the experiment 100 times (the seed node may be different from one trial to another). Figure 4.3 shows the mean and the variance for the conductance and the F1 measure while varying μ. Notice that, p-norm flow diffusion behaves similarly to the fine tuned ℓ_1-regularized PageRank in both the conductance and the F1 measure when $p = 2$, whereas it significantly outperforms other methods when $p = 4$ and $p = 8$. Observe that there is a slight gain in terms of conductance by raising $p = 4$ to $p = 8$, but such improvement is marginal. This is not really surprising, since qualitatively the 4-norm unit ball is already very close to the ∞-norm unit ball (i.e. the box).

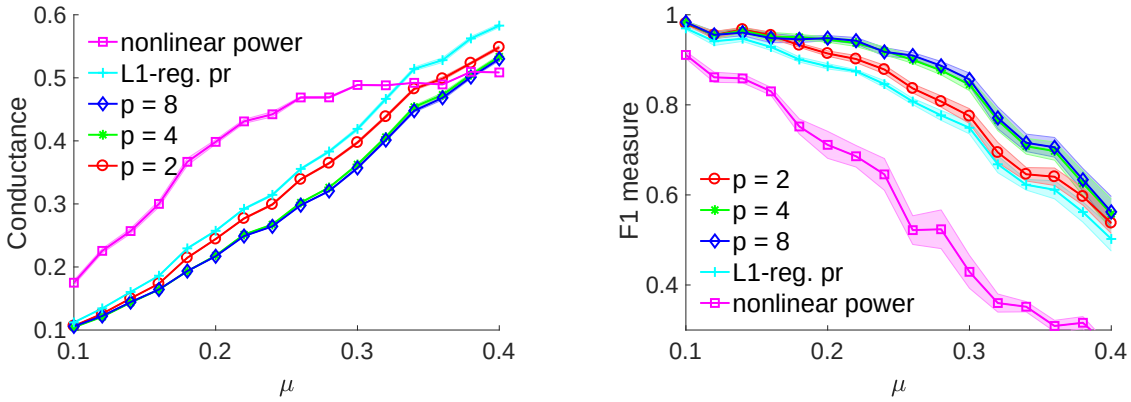


Figure 4.3: Conductance and F1 of the output cluster for the local clustering task in synthetic graphs generated by the LFR model. We start diffusion from a randomly selected single seed node. The bands show the variation over 100 trials. The ℓ_1-regularized PageRank is labelled as `L1-reg. pr`. The nonlinear diffusion method from [63] is labelled as `nonlinear power`.

4.4.3 Experiments on Real-world Graphs

We consider four real-world graphs which include both social and biological networks. The graphs are all unweighted and undirected. Table 4.1 shows some basic characteristics of these graphs.

Table 4.1: Summary of real-world graphs

dataset	number of nodes	number of edges	description
FB-Johns55	$5,157$	$186,572$	Facebook social network for Johns Hopkins University
Colgate88	$3,482$	$155,043$	Facebook social network for Colgate University
Sfld	232	$15,570$	Pairwise similarities of blasted sequences of proteins
Orkut	$3,072,441$	$117,185,083$	Large-scale on-line social network

The two Facebook graphs are chosen from the Facebook100 dataset based on their assortativity values in the first column of Table A.2 in [117], where the data were first introduced and analyzed. Each of these comes with some features, e.g., gender, dorm,

Table 4.2: Filtered "ground truth" clusters for real-world graphs

dataset	feature	volume	nodes	conductance
FB-Johns55	year 2006	81893	845	0.54
	year 2007	89021	842	0.49
	year 2008	82934	926	0.39
	year 2009	33059	910	0.21
	major index 217	10697	201	0.26
Colgate88	year 2004	14888	230	0.54
	year 2005	50643	501	0.50
	year 2006	62065	557	0.48
	year 2007	68382	589	0.41
	year 2008	62430	641	0.29
	year 2009	35379	641	0.11
Sfld	urease	31646	100	0.42
	AMP	3186	28	0.53
	phosphotriesterase	381	7	0.78
	adenosine	1062	10	0.83
	dihydroorotase3	494	7	0.83
	dihydroorotase2	3119	13	0.90
Orkut	A	49767	383	0.42
	B	31912	202	0.45
	C	16022	141	0.45
	D	11698	113	0.46
	E	26248	194	0.47
	F	4617	64	0.47
	G	13786	128	0.47
	H	14109	107	0.48
	I	18652	195	0.49
	J	41612	318	0.50
	K	20204	223	0.50

major index and year. We consider a set of nodes with the same feature as a ground truth cluster, and filter the "ground truth" clusters by setting a 0.6 threshold on maximum conductance. We also omit clusters whose volume is larger than one third of the volume of the entire graph, since discovering clusters whose sizes are close to the entire graph is not the purpose of local clustering.

The biology dataset Sfld contains pairwise similarities of blasted sequences of 232 proteins belonging to the amidohydrolase superfamily [25]. A gold standard is provided describing families within the given superfamily. According to the gold standard the amidrohydrolase superfamily contains 29 families. We consider each family as a ground truth cluster, and filter them by setting a 0.9 threshold on the maximum conductance.

The last dataset Orkut is a free on-line social network where users form friendship each other. It can be downloaded from [74]. This network comes with 5000 ground truth communities, which we filter by setting minimum community size 50, maximum cut conductance 0.5, and minimum ratio between internal connectivity (i.e., the smallest nonzero eigenvalue of the normalized Laplacian of the subgraph defined by the cluster) and cut conductance to 0.85. This resulted in 11 reasonably noisy clusters, having conductance between 0.4 and 0.5. We include the statistics of all ground truth clusters that we used in the experiments in Table 4.2.

We examine the performance of local graph diffusion methods for detecting the ground-truth clusters given a single seed node from that cluster. As in the precious section, we compare p-norm flow diffusion with the ℓ_1-regularized PageRank [8] and the nonlinear diffusion method from [63]. We always start diffusion from single seed node s within a target cluster, and set we set the source mass $\Delta(s) = t \cdot \mathrm{vol}(C)$ for some constant factor t and target cluster C. For the Facebook graphs we set $t = 3$ because the target clusters already have a large volume, and for both the Orkut network we set $t = 5$. On the other hand, because the clusters in Sfld are very noisy, we vary $t \in \{1, 2, \ldots, 10\}$ and pick the cluster with the lowest conductance. We use the same parameters settings for the ℓ_1-regularized PageRank and the nonlinear diffusion method as before. That is, we use the same parameters for the nonlinear diffusion method as suggested by the authors [63]. For the ℓ_1-regularized PageRank we use ground truth information to choose the teleportation parameter α giving the best conductance result. We tune α by picking $\alpha \in \{\lambda/8, \lambda/4, \lambda/2, \lambda, 2\lambda\}$, where λ is the smallest nonzero eigenvalue of the normalized Laplacian for the subgraph that corresponds to the target cluster. We pick the tolerance parameter ρ for ℓ_1-regularized PageRank so that $\rho = 1/\|\Delta\|_1$.

Table 4.3: Local clustering results for 3 small to medium size real-world networks

dataset	cluster	measure	$p = 2$	$p = 4$	ℓ_1-reg. pr [50]	nonlinear [63]
FB-Johns55	year 2006	f1	**0.36**	0.35	0.31	0.31
		cond	**0.34**	**0.34**	0.50	0.40
	year 2007	f1	**0.39**	**0.39**	0.38	0.36
		cond	0.31	**0.30**	0.45	0.41
	year 2008	f1	**0.51**	**0.51**	**0.51**	0.37
		cond	**0.34**	**0.34**	0.44	0.41
	year 2009	f1	0.84	**0.85**	0.83	0.49
		cond	0.23	**0.22**	0.24	0.40
	major index 217	f1	0.85	**0.87**	0.83	0.75
		cond	0.23	**0.22**	0.25	0.29
Colgate88	year 2004	f1	0.50	**0.51**	0.43	0.25
		cond	0.66	0.66	0.71	**0.36**
	year 2005	f1	**0.45**	**0.45**	0.41	0.37
		cond	0.51	0.51	0.53	**0.37**
	year 2006	f1	**0.45**	**0.45**	0.43	0.39
		cond	**0.37**	**0.36**	0.50	0.38
	year 2007	f1	0.49	0.49	**0.51**	0.45
		cond	**0.34**	**0.34**	0.45	0.39
	year 2008	f1	0.76	**0.80**	0.74	0.55
		cond	0.31	**0.30**	0.35	0.40
	year 2009	f1	0.96	**0.97**	0.96	0.82
		cond	0.13	**0.12**	0.13	0.24
Sfld	urease	f1	0.74	**0.76**	0.72	0.63
		cond	**0.44**	0.45	**0.44**	0.48
	AMP	f1	**0.83**	**0.83**	**0.83**	**0.83**
		cond	**0.41**	**0.41**	0.42	0.43
	phosphotriesterase	f1	0.93	0.93	**1.00**	0.13
		cond	0.81	0.81	0.81	**0.49**
	adenosine	f1	**0.44**	**0.44**	**0.44**	0.34
		cond	0.46	0.46	0.46	**0.44**
	dihydroorotase3	f1	**0.96**	**0.96**	**0.96**	0.07
		cond	0.84	0.84	0.84	**0.44**
	dihydroorotase2	f1	**0.39**	**0.39**	0.20	0.20
		cond	0.77	0.78	0.85	**0.48**

Table 4.4: Local clustering results for the large Orkut social network

cluster	measure	$p = 2$	$p = 4$	ℓ_1-reg. pr [50]
A	f1	0.56	**0.58**	0.49
	cond	0.48	**0.47**	0.51
B	f1	0.71	**0.73**	0.66
	cond	0.35	**0.33**	0.37
C	f1	0.63	**0.64**	0.57
	cond	0.33	**0.32**	0.35
D	f1	0.73	**0.76**	0.72
	cond	0.49	**0.48**	0.51
E	f1	0.61	**0.62**	0.56
	cond	0.52	**0.51**	0.54
F	f1	0.79	**0.81**	0.76
	cond	0.52	**0.51**	0.54
G	f1	0.72	**0.73**	0.68
	cond	0.52	**0.50**	0.53
H	f1	0.68	**0.70**	0.67
	cond	0.52	**0.51**	0.54
I	f1	0.60	**0.62**	0.56
	cond	0.49	**0.48**	0.52
J	f1	0.52	**0.54**	0.47
	cond	0.53	**0.52**	0.56
K	f1	0.54	**0.56**	0.51
	cond	0.57	**0.56**	0.60

The local clustering results are shown in Table 4.3 and Table 4.4. where we run the diffusion algorithms starting from each node in each cluster and report the average conductance and F1 measure. We omit the nonlinear diffusion method [63] for the Orkut network, as the method does not scale well to large graphs. The Facebook graphs contain ground-truth clusters ranging from low to medium conductance. For almost all clusters, p-norm flow diffusion has the best F1 measure and and the best conductance result. The six clusters in the biological dataset Sfld are very noisy, having median conductance around 0.8. Hence it is highly likely that a ground truth cluster is contained in some larger clusters (but not one of the 29 true families) that have much lower conductance. This partially explains why the nonlinear diffusion method returns low conductance clusters but has poor recovery accuracy in terms of the F1 measure. Note that the nonlinear diffusion method is the only global method that we compare with. Both p-norm flow diffusion and ℓ_1-regularized PageRank are local methods, and they take advantage of locality to find local clusters

that align well with the ground-truth cluster. Therefore, the results for the Sfld dataset demonstrate the advantage of local methods at recovering relatively small ground-truth clusters. On the Orkut network, p-norm flow diffusion with $p = 4$ gives best result on all clusters.

4.5 Proofs of Results

4.5.1 Proof of Theorem 4.10

From Equation (4.11) it is straightforward to verify a useful monotonicity property of $\nabla_i F$. We state this monotonicity property in Fact 4.12.

Fact 4.12 (Gradient monotonicity)**.** Fix $z \in \mathbb{R}$. We have that

- For all $i \in [n]$, $\nabla_i F(z + te_i)$ is monotonically increasing in t, i.e., $\nabla_i F(z + t_1 e_i) \geq \nabla_i F(z + t_2 e_i)$ if and only if $t_1 \geq t_2$;
- For all $i, j \in [n]$ where $i \neq j$, $\nabla_i F(z + te_j)$ is monotonically decreasing in t, i.e., $\nabla_i F(z + t_1 e_j) \geq \nabla_i F(z + t_2 e_j)$ if and only if $t_1 \leq t_2$.

Lemma 4.13 (Hölder smoothness)**.** *The function F in Equation (4.10) is coordinate-wise Hölder smooth with constant 2 and exponent $q - 1$:*

$$|\nabla_i F(z + te_i) - \nabla_i F(z)| \leq 2|t|^{q-1}, \text{ for all } i \in [n].$$

Proof. Note that if we have that

$$\left||s|^{q-1}\text{sign}(s) - |t|^{q-1}\text{sign}(t)\right| \leq 2|s - t|^{q-1}, \ \forall s, t \in \mathbb{R}, \tag{4.14}$$

then it follows immediately from the expression of $\nabla_i F$ in Equation (4.11) that

$$|\nabla_i F(z + te_i) - \nabla_i F(z)| \leq \sum_{j \sim i} 2d_i^{-1/q} \left| \frac{z_i + t}{d_i^{1/q}} - \frac{z_i}{d_i^{1/q}} \right|^{q-1} = \sum_{j \sim i} 2d_i^{-1}|t|^{q-1} = 2|t|^{q-1}.$$

So let us show that (4.14) holds. Let $s, t \in \mathbb{R}$. If both s and t are non-negative, or if both are non-positive, then (4.14) reduces to

$$\left||s|^{q-1} - |t|^{q-1}\right| \leq 2\left||s| - |t|\right|^{q-1}, \ \forall s, t \in \mathbb{R}.$$

Since $0 < q - 1 \leq 1$ we have that for $a, b \geq 0$,

$$1 \leq \left(\frac{a}{a+b}\right)^{q-1} + \left(\frac{b}{a+b}\right)^{q-1},$$

which implies $(a+b)^{q-1} - a^{q-1} \leq b^{q-1}$. Taking $a = \min(|s|, |t|)$ and $b = \max(|s|, |t|) - a$ yields $\big||s|^{q-1} - |t|^{q-1}\big| \leq \big||s| - |t|\big|^{q-1} \leq 2\big||s| - |t|\big|^{q-1}$ as required. On the other hand, if s and t have different signs, then (4.14) reduces to

$$|s|^{q-1} + |t|^{q-1} \leq 2\big(|s| + |t|\big)^{q-1}, \ \forall s, t \in \mathbb{R}.$$

This is always true since $a^{q-1} + b^{q-1} \leq 2\max(a, b)^{q-1} \leq 2(a+b)^{q-1}$ for $a, b \geq 0$. □

Lemma 4.14 (Progress per coordinate update)**.** *For all $z \in \mathbb{R}^n$,*

$$F\left(z - \left(\frac{|\nabla_i F(z)|}{2}\right)^{p-1} \operatorname{sign}(\nabla_i F(z)) e_i\right) \leq F(z) - \frac{|\nabla_i F(z)|^p}{p 2^{p-1}}.$$

Proof. Using Lemma 4.13 and adopting the usual argument for proving the descent lemma for Lipschitz smooth functions, i.e. invoking the Fundamental Theorem of Calculus, for example, see Lemma 1 in [139], we get

$$F(z + te_i) \leq F(z) + \nabla_i F(z) t + \frac{2}{q}|t|^q, \ \forall z \in \mathbb{R}^n, \ \forall t \in \mathbb{R}.$$

Setting

$$t = -\left(\frac{|\nabla_i F(z)|}{2}\right)^{p-1} \operatorname{sign}(\nabla_i F(z))$$

gives the required result. □

Lemma 4.15 (Progress per iteration)**.** *Let z^k denote the k^{th} iterate generated by Algorithm 1, we have that*

$$F(z^{k+1}) \leq F(z^k) - \frac{1}{p 2^{p-1}} \big\|[-\nabla F(z^k)]_+\big\|_p^p,$$

where $[a]_+ = \max\{a, 0\}$ denotes entry-wise maximum between a and 0.

Proof. Fix $k \geq 0$ and consider an arbitrary ordering $i_1, i_2, \ldots, i_{|A^k|}$ of the indices in A^k. Note that the inner loop of Algorithm 1 performs coordinate-wise update, the vector y^k is updated at every iteration of the inner loop, and correspondingly the gradient $\nabla F(y^k)$ changes at every iteration of the inner loop. In other words, for $j \in \{1, 2, \ldots, |A^k|\}$ let $y^{k,j}$ denote the vector obtained from sequentially updating coordinates $i_1, i_2, \ldots, i_j$ according to the inner loop of Algorithm 1, then $\nabla_i F(y^{k,a})$ may or may not equal to $\nabla_i F(y^{k,b})$ if $a \neq b$. In order to obtain the required result we will show that

$$|\nabla_{i_j} F(y^{k,j-1})| \geq |\nabla_{i_j} F(z^k)|, \text{ for } j \in \{2, 3 \ldots, |A^k|\}. \tag{4.15}$$

Assume that this is true, then we will get

$$\begin{aligned}
F(z^{k+1}) &= F(y^{k,|A^k|}) && \text{(by definition } z^{k+1} = y^{k,|A^k|}\text{)} \\
&\leq F(y^{k,|A^k|-1}) - \frac{\left|\nabla_{i_{|A^k|-1}} F(y^{k,|A^k|-1})\right|^p}{p2^{p-1}} && \text{(by Lemma 4.14 and definition of } A^k\text{)} \\
&\leq F(y^{k,|A^k|-1}) - \frac{\left|\nabla_{i_{|A^k|-1}} F(z^k)\right|^p}{p2^{p-1}} && \text{(by (4.15))} \\
&\leq F(z^k) - \sum_{i \in A_k} \frac{\left|\nabla_i F(z^k)\right|^p}{p2^{p-1}} && \text{(by induction)} \\
&= F(z^k) - \frac{1}{p2^{p-1}} \left\| [-\nabla F(z^k)]_+ \right\|_p^p && \text{(by definition of } A^k\text{)}
\end{aligned}$$

as required. It left to verify that (4.15) holds. To see this, note that Algorithm 1 ensures that $y^k \geq z^k$ throughout the inner loop, in other words, $y^{k,j} \geq z^k$ for all j. By the monotonicity property of $\nabla_i F(z + te_j)$ in Fact 4.12, this means that $\nabla_{i_j} F(y^{k,j-1}) \leq \nabla_{i_j} F(z^k)$, as updates on other coordinates can only decrease the value of the gradient at coordinate i_j. By the definition of A^k, we have $\nabla_{i_j} F(z^k) < 0$, and thus $|\nabla_{i_j} F(y^{k,j-1})| \geq |\nabla_{i_j} F(z^k)|$, which is exactly (4.15). □

Proposition 4.16 (Convergence)**.** *Let $-\infty < z^* := \min_{z \geq 0} F(z)$ where F is given in Equation (4.10). Then the iterates $z_1, z_2, \ldots$ generated by Algorithm 1 converges to z^* as $K \to \infty$.*

Proof. First of all, the sequence $\{z_k\}_{k \geq 0}$ must converge to some limit point $\bar{z}$, as otherwise the vector $[-\nabla_i F(z^k)]_+$ must be bounded away from 0 as k tends to infinity, which by Lemma 4.15 means that the sequence of function values $\{F(z^k)\}_{k \geq 0}$ is unbounded from below, and this contradicts our assumption that $z^* > -\infty$. We will show that $\bar{z} = z^*$.

To do this, it suffices to verify that $\bar{z}$ satisfies the optimality condition: (i) $\bar{z} \geq 0$, (ii) $\nabla F(\bar{z}) \geq 0$, (iii) $\nabla_i F(\bar{z}) \leq 0$ for all $i \in \text{supp}(\bar{z})$.

Since Algorithm 1 ensures that $z^k \geq 0$ for all k, this implies that $\bar{z} \geq 0$, so (i) is satisfied. To see that $\nabla F(\bar{z}) \geq 0$, let $\bar{A} = \{i \in [n] | \nabla_i F(\bar{z}) < 0\}$. Since the sequence $\{z_k\}_{k \geq 0}$ generated by Algorithm 1 converges to $\bar{z}$, we must have $\bar{A} = \emptyset$, as otherwise the sequence $\{z_k\}_{k \geq 0}$ would not have converged to $\bar{z}$. So (ii) is satisfied. Finally, note that if $\nabla_i F(z^k) \leq 0$ for all $i \in \text{supp}(z^k)$ for all $k \geq 0$, then by the continuity of $\nabla_i F$, and the fact that there must be a large enough $N \in \mathbb{N}$ such that $\text{supp}(z^k) = \text{supp}(\bar{z})$ for all $k \geq N$, we get that (iii) is satisfied, too.

Therefore, it remains to show that $\nabla_i F(z^k) \leq 0$ for all $i \in \text{supp}(z^k)$ for all $k \geq 0$. Suppose at some iteration $k \geq 0$ we have that $\nabla_i F(z^k) \leq 0$ for every $i \in \text{supp}(z^k)$, we will show that $\nabla_i F(z^{k+1}) \leq 0$ for every $i \in \text{supp}(z^{k+1})$. If $A^k = \emptyset$ then this trivially holds. So suppose that $A^k = \{i_1, i_2, \ldots, i_{|A^k|}\} \neq \emptyset$. Let $y^{k,0} = z^k$ and $y^{k,j}$ denote the vector obtained from sequentially updating the first j coordinates from A^k according to the inner loop of Algorithm 1. It follows from Lemma 4.13 that

$$\begin{aligned} |\nabla_{i_1} F(y^{k,1}) - \nabla_{i_1} F(z^k)| &= \left| \nabla_{i_1} F\left(z^k + \left(\frac{|\nabla_{i_1} F(z^k)|}{2} \right)^{p-1} e_{i_1} \right) - \nabla_{i_1} F(z^k) \right| \\ &\leq 2 \left| \frac{\nabla_{i_1} F(z^k)}{2} \right|^{(p-1)(q-1)} = |\nabla_{i_1} F(z^k)|. \end{aligned}$$

By the definition of A^k we know that $\nabla_{i_1} F(z^k) < 0$, and hence the above implies that $\nabla_{i_1} F(y^{k,1}) \leq 0$. In addition, by Fact 4.12, we know that updating a coordinate $i_j \in A^k$ according to Algorithm 1, where $i_j \neq i_1$, can only decrease the coordinate gradient value at i_1, and hence we get $\nabla_{i_1} F(z^{k+1}) \leq \nabla_{i_1} F(y^{k,1}) \leq 0$, as required for the coordinate i_1. It turns out that we may apply the same reasoning for every coordinate $i_j \in A^k$ and get that $\nabla_{i_j} F(z^{k+1}) \leq 0$. This shows that $\nabla_i F(z^{k+1}) \leq 0$ for all $i \in A^k$. Finally, by noticing that $\text{supp}(z^{k+1}) \backslash A^k \subseteq \text{supp}(z^k)$ and invoking Fact 4.12 for any $i \in \text{supp}(z^{k+1}) \backslash A^k$ if $\text{supp}(z^{k+1}) \backslash A^k \neq \emptyset$, we get $\nabla_i F(z^{k+1}) \leq 0$ for all $i \in \text{supp}(z^{k+1})$. This completes the induction step. To see that $\nabla_i F(z^k) \leq 0$ for all $i \in \text{supp}(z^k)$ and for all $k \geq 0$, it suffices to observe that the base case $k = 0$ trivially holds as $\text{supp}(z^0) = \emptyset$. □

We are now ready to prove Theorem 4.10. Suppose that

$$k \geq 2^{p-1} \left(\frac{\|\Delta\|_1}{\epsilon} \right)^p,$$

we need to show that

$$\|D^{1/q}[-\nabla F(z^k)]_+\|_1 \leq \epsilon\|\Delta\|_1.$$

Using Lemma 4.15, we get that

$$\sum_{j=0}^{k-1} \|[-\nabla F(z^j)]_+\|_p^p \leq p2^{p-1}(F(z^0) - F(z^k)) \leq p2^{p-1}(F(z^0) - F(z^*)).$$

Therefore there is $j' \leq k$ such that

$$\|[-\nabla F(z^{j'})]_+\|_p^p \leq \frac{p2^{p-1}(F(z^0) - F(z^*))}{k} = \frac{p2^{p-1}|F(z^*)|}{k}.$$

Note that since $\text{vol}(\text{supp}(z^*)) \leq \|\Delta\|_1$ according to Lemma 4.1, we trivially get that

$$F(z^0) - F(z^*) = |F(z^*)| \leq \|\Delta\|_1^{p+1}.$$

In the above, we upper bound $|F(z^*)|$ using the fact that $|F(z^*)| = \frac{1}{p}\|f^*\|_p^p$ where f^* is the optimal solution for Problem (4.9). Since the total amount of source mass is $\|\Delta\|_1$, the maximum amount of flow over any edge is at most $\|\Delta\|_1$, and by Lemma 4.1 we know that $|\text{supp}(f^*)| \leq \|\Delta\|_1$, therefore $\|f^*\|_p^p \leq \|\Delta\|_1^{p+1}$. It follows that our choice of k satisfies

$$k \geq 2^{p-1}\|\Delta\|_1^{p+1} \cdot \frac{1}{\epsilon^p\|\Delta\|_1} \geq p2^{p-1}|F(z^*)| \cdot \frac{1}{\epsilon^p\|\Delta\|_1}.$$

This means that

$$\|[-\nabla F(z^{j'})]_+\|_p^p \leq \epsilon^p\|\Delta\|_1,$$

and hence

$$\|[-\nabla F(z^{j'})]_+\|_p \leq \epsilon\|\Delta\|_1^{1/p}.$$

Using the fact that $\text{supp}(z^{j'}) \subseteq \text{supp}(z^*)$ and the generalized Hölder inequality we get

$$\begin{aligned}
\|D^{1/q}[-\nabla F(z^{j'})]_+\|_1 &= \sum_{i\in\text{supp}(z^*)} d_i^{1/q}[-\nabla_i F(z^{j'})]_+ \\
&\leq \left(\sum_{i\in\text{supp}(z^*)} d_i\right)^{1/q} \left(\sum_{i\in[n]} [-\nabla_i F(z^{j'})]_+^p\right)^{1/p} \\
&= \text{vol}(\text{supp}(z^*))^{1/q}\|[-\nabla F(z^{j'})]_+\|_p \leq \|\Delta\|_1^{1/q}\epsilon\|\Delta\|_1^{1/p} = \epsilon\|\Delta\|_1.
\end{aligned}$$

Since $\|D^{1/q}[-\nabla F(z^k)]_+\|_1$ measures the total amount of excess mass with the flows defined according to z^k, this quantity is monotonically deceasing in k. Therefore we have

$$\|D^{1/q}[-\nabla F(z^k)]_+\|_1 \leq \epsilon\|\Delta\|_1, \ \forall k \geq 2^{p-1}\left(\frac{\|\Delta\|_1}{\epsilon}\right)^p$$

as required. This finishes the proof of Theorem 4.10.

4.5.2 Proof of Theorem 4.8

Consider our diffusion setting as described in Section 4.2. Let K satisfy the required condition in Theorem 4.10 with $\epsilon = \beta/4$. Then we have that

$$\|D^{1/q}[-\nabla F(z^K)]_+\|_1 \leq \frac{\mathrm{vol}(C)}{2}.$$

It follows that

$$\begin{aligned}
-\frac{\mathrm{vol}(C)}{2} &\leq \sum_{i\in C} d_i^{1/q}\nabla_i F(z^K) \\
&= \sum_{i\in C}\left(d_i - \Delta_i + \sum_{j\sim i}\left|\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right|^{q-1}\mathrm{sign}\left(\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right)\right) \\
&= \mathrm{vol}(C) - \Delta(C) + \sum_{(i,j)\in\partial(C)}\left|\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right|^{q-1}\mathrm{sign}\left(\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right) \\
&= -\mathrm{vol}(C) + \sum_{(i,j)\in\partial(C)}\left|\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right|^{q-1}\mathrm{sign}\left(\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right),
\end{aligned}$$

and hence

$$\sum_{(i,j)\in\partial(C)}\left|\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right|^{q-1}\mathrm{sign}\left(\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right) \geq \frac{\mathrm{vol}(C)}{2}.$$

This implies that by

$$\|BD^{-1/q}z^K\|_q^q \geq \sum_{(i,j)\in\partial(C)}\left|\frac{z_i^K}{d_i^{1/q}} - \frac{z_j^K}{d_j^{1/q}}\right|^q \geq |\partial(C)|\left(\frac{\mathrm{vol}(C)}{2|\partial(C)|}\right)^p = \frac{\mathrm{vol}(C)^p}{2^p|\partial(C)|^{p-1}}.$$

Therefore we have that

$$\|Bx^K\|_q^{q-1} = \|BD^{-1/q}z^K\|_q^{q-1} \geq \frac{\mathrm{vol}(C)}{2|\partial(C)|^{1/q}}.$$

For each $e = (i,j) \in E$ define

$$\tilde{l}(e) = \begin{cases} \max\left(\dfrac{1}{\mathrm{vol}(C)^{1/q}}, \dfrac{|x_i^K - x_j^K|}{\|Bx^K\|_q}\right), & \text{if } |x_i^K - x_j^K| > 0, \\ 0, & \text{othereise.} \end{cases}$$

Then

$$\begin{aligned}
&\sum_{e=(i,j)\in E} |x_i^K - x_j^K| \cdot \tilde{l}(e)^{q-1} \\
&\leq \sum_{e:\tilde{l}(e)>\frac{1}{\mathrm{vol}(C)^{1/q}}} |x_i^K - x_j^K| \cdot \frac{|x_i^K - x_j^K|^{q-1}}{\|Bx^K\|_q^{q-1}} + \sum_{e:\tilde{l}(e)=\frac{1}{\mathrm{vol}(C)^{1/q}}} \frac{\|Bx^K\|_q}{\mathrm{vol}(C)^{1/q}} \cdot \frac{1}{\mathrm{vol}(C^{(q-1)/q}} \\
&\leq \|Bx^K\|_q + \frac{2}{\beta}\|Bx^K\|_q \leq \frac{3}{\beta}\|Bx^K\|_q.
\end{aligned}$$

Because $F(x^0) = 0$, we know that $F(x^K) < 0$ and this means that

$$(\Delta - d)^T x^K \geq \frac{1}{q}\|Bx^K\|_q^q.$$

Hence, let $S_h = \{i : x_i^K \geq h\}$, we have that

$$\begin{aligned}
\frac{\int_0^\infty \sum_{e\in\partial(S_h)} \tilde{l}(e)^{q-1} dh}{\int_0^\infty (\Delta(S_h) - \mathrm{vol}(S_h)) dh} &= \frac{\sum_{e=(i,j)} |x_i^K - x_j^K| \cdot \tilde{l}(e)^{q-1}}{(\Delta - d)^T x^K} \\
&\leq \frac{\frac{3}{\beta}\|Bx^K\|_q}{\frac{1}{q}\|Bx^K\|_q^q} = \frac{3q}{\beta\|Bx^K\|_q^{q-1}} \leq \frac{12|\partial(C)|^{1/q}}{\beta\mathrm{vol}(C)}.
\end{aligned}$$

This means that there must exist some h with S_h non-empty and

$$\frac{\sum_{e\in\partial(S_h)} \tilde{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)} \geq \frac{12|\partial(C)|^{1/q}}{\beta\mathrm{vol}(C)}.$$

All it remains is to connect the left hand side in the above inequality to the conductance of S_h. For the denominator, since the source mass has density at most $\delta = 2/\alpha$ at any node, we get $\Delta(S_h) - \mathrm{vol}(S_h) \leq 2\mathrm{vol}(s_h)/\alpha$. For the numerator, since any edge $e = (i, j)$ crossing a level cut S_h must have x_i^K, x_j^K on different sides of h, this means that $x_i^K \neq x_j^K$ and hence $\tilde{l}(e)$ is at least $1/\mathrm{vol}(C)^{1/q}$. This gives

$$\sum_{e\in\partial(S_h)} \tilde{l}(e)^{q-1} \geq \frac{|\partial(S_h)|}{\mathrm{vol}(C)^{(q-1)/q}}.$$

Therefore we get

$$\Phi(S_h) = \frac{|\partial(S_h)|}{\mathrm{vol}(S_h)} \leq \frac{\mathrm{vol}(C)^{(q-1)/q} \sum_{e\in\partial(S_h)} \tilde{l}(e)^{q-1}}{\frac{\alpha}{2}(\Delta(S_h) - \mathrm{vol}(S_h))} \leq \frac{24|\partial(C)|^{1/q}}{\alpha\beta\mathrm{vol}(C)^{1/q}} = \frac{24\Phi(C)^{1/q}}{\alpha\beta}.$$

By substituting $\epsilon = 4/\beta$ and using Assumption 4.3 that $\|\Delta\|_1 \leq 2\mathrm{vol}(C)/\beta$ in the lower bound for K in Theorem 4.10, this proves Theorem 4.8.

4.5.3 Proof of Theorem 4.11

First of all, based on Equation (4.11) it is easy to see that when $q = 2$ we have

$$|\nabla_i F(z + te_i) - \nabla_i F(z)| \leq |t|, \ \forall i \in [n], \forall z \in \mathbb{R}^n, \forall t \in \mathbb{R}. \tag{4.16}$$

Compared with Lemma 4.13, the constant in the front of $|t|$ on the right hand side improves from 2 to 1. This is the reason why in Algorithm 2 we use a slightly larger step-size for the coordinate-wise updates. Then, by following the same lines of reasoning as in the proofs of Lemma 4.14 and Lemma 4.15, respectively, we get that

$$F\Big(z - \nabla_i F(z) \cdot e_i\Big) \leq F(z) - \frac{|\nabla_i F(z)|^2}{2}, \ \forall i \in [n], \forall z \in \mathbb{R}^n,$$

and consequently, if z^k is the k^{th} iterate generated by Algorithm 2, then

$$F(z^{k+1}) \leq F(z^k) - \frac{1}{2}\big\|[-\nabla F(z^k)]_+\big\|_2^2. \tag{4.17}$$

In addition, when $q = 2$ the function $F(z)$ is twice differentiable, and that

$$\nabla^2 F(z) = D^{-1/2} B^T B D^{-1/2}.$$

Because the iterates z^k generated by Algorithm 2 satisfies $\text{supp}(z^k) \subseteq \text{supp}(z^*)$ for all k, and we always have that $|\text{supp}(z^*)| \leq n - 1$, the function $F(z)$ is strongly convex in the subspace of $\mathbb{R}^n$ spanned by $\{z^k\}_{k \geq 0}$, with a strong convexity parameter σ that satisfies

$$\sigma \geq \frac{1}{d_{\max}(\bar{S})} \frac{\Phi^*(\bar{S})^2}{2},$$

where

$$\begin{aligned} \bar{S} &= \text{supp}(z^*), \\ d_{\max}(\bar{S}) &= \max_{i \in \bar{S}} \deg(i), \\ \Phi^*(\bar{S}) &= \min_{S' \subseteq \bar{S}} \Phi(S'). \end{aligned}$$

The above inequality follows the relation between the local Cheeger constant $\Phi^*(S)$ and the smallest Dirichlet eigenvalue $\lambda_{\min}(B_S^T B_S)$, where B_S denotes the sub-matrix of B whose columns correspond to the nodes in S. In particular,

$$\lambda_{\min}(B_S^T B_S) \geq \frac{\Phi^*(S)^2}{2},$$

see, for example, [36]. Due to $F(z)$ being strongly convex over the subspace spanned by $\{z^k\}_{k\geq 0}$, we have that (when $q=2$)

$$F(z^k)-F(z^*)\geq \frac{\Phi^*(\bar{S})^2}{4d_{\max}(\bar{S})}\|z^k-z^*\|_2^2. \tag{4.18}$$

Denote $\delta_k = F(z^k)-F(z^*)$. By (4.17), we have

$$\delta_{k+1}\leq \delta_k - \frac{1}{2}\left\|[-\nabla F(z^k)]_+\right\|_2^2.$$

By Proposition 4.16, we know that $z^k\leq z^{k+1}\leq z^*$ for all $k\geq 0$, this means that $z^*-z^k\geq 0$ for all $k\geq 0$. Using this and the convexity of F we get that

$$\delta_k\leq \nabla F(z^k)^T(z^k-z^*)\leq [-\nabla F(z^k)]_+^T(z^*-z^k)\leq \left\|[-\nabla F(z^k)]_+\right\|_2\left\|z^*-z^k\right\|_2,$$

which means that

$$\left\|[-\nabla F(z^k)]_+\right\|_2^2\geq \frac{\delta_k^2}{\|z^*-z^k\|_2^2}.$$

Therefore

$$\delta_{k+1}\leq \delta_k-\frac{1}{2\|z^*-z^k\|_2^2}\delta_k^2=\delta_k\left(1-\frac{\delta_k}{2\|z^*-z^k\|_2^2}\right).$$

Using induction and (4.18) we get

$$\begin{aligned}
F(z^k)-F(z^*)&\leq (F(z^0)-F(z^*))\prod_{j=0}^{k-1}\left(1-\frac{F(z^j)-F(z^*)}{2\|z^*-z^j\|_2^2}\right)\\
&\leq \left(1-\frac{\Phi^*(\bar{S})^2}{8d_{\max}(\bar{S})}\right)^k(F(z^0)-F(z^*))\\
&\leq \exp\left(-\frac{\Phi^*(\bar{S})^2}{8d_{\max}(\bar{S})}\cdot k\right)(F(z^0)-F(z^*)).
\end{aligned}$$

Use the same reasoning as before, for $p=q=2$,

$$F(z^0)-F(z^*)=|F(z^*)|=\frac{1}{2}\|f^*\|_2^2\leq\frac{1}{2}\|\Delta\|_1^3.$$

It follows that for we can get $F(z^K)-F(z^*)\leq\epsilon'$ for

$$K\geq\frac{8d_{\max}(\bar{S})}{\Phi^*(\bar{S})^2}\log\left(\frac{\|\Delta\|_1^3}{\epsilon'}\right)$$

Take $\epsilon' = \epsilon^2\|\Delta\|_1/2$ so we have

$$F(z^K) - F(z^*) \leq \frac{\epsilon^2\|\Delta\|_1}{2}.$$

Then by (4.17) this means that

$$\|[-\nabla F(z^{K+1})]_+\|_2^2 \leq \epsilon^2\|\Delta\|_1,$$

or equivalently,

$$\|[-\nabla F(z^{K+1})]_+\|_2 \leq \epsilon\|\Delta\|_1^{1/2}.$$

Using the fact that $\operatorname{supp}(z^k) \subseteq \operatorname{supp}(z^*)$ and the generalized Hölder inequality we get

$$\begin{aligned}
\|D^{1/2}[-\nabla F(z^{K+1})]_+\|_1 &= \sum_{i\in\operatorname{supp}(z^*)} d_i^{1/2}[-\nabla_i F(z^{K+1})]_+ \\
&\leq \left(\sum_{i\in\operatorname{supp}(z^*)} d_i\right)^{1/2} \left(\sum_{i\in[n]}[-\nabla_i F(z^{K+1})]_+^2\right)^{1/2} \\
&= \operatorname{vol}(\operatorname{supp}(z^*))^{1/2}\|[-\nabla F(z^{K+1})]_+\|_2 \leq \|\Delta\|_1^{1/2}\epsilon\|\Delta\|_1^{1/2} = \epsilon\|\Delta\|_1.
\end{aligned}$$

This proves Theorem 4.11.

Chapter 5

p-Norm Flow Diffusion on Hypergraphs

5.1 Diffusion as an Optimization Problem

Similar to the p-norm flow diffusion problem on graphs, we formulate its hypergraph extension as a pair of convex optimization problems. Because our definition of a hypergrpah $H = (V, E, \mathcal{W})$ is much more general, and hence much more complex, than our definition of a graph $G = (V, E, w)$, the resulting optimization problems are quite heavy in terms of notation. In order to provide the reader with a smooth transition from graphs to hypergraphs, we start with simplified formulations in Section 5.1.1, which can be seen as a direct extension of the p-norm flow diffusion problem from the graph setting. Then in Section 5.1.2 we extend to the general formulations which involve additional variables and hyperparameter.

5.1.1 First Step: Direct Extension from Graphs

Given a hypergraph $H = (V, E, \mathcal{W})$ where $V = \{1, 2, \ldots, n\}$ and $E \subseteq 2^V$. Recall that $\mathcal{W} = \{(\vartheta_e, w_e)\}_{e \in E}$ consists of a set of submodular edge weight functions w_e and their corresponding normalization factor ϑ_e. In this section we assume that $\vartheta_e = 1$ for every $e \in E$. We will remove this assumption later.

Given the function $\Delta : V \to \mathbb{R}_+$ which specifies the source mass at each node, the p-norm flow diffusion on hypergraphs can be formulated as the following pair of convex

optimization problems

$$\begin{aligned} \min_{\phi \in \mathbb{R}_+^{|E|}, r_e \in \mathbb{R}^{|V|}, \forall e} \quad & \frac{1}{p}\|\phi\|_p^p \\ \text{s.t.} \quad & \Delta - \sum_{e \in E} r_e \leq d, \\ & r_e \in \phi_e B_e, \forall e \in E, \end{aligned} \tag{5.1}$$

and its dual formulation with q such that $1/q + 1/p = 1$,

$$\min_{x \geq 0} \ \frac{1}{q} \sum_{e \in E} f_e(x)^q + x^T(d - \Delta). \tag{5.2}$$

In the above, B_e denotes the **base polytope** [15] of the submodular edge weight function w_e, that is,

$$B_e = \{\rho_e \in \mathbb{R}^{|V|} \mid \rho_e(S) \leq w_e(S), \forall S \subseteq V, \text{ and } \rho_e(V) = w_e(V)\},$$

so

$$\phi_e B_e = \{\phi_e \rho_e : \rho_e \in B_e\},$$

and recall that d denotes the vector of node degrees, i.e.,

$$d(v) = \deg_H(v), \ \forall v \in V.$$

In Problem (5.2), the function f_e is the **Lovász extension** of w_e, given by

$$f_e(x) = \max_{\rho_e \in B_e} \rho_e^T x.$$

To see how the above pair of primal and dual formulations are direct extensions from the graph setting, we refer the reader to Problem (4.9) and Problem (4.8) for the closely related pair of primal and dual formulations on graphs for a quick comparison. We provide details below by discussing the interpretation of Problem (5.1) as a diffusion in the hypergraph. The readers should find the following interpretation of Problem (5.1) matches exactly with the interpretation of Problem (4.1) in Section 4.1. Before we start, we refer the reader to Section 3.4 for the definition of **proper flow routing** $r_e \in \mathbb{R}^{|V|}$ associated with an edge $e \in E$. The set of flow variables $\{r_e\}_{e \in E}$ appear in our primal formulation Problem (5.1) and serves as a basis for interpreting Problem (5.1) as a diffusion.

The Primal Problem: Capacity Constraint. As in the graph setting, we consider a diffusion on a hypergraph $H = (V, E, \mathcal{W})$ as the task of spreading mass from a small set

of nodes to a larger set of nodes. The function Δ will specify the amount of source mass starting at each node. At the same time, each node has a sink capacity which is the most amount of mass we allow at a node after spreading. Here, the define the sink capacity at a node to be the degree of that node. We sometimes refer to the density (of mass) at a node v as the ratio of the amount of mass at v over its degree $d(v)$. Naturally in a diffusion, we start with Δ having small support and high density, and the goal is to reach a state with bounded density enforced by the sink capacity. The total amount of mass in the hypergraph at any time is $\|\Delta\|_1 = \sum_{v\in V} \Delta(v)$, which remains constant throughout the diffusion as (proper) flow routing converses mass.

Given a set of proper flow routing r_e for $e \in E$, recall that $\sum_{e\in E} r_e(v)$ specifies the amount of net outflow at node v. Since each node $v \in V$ starts with $\Delta(v)$ source mass initially, the vector

$$m = \Delta - \sum_{e\in E} r_e$$

gives the amount of mass $m(v)$ at each node v after we spread mass according to the set of flow routing $\{r_e\}_{e\in E}$. That is,

$$m(v) = \Delta(v) - \sum_{e\in E} r_e(v)$$

is the sum of initial source mass and the nest amount of mass *routed out* of v. We say $\{r_e\}_{e\in E}$ is a set of **feasible flow routing** for a diffusion problem when $m(v) \le d(v)$ for all nodes, i.e. the mass obeys the sink capacity at each node. We say v's sink is **saturated** if $m(v) \ge d(v)$ and $\text{excess}(v) = \max\{m(v) - d(v), 0\}$ the **excess mass** at v. In order to force the spread of source mass we require that $\text{excess}(v) = 0$ for all v. This is precisely the first constraint of Problem (5.1).

The Primal Problem: Structural Constraint. So far we have not yet talked about how the edge weight functions $\{w_e\}_{e\in E}$ are used in our primal formulation Problem (5.1), and how they might affect diffusion. In particular, since higher-order relations among nodes within a hyperedge e are precisely captured by the corresponding weight function w_e, we need to find a way and inject the higher-order structural information modeled by $\{w_e\}_{e\in E}$ into the flow variables $\{r_e\}_{e\in E}$. Compared with the graph setting, this marks an important distinction for hypergraphs, as the edge weights in a graph $G = (V, E, w)$ are much simpler, and as a result the flow variables in the graph setting does not require the additional constraint which we discuss in the next. To that end, we consider

$$r_e = \phi_e \rho_e \text{ for some } \phi_e \in \mathbb{R}_+ \text{ and } \rho_e \in B_e,$$

where

$$B_e = \{\rho_e \in \mathbb{R}^{|V|} \mid \rho_e(S) \leq w_e(S), \forall S \subseteq V, \text{ and } \rho_e(V) = w_e(V)\}$$

is the base polytope for the submodular edge weight function w_e associated with hyperedge e. Recall that a proper edge weight function w_e satisfies $w_e(\emptyset) = w_e(e) = 0$, and thus, for any r_e, ϕ_e such that $r_e \in \phi_e B_e$, it is straightforward to verify that $r_e(v) = 0$ for every $v \notin e$ and $r_e(e) = r_e(V) = 0$. Therefore r_e defines a proper flow routing. For any $e \in E$, recall that $r_e(S)$ represents the net amount of mass that moves from S to $e \setminus S$ over the hyperedge e. Therefore, in the definition of B_e, the constraint $\rho_e(S) \leq w_e(S)$ for $S \subseteq e$ means that the directional flow $r_e(S)$ is upper bounded by a submodular function $\phi_e w_e(S)$. Intuitively, one may think of ϕ_e and ρ_e as the *scale* and the *shape* of r_e, respectively.

The Primal Problem: Objective Function. Just as in the graph setting, the goal of p-norm flow diffusion on hypergraphs is to find low cost flow routing $r_{e e \in E}$ such that the capacity constraint $\Delta - \sum_{e \in E} r_e \leq d$ and the structural constraint $r_e \in \phi_e B_e$ are satisfied. Because the variable $\phi_e \geq 0$ measures the scale of flow routing r_e, the p-norm cost for a set of flow routing $\{r_e\}_{e \in E}$ is $\|\phi\|_p^p = \sum_{e \in E} \phi_e^p$. The is the cost we try to minimize in Problem (5.1).

The Dual Problem. We will show that Problem (5.2) is the dual of Problem (5.1) later when we introduce the more general formulation for hypergraphs. Here, the dual problem can be interpreted in a similar way as in the graph setting. One can view the solution x as assigning heights to nodes, and the goal is to separate/cut the nodes with source mass from the rest of the hypergraph. Observe that the linear term in the dual objective function encourages raising x higher on the seed nodes and setting it lower on others. The cost $f_e(x)$ captures the discrepancy in node heights over a hyperedge e and encourages smooth height transition over adjacent nodes. The dual solution embeds nodes into the nonnegative real line. This embedding is what we consider as the output of diffusion and we will show that x nicely captures the local clustering structure in H.

5.1.2 Second Step: Additional Modification for Hypergraphs

Now we relax the assumption that the normalization factor $\vartheta_e = 1$ in the set of edge weights $\mathcal{W} = \{(\vartheta_e, w_e)\}_{e \in E}$. Furthermore, it turns out that neither Problem (5.1) nor Problem (5.2) is easy to solve from a numerical optimization point of view. Therefore, in order to apply efficient optimization algorithms, we regularize the problems by adding

additional variables and a hyperparameter $\sigma \geq 0$. The final primal formulation is

$$\begin{aligned} \min_{\phi \in \mathbb{R}_+^{|E|}, z \in \mathbb{R}_+^{|V|}, r_e \in \mathbb{R}^{|V|}, \forall e} \quad & \frac{1}{p} \sum_{e \in E} \vartheta_e \phi_e^p + \frac{\sigma}{p} \sum_{v \in V} d_v z_v^p \\ \text{s.t.} \quad & \Delta - \sum_{e \in E} \vartheta_e r_e \leq d + \sigma D z \\ & r_e \in \phi_e B_e, \forall e \in E, \end{aligned} \tag{5.3}$$

and its dual formulation with q such that $1/q + 1/p = 1$,

$$\min_{x \geq 0} \ \frac{1}{q} \sum_{e \in E} \vartheta_e f_e(x)^q + \frac{\sigma}{q} \sum_{v \in V} d_v x_v^q + x^T (d - \Delta). \tag{5.4}$$

Lemma 5.1. *Problem (5.4) is dual to Problem (5.3).*

We leave the proof of Lemma 5.1 to Section 5.5.2. Let us explain the physical interpretation of the additional variables $z \in \mathbb{R}_+^{|V|}$ in Problem (5.3) in the context of diffusion. For $\sigma > 0$, one can think of the variables z as introducing additional flexibility to the underlying diffusion process, by allowing each node v to hold more mass than its fixed capacity $d(v)$. Compared with Problem (5.1), this is reflected in the additional term $\sigma D z$ in the capacity constraint of Problem (5.3). However, holding additional mass comes at a cost, and this is reflected in the objective function of Problem (5.3). When $\sigma = 0$, Problem (5.3) reduces to Problem (5.1) (up to ϑ_e which simply assigns scalar weight a $e \in E$). In Section 5.3 we will show that having $\sigma > 0$ is crucial for us to write Problem (5.3) into an alternative but equivalent formulation which allows for efficient optimization algorithms. However, the additional flexibility in the underlying diffusion process due to $\sigma > 0$ also changes the underlying diffusion process, which may render the dual solution x^* not useful for local clustering. In Section 5.2 we show that as long as σ is not too large, then the resulting perturbation in the underlying diffusion process remains small, and hence x^* would have the same structural properties as if $\sigma = 0$.

Similar to the graph setting, both the primal and dual solutions are sparse. The number of nonzero entries in the solutions $\phi \in \mathbb{R}^{|E|}, z \in \mathbb{R}^{|V|}, r_e \in \mathbb{R}^{|V|}$ for all e, and $x \in \mathbb{R}^{|V|}$, does not depend on either $|V|$ or $|E|$. We state this property in Lemma 5.2 and leave its proof to Section 5.5.3

Lemma 5.2. *Let $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e \in E})$ be an optimal solution for Problem (5.3) and $\hat{x}$ an optimal solution for Problem (5.4). Then*

$$|\mathrm{supp}(\hat{\phi})| \leq \mathrm{vol}(\mathrm{supp}(\hat{x})) \leq \|\Delta\|_1.$$

Moreover, if $\sigma > 0$,

$$\mathrm{vol}(\mathrm{supp}(\hat{z})) = \mathrm{vol}(\mathrm{supp}(\hat{x})).$$

5.2 Local Hypergraph Clustering

In this section we discuss the performance of an optimal solution $\hat{x}$ of Problem (5.4) in the context of local hypergraph clustering. We consider a generic hypergraph $H = (V, E, \mathcal{W})$, where $\mathcal{W} = \{(\vartheta_e, w_e)\}_{e \in E}$ is a set of generalized edge weights, and each w_e is a submodular function. Given a set of seed nodes $S \subset V$, the goal of local hypergraph clustering is to identify a target cluster $C \subset V$ of low conductance, which contains or overlaps well with the seed nodes S. This generalizes the definition of local clustering over graphs which we considered in the last chapter. To the best of our knowledge, we are the first one to consider this problem for hypergraphs associated with general submodular edge weights. Following prior work on local clustering, we assume the existence of an unknown target cluster C with $\mathrm{vol}(C) \leq \mathrm{vol}(V)/2$ and conductance $\Phi(C)$. We will show that the sweep cut procedure (cf. Section 4.2.1) on an optimal solution $\hat{x}$ returns a cluster $\hat{C}$ having the same approximation guarantee as in the graph setting (cf. Theorem 4.4).

Similar to the graph setting, to specify a particular diffusion problem, we need to provide the source mass Δ. Given a set of seed nodes S, we pick a scalar δ and let

$$\Delta(i) = \begin{cases} \delta \cdot \deg(i), & \text{if } i \in S, \\ 0, & \text{otherwise.} \end{cases} \tag{5.5}$$

Note that this is exactly the same as how we initialize source mass in a graph, see Equation (4.3). We refer the reader to Section 4.2.1 for a discussion on how to pick δ. Here we pick δ in the same way. In addition, we make similar assumptions which we state below.

Assumption 5.3. The seed set $S \subset V$ satisfies

1. $\mathrm{vol}(S \cap C) \geq \alpha \mathrm{vol}(C)$ for some $\alpha \in (0, 1]$;
2. $\mathrm{vol}(S \cap C) \geq \beta \mathrm{vol}(S)$ for some $\beta \in (0, 1]$.

Assumption 5.4. The source mass Δ as specified in (5.5) satisfies $\delta = 3/\alpha$, which gives $\Delta(C) \geq 3\mathrm{vol}(C)$.

We also assume that the hyperparamter σ in Problem (5.3) and Problem (5.4) is not too large.

Assumption 5.5. The σ in Problem (5.3) and Problem (5.4) satisfies $0 \leq \sigma \leq \beta\Phi(C)/3$.

Theorem 5.6. *Under Assumption 5.3, Assumption 5.4, Assumption 5.5, the cluster $\hat{C}$ returned by the sweep cut procedure on an optimal solution $\hat{x}$ of Problem (5.4) satisfies*

$$\Phi(\hat{C}) \leq \frac{12\Phi(C)^{1/q}}{\alpha\beta}.$$

The proof of Theorem 5.6 shares the same high-level structure as the proof of Theorem 4.4 where we worked with diffusion on graphs. In what follows we provide a sketch of the proof. Since our hypergraph formulation introduces additional terms involving σ, the objective function of Problem (5.3) becomes the sum of the cost of flow and the cost of "excess mass". Therefore, an important part of the proof is showing that the the cost of flow remains large, even if our diffusion problem allows a certain amount of excess mass on each node due to $\sigma > 0$. This is stated in Lemma 5.7.

Lemma 5.7. *Assume the same diffusion setup and assumptions as assumed in Theorem 5.6. Let $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ be an optimal solution for Problem (5.3). Then*

$$\sum_{e\in E} \vartheta_e \hat{\phi}_e^p \geq \frac{\mathrm{vol}(C)^p}{\mathrm{vol}(\partial(C))^{p-1}}.$$

Let $\hat{x}$ be an optimal solution of Problem (5.4) and for $h > 0$ define the level cuts

$$S_h = \{v \in V \mid \hat{x}_v \geq h\}.$$

For each $e \in E$, define

$$\hat{l}(e) = \begin{cases} \max\left(\frac{1}{\mathrm{vol}(C)^{1/q}}, \frac{f_e(\hat{x})}{\sum_{e\in E}\vartheta_e\hat{\phi}_e^p}\right), & \text{if } f_e(\hat{x}) > 0, \\ 0, & \text{othereise.} \end{cases}$$

The follow claim follows from simple algebraic computations and the locality of solutions due to Lemma 5.2. It is a straightforward generalization from the graph setting.

Claim 5.8.

$$\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1} \leq \frac{4\left(\sum_{e\in E}\vartheta_e\hat{\phi}_e^p\right)^{1/q}}{\beta}.$$

Using the representation of the Lovász extension of submodular function as a Choquet integral, and the definition of w_e, we can show that

$$\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1} = \int_{h=0}^{\infty} \sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1} dh. \tag{5.6}$$

By the strong duality between Problem (5.3) and Problem (5.4) we know that

$$\hat{x}^T(\Delta - d) \geq \sum_{e\in E} \vartheta_e \hat{\phi}_e^p. \tag{5.7}$$

By combining Lemma 5.7, Claim 5.8, Equations (5.6) and (5.7) we get

$$\begin{aligned}\int_{h=0}^{\infty} \frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)} dh &= \frac{\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1}}{\hat{x}^T(d-\Delta)} \\ &\leq \frac{4}{\beta\left(\sum_{e\in E} \vartheta_e \hat{\phi}_e^p\right)^{1/p}} \leq \frac{4\mathrm{vol}(\partial(C))^{1/q}}{\beta\mathrm{vol}(C)}.\end{aligned}$$

This means that there must exist some h with S_h non-empty and

$$\frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)} \leq \frac{4\mathrm{vol}(\partial(C))^{1/q}}{\beta\mathrm{vol}(C)}.$$

The above is essentially the same (up to a constant) as (4.5). Then, by following the same steps as in the last part of Section 4.2.2 after (4.5), one gets

$$\Phi(S_h) = \frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S)}{\mathrm{vol}(S_h)} \leq \frac{12\Phi(C)^{1/q}}{\alpha\beta}.$$

We provide the details in Sections 5.5.4 to 5.5.6.

5.3 Optimization Algorithm and Complexity

We use a simple Alternating Minimization (AM) [20] method which applies to the primal formulation (5.3). The following Lemma 5.9 allows us to cast Problem (5.3) into an equivalent separable formulation amenable to the AM method.

Lemma 5.9. *The following problem is equivalent to Problem (5.3) for any $\sigma > 0$, in the sense that $(\hat{\phi}, z, \{\hat{r}_e\}_{e\in E})$ is optimal in Problem (5.3) for some $z \in \mathbb{R}^{|V|}$ if and only if $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{s_e\}_{e\in E})$ is optimal in Problem (5.8) for some $s_e \in \mathbb{R}^{|V|}$ for all $e \in E$.*

$$\begin{aligned} \min_{\phi\in\mathbb{R}_+^{|E|}, r_e, s_e\in\mathbb{R}^{|V|}, \forall e} \quad & \frac{1}{p}\sum_{e\in E}\vartheta_e\left(\phi_e^p + \frac{1}{\sigma^{p-1}}\left\|s_e - r_e\right\|_p^p\right) \\ \text{s.t.} \quad & \Delta - \sum_{e\in E}\vartheta_e s_e \le d \\ & s_{e,v} = 0, \forall v \notin e, \forall e \in E \\ & r_e \in \phi_e B_e, \forall e \in E. \end{aligned} \tag{5.8}$$

Corollary 5.10. *Let $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ be an optimal solution for Problem (5.8). Then the vector $\hat{x} \in \mathbb{R}^{|V|}$ defined by*

$$\hat{x}(v) = \left(\frac{\max\left\{0,\ \Delta(v) - \sum_{e\in E}\vartheta_e\hat{r}_e(v) - d(v)\right\}}{\sigma\cdot d(v)}\right)^{p-1},\ \forall v \in V,$$

is an optimal solution for Problem (5.4).

The proof of Lemma 5.9 follows from defining

$$z = \frac{1}{\sigma}D^{-1}\left[\Delta - \sum_{e\in E}\vartheta_e\hat{r}_e - d\right]_+$$

given $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ optimal in Problem (5.8), and defining

$$s_e = \hat{r}_e + \sigma A_e\hat{z},\ \forall e \in E,$$

given $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ optimal in Problem (5.3), where $A_e \in \mathbb{R}^{|V|\times|V|}$ is a diagonal matrix such that $[A_e]_{v,v} = 1$ if $v \in e$ and 0 otherwise, and showing that these variables attain the optimal objective value in their respective problems. We leave the details to Section 5.5.7. For the local clustering problem, since we require using a solution to the dual formulation Problem (5.4), Corollary 5.10 shows that one can recover an optimal dual solution from an optimal solution of Problem (5.8).

To simplify notation, let us denote

$$\mathcal{C} = \left\{(\phi, r) \in \mathbb{R}^{|E|} \times \left(\bigotimes_{e\in E}\mathbb{R}^{|V|}\right) \;\middle|\; r_e \in \phi_e B_e,\ \forall e \in E\right\}$$

which is the product of $|E|$ convex cones where each cone is generated by the base polytope B_e. Here, we abuse the notation and write $r \in \bigotimes_{e\in E} \mathbb{R}^{|V|}$ to represent a vector r in $\mathbb{R}^{|V||E|}$, where each $r_e \in \mathbb{R}^{|V|}$ corresponds to a block in r indexed by $e \in E$. For a vector $r_e \in \mathbb{R}^{|V|}$, $r_{e,v} = r_e(v)$ is the entry in r_e that corresponds to $v \in V$. For a vector $a \in \mathbb{R}^{|V|}$, $[a]_+ = \max\{a, 0\}$ where the maximum is taken entry-wise. Finally, recall that we use $A_e \in \mathbb{R}^{|V|\times|V|}$ to denote the diagonal matrix such that $[A_e]_{v,v} = 1$ if $v \in e$ and 0 otherwise.

Algorithm 4 Alternating Minimization for Problem (5.8)

Initialization:

$$\phi^{(0)} = 0,\ r_e^{(0)} = 0,\ s_e^{(0)} = D^{-1}A_e\left[\Delta - d\right]_+, \forall e \in E.$$

For $k = 0, 1, 2, \ldots, K$ **do:**

$$\texttt{r-step:}\ (\phi^{(k+1)}, r^{(k+1)}) = \operatorname*{argmin}_{(\phi,r)\in\mathcal{C}} \sum_{e\in E} \vartheta_e \left(\phi_e^p + \frac{1}{\sigma^{p-1}}\|s_e^{(k)} - r_e\|_p^p\right)$$

$$\texttt{s-step:}\ s^{(k+1)} = \operatorname*{argmin}_{s\in\bigotimes_{e\in E}\mathbb{R}^{|V|}} \sum_{e\in E} \vartheta_e\|s_e - r_e^{(k+1)}\|_p^p$$
$$\text{s.t.}\quad \Delta - \sum_{e\in E}\vartheta_e s_e \le d$$
$$s_{e,v} = 0,\ \forall v \notin e, \forall e \in E$$

The AM method is given in Algorithm 4. We will discuss the sub-problems shortly, but let us first state the rate of convergence for Algorithm 4. Theorem 5.11 is a straightforward application of Theorem 3.2 in [20] to Problem (5.8), and we discuss details in Section 5.5.8.

Theorem 5.11 ([20]). *Let $\{\phi^{(k)}, r^{(k)}, s^{(k)}\}_{k\ge 0}$ be the sequence generated by Algorithm 4. Denote $g(\phi, r, s)$ the objective function of Problem (5.8) and g^* the optimal value with $p \ge 2$. Then for any $k \ge 1$,*

$$g(\phi^{(k)}, r^{(k)}, s^{(k)}) - g^* \le \frac{3\max\{g(\phi^{(0)}, r^{(0)}, s^{(0)}) - g^*, LR^2\}}{k},$$

where

$$R = \max_{\substack{(\phi,r,s)\in\mathcal{F} \\ (\hat{\phi},\hat{r},\hat{s})\in\mathcal{O}}} \left\{ \|\phi-\hat{\phi}\|_2^2 + \sum_{e\in E}\|r_e-\hat{r}_e\|_2^2 + \sum_{e\in E}\|s_e-\hat{s}_e\|_2^2 \;\middle|\; g(\phi,r,s) \le g(\phi^{(0)},r^{(0)},s^{(0)}) \right\},$$

$$L = \frac{(p-1)\vartheta_{\max}^{2/p}\|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p}\sigma^{p-1}},$$

where $\mathcal{F}$ *and* $\mathcal{O}$ *denote the feasible set and set of optimal solutions, respectively, and*

$$\vartheta_{\max} = \max_{e\in E}\vartheta_e,\ d_{\min} = \min_{v\in\mathrm{supp}(\Delta)} d_v.$$

Observe that the complexity of optimization given in Theorem 5.11 with respect to p (mostly notably in the expression of L which increases with p) and the clustering performance given in Theorem 5.6 with respect to q, where $1/p+1/q=1$, show a natural tradeoff between computation and accuracy in the context of local hypergraph clustering. Unlike in the graph setting, here we do not establish a close correspondence between the total time complexity to compute an approximate solution and the clustering guarantee achievable by such an approximate solution (e.g. see Theorem 4.8 for the desired result in the graph setting). This is because we could not prove that Algorithm 4 has a strongly local running time, due to some intricate complexities introduced by the hyperedges. Rigorously proving that Problem (5.8) and more broadly other related optimization problems on hypergraphs can be solved to ϵ-accuracy in strongly local running time is still an open problem, which is part of an on-going project at the time this thesis is written. Empirically, our experiments in Section 5.4.1 indicate that Algorithm 4 indeed behaves like a local algorithm whose complexity does not depend on the size of the input hypergraph.

Finally, we would like to point out that reformulating Problem (5.3) as Problem (5.8) for $\sigma > 0$ turns out to be a crucial step from an algorithmic point of view. If $\sigma = 0$, then the primal formulation Problem (5.3) has complicated coupling constraints which are hard to deal with. In this case, one has to resort to the dual formulation Problem (5.4). However, Problem (5.4) has a nonsmooth objective function, which prohibits applicability of efficient first-order optimization methods for smooth objective functions. Even though the subgradient method may be applied, we have observed empirically that its convergence rate is extremely slow for our problem, and early stopping results in a poor quality output.

5.3.1 Alternating Minimization Sub-Problem: the `s-step`

We start with Lemma 5.12 which gives the closed-form solution for the `s-step` in Algorithm 4. It follows from projecting each s_e separately onto an affine space, and we provide details in Section 5.5.9.

Lemma 5.12. *The optimal solution to the following problem*

$$\begin{aligned} \min_{s \in \bigotimes_{e\in E} \mathbb{R}^{|V|}} & \sum_{e\in E} \vartheta_e \|s_e - r_e\|_p^p \\ \text{s.t.} \quad & \Delta - \sum_{e\in E} \vartheta_e s_e \leq d \\ & s_{e,v} = 0, \ \forall v \notin e, \forall e \in E \end{aligned} \tag{5.9}$$

is given by

$$s_e^* = r_e + A_e D^{-1} \Big[\Delta - \sum_{e'\in E} \vartheta_{e'} r_{e'} - d\Big]_+, \ \forall e \in E. \tag{5.10}$$

5.3.2 Alternating Minimization Sub-Problem: the `r-step`

For the `r-step` sub-problem with respect to (ϕ, r),

$$(\phi^{(k+1)}, r^{(k+1)}) := \operatorname*{argmin}_{(\phi,r)\in\mathcal{C}} \sum_{e\in E} \vartheta_e \left(\phi_e^p + \frac{1}{\sigma^{p-1}} \|s_e^{(k)} - r_e\|_p^p\right),$$

note that it decomposes into $|E|$ independent problems that can be minimized separately. That is, for $e \in E$, we have

$$\begin{aligned} (\phi_e^{(k+1)}, r_e^{(k+1)}) &= \operatorname*{argmin}_{\phi_e\geq 0, r_e \in \phi_e B_e} \vartheta_e \phi_e^p + \frac{1}{\sigma^{p-1}} \vartheta_e \|s_e^{(k)} - r_e\|_p^p \\ &= \operatorname*{argmin}_{\phi_e\geq 0, r_e \in \phi_e B_e} \frac{1}{p} \phi_e^p + \frac{1}{p\sigma^{p-1}} \|s_e^{(k)} - r_e\|_p^p. \end{aligned} \tag{5.11}$$

Problem (5.11) is strictly convex so it has a unique minimizer.

We focus on $p = 2$ first. In this case, Problem (5.11) can be solved in sub-linear time using either the conic Frank-Wolfe algorithm or the conic Fujishige-Wolfe minimum norm algorithm in [76]. Notice that the dimension of of the variables in Problem (5.11) is the

size of the corresponding hyperedge e. Therefore, as long as the hyperedge is not extremely large, we can easily obtain a good update $(\phi_e^{(k+1)}, r_e^{(k+1)})$.

If the base polytope B_e has a special structure, for example, if the hyperedge weight function w_e models **unit cut-cost**, i.e. $w_e(S) = 1$ for any $S \cap e \in 2^e \setminus \{\emptyset, e\}$ and 0 otherwise, then an exact solution for Problem (5.11) can be computed in time $O(|e| \log |e|)$ [76], where the time complexity is dominated by sorting an array of length $|e|$. For completeness we transfer the algorithmic details from [76] to our setting in Algorithm 5. The basic idea is to find optimal dual variables achieving dual optimality, and then recover primal optimal solution from the dual. We refer the reader to [76] for details. Given $e \in E$, $s_e \in \mathbb{R}^{|V|}$, and $a, b \in \mathbb{R}$, denote

$$e_{\geq}(a) := \{v \in e \mid s_{e,v} \geq \sigma a\} \quad \text{and} \quad e_{\leq}(b) := \{v \in e \mid s_{e,v} \leq \sigma b\}.$$

Define

$$\gamma(a, b) := a - b + \sum_{v \in e_{\geq}(a)} \sigma \left(a - \frac{s_{e,v}}{\sigma}\right).$$

Algorithm 5 An Exact Projection Algorithm for Problem (5.11) ($p = 2$, base polytope B_e corresponds to unit cut-cost w_e) [76]

Input: e, s_e.
$a \leftarrow \max_{v \in e} s_{e,v}/\sigma, \quad b \leftarrow \min_{v \in e} s_{e,v}/\sigma$
While true:
  $w_a \leftarrow \sigma\,|e_{\geq}(a)|, \quad w_b \leftarrow \sigma\,|e_{\leq}(b)|$
  $a_1 \leftarrow \max_{v \in e \setminus e_{\geq}(a)} s_{e,v}/\sigma, \quad b_1 \leftarrow b + (a - a_1) w_a / w_b$
  $b_2 \leftarrow \min_{v \in e \setminus e_{\leq}(b)} s_{e,v}/\sigma, \quad a_2 \leftarrow a - (b_2 - b) w_b / w_a$
  $i^* \leftarrow \operatorname{argmin}_{i \in \{1,2\}} b_i$
  If $a_{i^*} \leq b_{i^*}$ **or** $\gamma(a_{i^*}, b_{i^*}) \leq 0$ **break**
  $a \leftarrow a_{i^*}, \quad b \leftarrow b_{i^*}$
$a \leftarrow a - \gamma(a, b) w_b / (w_a w_b + w_a + w_b), \quad b \leftarrow b + \gamma(a, b) w_a / (w_a w_b + w_a + w_b)$
For $v \in e$ **do**:
  If $v \in e_{\geq}(a)$ **then** $r_{e,v} \leftarrow s_{e,v} - \sigma a$
  Else if $v \in e_{\leq}(b)$ **then** $r_{e,v} \leftarrow s_{e,v} - \sigma b$
  Else $r_{e,v} \leftarrow 0$
Return: r_e

Next we discuss the case $p > 2$ in Problem (5.11). The dual of Problem (5.11) is

$$\min_{y_e} \frac{1}{q} f_e(y_e)^q + \frac{\sigma}{q} \|y_e\|_q^q - y_e^T s_e^{(k)}. \tag{5.12}$$

Let (ϕ_e^*, r_e^*) and y_e^* be optimal solutions of Problem (5.11) and Equation (5.12), respectively. Then one has

$$r_e^* = s_e^{(k)} - \sigma(y_e^*)^{q-1} \text{ and } \phi_e^* = \left((r_e^*)^T y_e^*\right)^{1/q}.$$

The derivation of Equation (5.12) and the above relations between (ϕ_e^*, r_e^*) and y_e^* follow from the same reasoning and algebraic computations used in the proofs of Lemma 5.1 and Lemma 5.13. Therefore, we can use the subgradient method to compute y_e^* first and then recover ϕ_e^* and r_e^*. For simple cases like the unit cut-cost where the base polytope B_e has a special structure, a similar approach as Algorithm 5 can be adopted to obtain a nearly exact solution (e.g. up to finite precision machine epsilon), by modifying Steps 2-6 to work with general ℓ_p-norm and replacing Step 10 with binary search. See Algorithm 6 for details.

Notations in Algorithm 6. To simplify notation in Algorithm 6, for $c \in \mathbb{R}$ and $p > 0$, c^p is to be interpreted as $c^p := |c|^p \text{sign}(c)$, where we treat $\text{sign}(0) := 0$. For $q = p/(p-1)$, we define

$$\gamma_p(a, b) := (a - b)^{q-1} + \sum_{v \in e_{\geq}(a^{q-1})} \sigma \left(a^{q-1} - \frac{s_{e,v}}{\sigma} \right).$$

Algorithm 6 An ℓ_p-Projection Algorithm for Problem (5.11) ($p > 2$, base polytope B_e corresponds to unit cut-cost w_e)

1: **Input:** e, s_e.
2: $a \leftarrow \max_{v \in e}(s_{e,v}/\sigma)^{p-1}$, $\quad b \leftarrow \min_{v \in e}(s_{e,v}/\sigma)^{p-1}$, $\quad q \leftarrow p/(p-1)$
3: **While** true:
4: $\quad w_a \leftarrow \sigma\, |e_{\geq}(a^{q-1})|$, $\; w_b \leftarrow \sigma\, |e_{\leq}(b^{q-1})|$
5: $\quad a_1 \leftarrow \max_{v \in e \setminus e_{\geq}(a^{q-1})}(s_{e,v}/\sigma)^{p-1}$, $\; b_1 \leftarrow (b^{q-1} + (a^{q-1} - a_1^{q-1}) w_a / w_b)^{p-1}$
6: $\quad b_2 \leftarrow \min_{v \in e \setminus e_{\leq}(b^{q-1})}(s_{e,v}/\sigma)^{p-1}$, $\quad a_2 \leftarrow (a^{q-1} - (b_2^{q-1} - b^{q-1}) w_b / w_a)^{p-1}$
7: $\quad i^* \leftarrow \text{argmin}_{i \in \{1,2\}}\, b_i$
8: $\quad$ **If** $a_{i^*} \leq b_{i^*}$ **or** $\gamma_p(a_{i^*}, b_{i^*}) \leq 0$ **break**
9: $\quad a \leftarrow a_{i^*}$, $\; b \leftarrow b_{i^*}$
10: Employ binary search for $\hat{a} \in [b, a]$ such that $\gamma_p(\hat{a}, \hat{b}) = 0$ while maintaining $\hat{b} = (b^{q-1} + (a^{q-1} - \hat{a}^{q-1}) w_a / w_b)^{p-1}$ and $\hat{b} \leq \hat{a}$
11: **For** $v \in e$ **do**:
12: $\quad$ **If** $v \in v \in e_{\geq}(\hat{a}^{q-1})$ **then** $\; r_{e,v} \leftarrow s_{e,v} - \sigma \hat{a}^{q-1}$
13: $\quad$ **Else if** $v \in e_{\leq}(\hat{b}^{q-1})$ **then** $\; r_{e,v} \leftarrow s_{e,v} - \sigma \hat{b}^{q-1}$
14: $\quad$ **Else** $\; r_{e,v} \leftarrow 0$
15: **Return:** r_e

5.4 Empirical Results

In this section we evaluate the performance of p-norm flow diffusion on hypergraphs for the local clustering task and the related node ranking task. We primarily focus on the case $p = 2$. The reason that we focus on $p = 2$ is that it already produces very good empirical performance. Unlike local clustering in the graph setting where a lower conductance almost always translates to a better clustering result, this isn't necessarily the case for hypergraphs, where the conductance of a cluster depends on the choice of edge weight functions $\{w_e\}_{e \in E}$. That is, the conductance of the same cluster can be very different under different sets of edge weights (recall that conductance is defined based on the edge weights). Consequently, for hypergraphs, we cannot always use conductance as a measure of clustering quality if different edge weights are used by different local clustering algorithms. It turns out that designing and employing an appropriate set of edge weights is much more important for practical applications. Therefore, since the case $p = 2$ is also computationally more efficient, we primarily focus on $p = 2$ and evaluate the performance under different sets of edge weights. Since we fix $p = 2$, we will simply refer to our method as **Hypergraph Flow Diffusion (HFD)**. Let us mention that, even when $p = 2$, the diffusion process modeled by Problem (5.3) and the corresponding diffusion method provided in Algorithm 4 are the first to work with hypergraphs associated with general submodular edge weights. As we will show in the experiments in this section, this can lead to significant performance improvement for the local clustering task over both synthetic and real-world hypergraphs.

The rest of the section is organized as follows. In Section 5.4.1 we show some empirical evidence that Algorithm 4 is a local method, which means that it does not necessarily need to touch the entire hypergraph before converging to an optimal solution. In Section 5.4.2 we carry out experiments on synthetic hypergraphs with varying target cluster conductance and varying hyperedge size. For the unit cut-cost setting, we show that HFD is more robust and has better performance when the target cluster is noisy; for a cardinality-based cut-cost setting, we show that the edge-size-independent approximation guarantee is important for obtaining good recovery results. In Section 5.4.3 we carry out experiments using real-world data. We show that HFD significantly outperforms existing state-of-the-art diffusion methods for both unit and cardinality-based cut-costs. Moreover, we provide a compelling example where specialized submodular cut-cost is necessary for obtaining good results.[1]

[1]Code that reproduces all results is available at https://github.com/s-h-yang/HFD.

5.4.1 Empirical Locality of Algorithm 4

As noted in Lemma 5.2, the number of nonzero entries in the optimal solutions is upper bounded by $\|\Delta\|_1$. In Figure 5.1 we plot the number of nodes having positive excess (which equals the number of nonzero entries in the dual variables x) at every iteration of Algorithm 4. Figure 5.1 indicates that empirically Algorithm 4 is strongly local, meaning that it works only on a small fraction of nodes (and their incident hyperedges) as opposed to producing dense iterates which requires touching the entire hypergraph. This important empirical observation has enabled our algorithm to scale to large datasets by simply keeping track of all active nodes and hyperedges. In our implementation of Algorithm 4, we only access a node and its neighbors whenever there is a positive amount of excess mass on that node. Theoretially, proving that the worst-case running time of Algorithm 4 depends only on the number of nonzero nodes at optimality as opposed to size of the entire hypergraph is still an open problem.

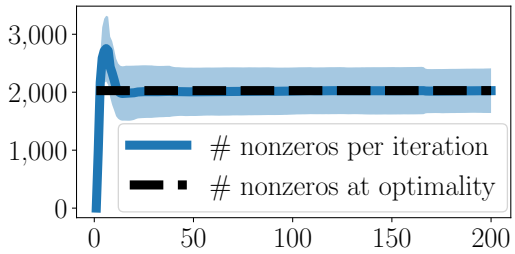
5.4.2 Experiments on Synthetic Hypergraphs

The generative model. We generalize the standard k-uniform hypergraph stochastic block model (kHSBM) [55] to allow different types of inter-cluster hyperedges appear with possibly different probabilities according to the cardinality of hyperedge cut. Let $V = \{1, 2, \ldots, n\}$ be a set of nodes and let $k \geq 2$ be the required constant hyperedge size. We consider kHSBM with parameters k, n, p, q_j, $j = 1, 2, \ldots, \lfloor k/2 \rfloor$. The model samples a k-uniform hypergraph according to the following rules:

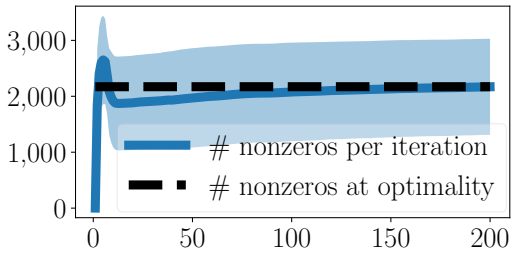
1. The community label $\sigma_i \in \{0, 1\}$ is chosen uniformly at random for $i \in V$;
2. Each size k subset $e = \{v_1, v_2, \ldots, v_k\}$ of V appears independently as a hyperedge with probability

$$\mathbb{P}(e \in E) = \begin{cases} p, & \text{if } \sigma_{v_1} = \sigma_{v_2} = \cdots = \sigma_{v_k}, \\ q_j, & \text{if } \min\{k - \sum_{i=1}^{k} \sigma_{v_i},\ \sum_{i=1}^{k} \sigma_{v_i}\} = j. \end{cases}$$

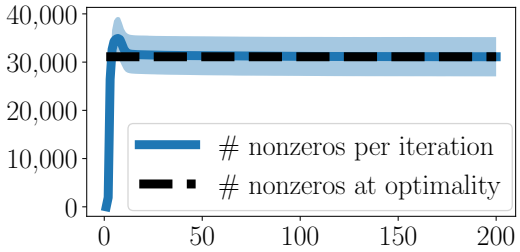
If $k = 3$ or all q_j's are the same, then we obtain the standard two-block kHSBM. We use this setting to evaluate HFD for unit cut-cost. If q_j's are different, then we obtain a cardinality-based kHSBM. In particular, when $q_1 \geq q_2 \geq \cdots \geq q_{\lfloor k/2 \rfloor}$, it models the scenario where hyperedges containing similar numbers of nodes from each block are rare, while small noises (e.g., hyperedges that have one or two nodes in one block and all the rest in the other block) are more frequent. We use $q_1 \gg q_j$, $j \geq 2$, to evaluate HFD for



(a) Cluster 12 - Gift Cards



(b) Cluster 18 - Magazine Subs.



(c) Cluster 24 - Prime Pantry

Figure 5.1: The blue solid line plots the number of nonzero entries in the dual variables $x \in \mathbb{R}^{|V|}$ over 200 iterations of Algorithm 4, when it is applied to solve Problem (5.8) on the Amazon-reviews hypergraph for local clustering. See Section 5.4.3 for details about the dataset. The error bars show standard deviation over 10 trials. In each trial we pick a different seed node and set the same amount of source mass. The black dashed line shows the average number of nonzero entries at optimality. The algorithm touches only a small fraction of nodes out the total 2,268,264 nodes in the Amazon-reviews dataset.

cardinality-based cut-cost. There are other random hypergraph models, for example the Poisson degree-corrected HSBM [33] that deals with degree heterogeneity and edge size heterogeneity. In our experiments we focus on kHSBM because it allows stronger control over hyperedge sizes.

Data generation. We generate four sets of hypergraphs using the generalized kHSBM described in the above. All hypergraphs have $n = 100$ nodes.

- *1st set of hypergraphs.* We generate the first set of hypergraphs with $k = 3$, constant $p = 0.0765$ and varying $q \in [0.0041, 0.0735]$. Recall that for $k = 3$ there is only one possible inter-cluster probability $q \equiv q_1$. We pick $p = 0.0765$ so the expected number of intra-cluster hyperedges is 1500 for each block of size 50. We set a wide range for q so that the interval covers both extremes, i.e., when the ground-truth target cluster

is very clean or very noisy. These hypergraphs are used to evaluate the performance of algorithms for the unit cut-cost setting when the target cluster conductance varies. Figure 5.2 uses the local clustering results on these hypergraphs.

- *2nd set of hypergraphs.* For the second set of hypergraphs, we vary $k \in \{3, 4, 5, 6\}$. Moreover, we set $q_2 = \cdots = q_{\lfloor k/2 \rfloor} = 0$, so every inter-cluster hyperedge contains a single node on one side and the rest on the other side. In this setting, separating the two ground-truth communities will incur a small penalty using the cardinality cut-cost, but a large penalty using the unit cut-cost. Therefore, methods that exploit appropriate cardinality-based cut-cost should perform better. The hypergraphs are sampled so that the conductance of a ground-truth community stays the same across different k's. We compute the conductance based on the unit cut-cost when generating the hypergraphs, because the scale of conductance based on the unit cut-cost is less affected by k than the scale of conductance based on the cardinality cut-cost. The second set of hypergraphs is used to evaluate the performance of algorithms for both unit and cardinality cut-costs when the hyperedge size varies. Figure 5.3 uses the local clustering results on these hypergraphs.

- *3rd set of hypergraphs.* For the third set of hypergraphs, we set $q_2 = \cdots = q_{\lfloor k/2 \rfloor} = 0$. We consider constant $k = 4$ or $k = 5$, constant p and varying q_1. These hypergraphs are used to evaluate the performance of algorithms for both unit and cardinality cut-costs when the target cluster conductance varies. Figure 5.4 and Figure 5.5 are based on the local clustering results on these hypergraphs.

Task and methods. We consider the local hypergraph clustering problem. We assume that we are given a *single seed node* and the goal is to recover all nodes having the same community label. Using a single seed node the most common practice for local clustering tasks. We test the performance of HFD with two other methods: (i) **Localized Quadratic Hypergraph Diffusions (LH)** [80], which can be seen as a hypergraph analogue of Approximate Personalized PageRank (APPR); (ii) **ACL** [8], which is used to compute APPR vectors on a standard graph obtained from reducing a hypergraph through star expansion [144]. There are other heuristic methods, such as first reducing a hypergraph to a graph by clique expansion [21] and then applying diffusion methods for standard graphs. We do not compare with this approach because clique expansion often results in a dense graph and consequently makes the computation very slow. Moreover, it has been shown in [80] that clique expansion did not offer significant performance improvement over star expansion.

Cut-costs (i.e. edge weights). We consider both *unit cut-cost*, i.e., $w_e(S) = 1$ if $S \cap e \neq \emptyset$ and $e \setminus S \neq \emptyset$, and *cardinality cut-cost* $w_e(S) = \min\{|S \cap e|, |e \setminus S|\}/\lfloor |e|/2 \rfloor$. HFD

that uses unit and cardinality cut-costs are denoted by **U-HFD** and **C-HFD**, respectively. LH also works with both unit and cardinality cut-costs and we specify them by **U-LH** and **C-LH**, respectively.

Parameters. For HFD, we initialize the source mass on the single seed node so that $\|\Delta\|_1$ is three times the volume of the target cluster (recall from Assumption 5.4 this is without loss of generality). We set the hyperparameter $\sigma = 0.01$ in Problem (5.8) which we solve by Algorithm 4. We tune the parameters for LH as suggested by the authors [80]. Specifically, LH has a regularization parameter κ and we let $\kappa = c \cdot r$ where r is the ratio between the number of seed node(s) and the size of the target cluster. We perform a binary search on c and find that $c = 0.35$ gives good results. An important parameter for LH is δ. When $\delta = 1$ it models unit cut-cost and when $\delta \geq 1$ it models cardinality-based cut-cost with an upper bound δ [80]. We consider both cases $\delta = 1$ (U-LH) and $\delta \geq 1$ (C-LH). In principle, for k-uniform hypergraphs LH should produce the same result for any $\delta \geq k$, so one could simply set $\delta = k$ for C-LH. However in our experiments we find that the δ value that gives the best clustering result can be much larger than k. In order to get the best performance out of C-LH, we run C-LH for $\delta = 2^i$, $i = 0, 1, \ldots, 12$. Among the 13 output clusters from C-LH we pick the one with the lowest conductance. For ACL, we use the same set of parameter values used in [80] because that parameter setting also produces good results in our experiments.

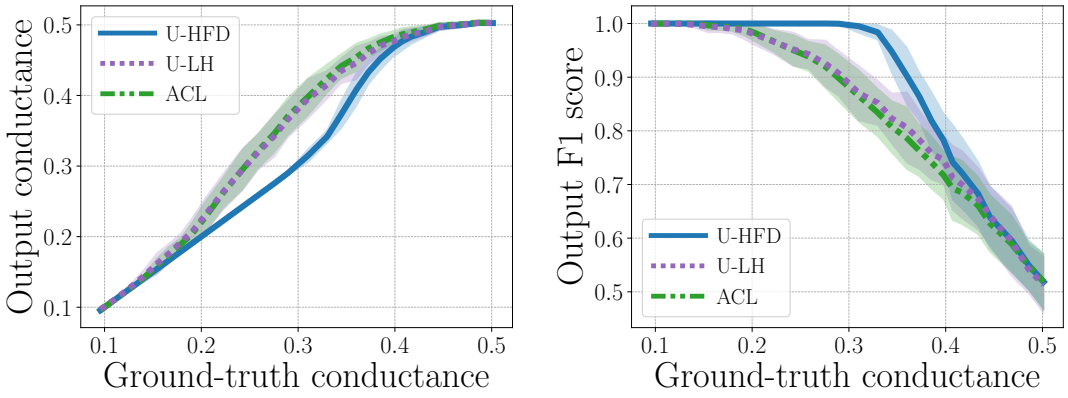


Figure 5.2: Conductance and F1 of the output cluster against the conductance of the ground-truth cluster on k-uniform hypergraphs with $k = 3$. The error bars show variation over 50 runs using different seed nodes. Both the ground-truth and the target conductance are computed using unit cut-cost.

Results. For each hypergraph, we randomly pick a ground-truth community as the target cluster. We run each method 50 times. Each time we choose a different node from

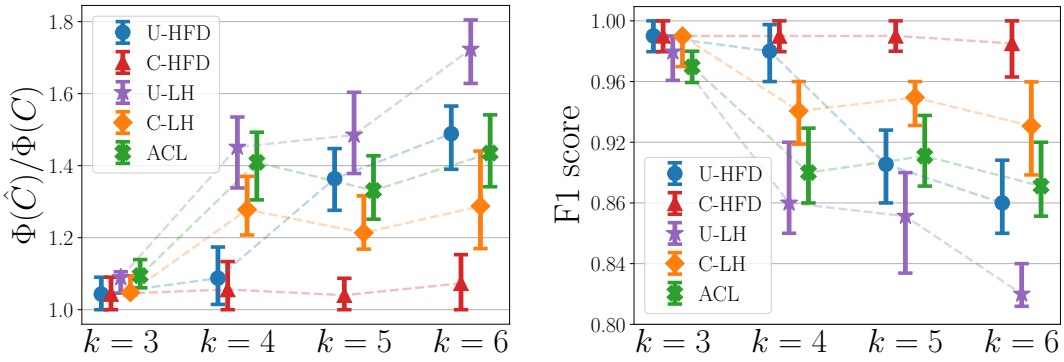


Figure 5.3: Ratio of output-to-target conductance (i.e. $\Phi(\hat{C})/\Phi(C)$ where $\hat{C}$ denotes the output cluster and C denotes the target cluster) and F1 on k-uniform hypergraphs

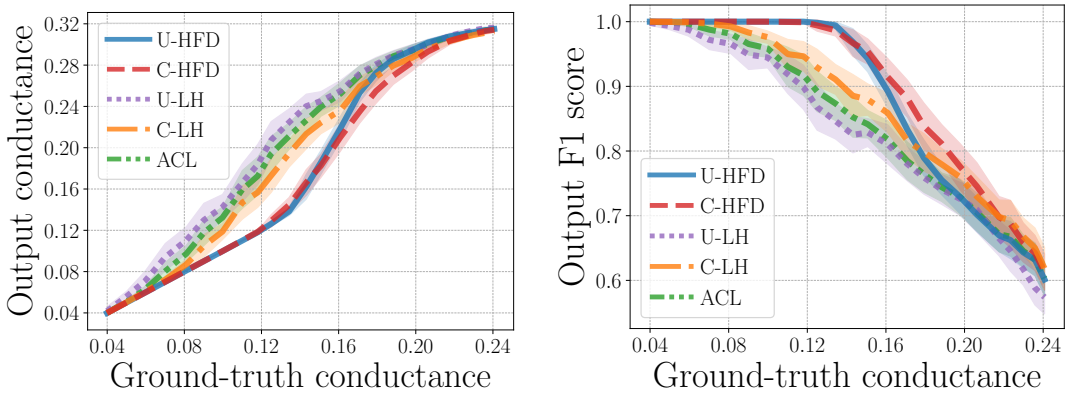


Figure 5.4: Conductance and F1 of the output cluster against the conductance of the ground-truth cluster on k-uniform hypergraphs with $k = 4$. The error bars show variation over 50 runs using different seed nodes. Both the ground-truth and the target conductance are computed using cardinality-based cut-cost.

the target cluster as the single seed node. Below we discuss the results obtained from adopting the unit cut-cost and cardinality cut-cost separately.

- *Unit cut-cost.* Figure 5.2 shows local clustering results when we fix $k = 3$ but vary the conductance of the target cluster (i.e., constant p but varying q_1). Observe that the performances of all methods become worse as the target cluster becomes noisier, but U-HFD has significantly better performance than both U-LH and ACL when the conductance of the target cluster is between 0.2 and 0.4. The reason that U-HFD performs better is in part because it requires a much weaker condition for the

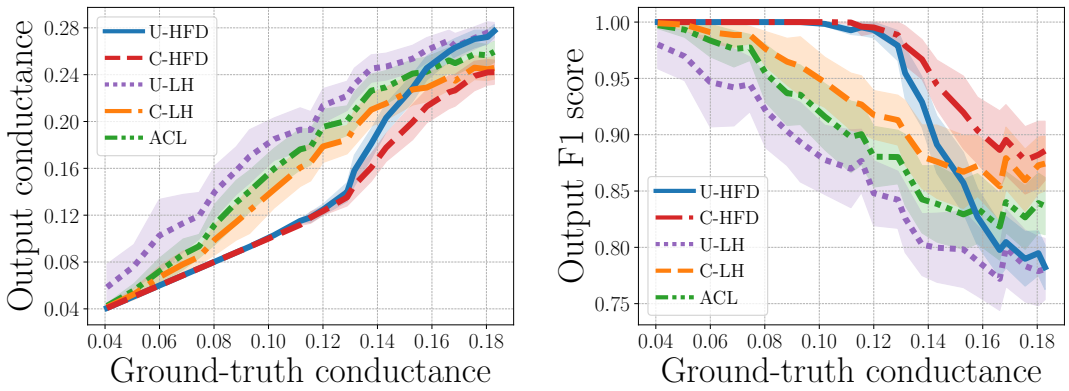


Figure 5.5: Conductance and F1 of the output cluster against the conductance of the ground-truth cluster on k-uniform hypergraphs with $k = 5$. The error bars show variation over 50 runs using different seed nodes. Both the ground-truth and the target conductances are computed using cardinality-based cut-cost.

theoretical guarantee in terms of conductance to hold. On the contrary, LH assumes an upper bound on the conductance of the target cluster [80]. This upper bound is dataset-dependent and could become very small in many cases, leading to poor practical performance. ACL with star expansion is a heuristic method and has no performance guarantee for hypergraphs.

- *Cardinality cut-cost.* Figure 5.3 shows the median (markers) and 25-75 percentiles (lower-upper bars) of the ratio between output conductance and ground-truth conductance (lower is better, and a ratio of 1 means the output cluster has the same conductance as the target cluster), and F1 scores for different methods for $k \in \{3, 4, 5, 6\}$. The target cluster for each k has conductance around 0.3. For $k = 3$, unit and cardinality cut-costs are equivalent, therefore all methods have similar performance. As k increases, cardinality cut-cost provides better performance than unit cut-cost in terms of both conductance and F1. However, since the theoretical approximation guarantee of C-LH depends on hyperedge size [80], there is a noticeable performance degradation for C-LH when we increase $k = 3$ to $k = 4$. On the other hand, the performance of C-HFD appears to be independent from k, which aligns with Theorem 5.6 where the bound is independent from the size of hyperedges. Figure 5.4 and Figure 5.5 show how the algorithms perform on k-uniform hypergraphs for $k = 4, 5$, respectively, as we vary the conductance of the target cluster. The plots show that as the target cluster becomes noisier, the performance of all methods degrades. However, C-HFD is better in terms of both conductance and F1 score, especially when

the target cluster is noisy but not complete noise (i.e., the ground-truth conductance is high but not too high). For $k = 5$ and in the high-conductance regime, methods that use unit cut-cost, e.g., U-HFD, have poor performance because they find low-conductance clusters based on the unit cut-cost as opposed to the cardinality cut-cost. In general, lower unit cut-cost conductance does not necessarily translates to lower cardinality-based conductance or higher F1 score.

5.4.3 Experiments on Real-world Hypergraphs

Datasets and ground-truth clusters. We describe the real-world hypergraphs used in the experiments below. In addition, Table 5.1 provides summary statistics about the hypergraphs. Table 5.2 includes the statistics of all ground truth clusters that we used in the experiments.

- *Amazon-reviews* [94, 120]. This is a hypergraph constructed from Amazon product review data, where each node represents a product. A set of products are connected by a hyperedge if they are reviewed by the same person. We use product category labels as ground truth cluster identities. In total there are 29 product categories. Because we are mostly interested in local clustering, we consider all clusters consisting of less than 10,000 nodes.

- *Trivago-clicks* [33]. The nodes in this hypergraph are accommodations/hotels. A set of nodes are connected by a hyperedge if a user performed "click-out" action during the same browsing session, which means the user was forwarded to a partner site. We use geographical locations as ground truth cluster identities. There are 160 such clusters. We consider all clusters in this dataset that consists of less than 1,000 nodes and has conductance less than 0.25.

- *Florida Bay food network* [77]. Nodes in this hypergraph correspond to different species or organisms that live in the Bay, and hyperedges correspond to transformed network motifs of the original dataset. Each species is labelled according its role in the food chain.

- *High-school-contact* [90, 33]. Nodes in this hypergraph represent high school students. A group of people are connected by a hyperedge if they were all in proximity of one another at a given time, based on data from sensors worn by students. We use the classroom to which a student belongs to as ground truth. In total there are 9 classrooms.

Table 5.1: Summary of real-world hypergraphs

Dataset	Number of nodes	Number of hyperedges	Maximum hyperedge size	Maximum node degree	Median / Mean hyperedge size	Median / Mean node degree
Amazon-reviews	2,268,231	4,285,363	9,350	28,973	8.0 / 17.1	11.0 / 32.2
Trivago-clicks	172,738	233,202	86	588	3.0 / 4.1	2.0 / 5.6
Florida-Bay	126	141,233	4	19,843	4.0 / 4.0	3,770.5 / 4,483.6
Microsoft-academic	44,216	22,464	187	21	3.0 / 5.4	2.0 / 2.7
High-school-contact	327	7,818	5	148	2.0 / 2.3	53.0 / 55.6
Oil-trade	229	100,639	4	16,394	4.0 / 4.0	175.0 / 1,757.9

- *Microsoft-academic* [110, 5]. The original co-authorship network is a subset of the Microsoft Academic Graph where nodes are authors and hyperedges correspond to a publication from those authors. We take the dual of the original hypergraph by converting hyperedges to nodes and nodes to hyperedges. After constructing the dual hypergraph, we removed all hyperedges having just one node and we kept the largest connected component. In the resulting hypergraph, each node represents a paper and is labelled by its publication venue. A set of papers are connected by a hyperedge if they share a common coauthor. We combine similar computer science conferences into four broader categories: Data (KDD, WWW, VLDB, SIGMOD), ML (ICML, NeurIPS), TCS (STOC, FOCS), CV (ICCV, CVPR).

Methods and parameters. For HFD, we use $\sigma = 0.0001$ for all the experiments. We set the total amount of source mass $\|\Delta\|_1$ at the seed node as a constant factor t times the volume of the target cluster. For Amazon-reviews, on the smaller clusters 1, 2, 3, 12, 18, we use $t = 200$; on the larger clusters 15, 17, 24, 25, we use $t = 50$. For both Trivago-clicks, High-school-contact and Microsoft-academic, we use $t = 3$. For Florida Bay food network, we use $t = 20, 10, 5$ for clusters 1, 2, 3, respectively. In all experiments, the choice of t is to ensure that the diffusion process will cover some part of the target and incur a high cost in the objective function. For LH and ACL, e use the parameters as suggested by the authors [80]. For both *-LH-2.0 and *-LH-1.4, we set $\gamma = 0.1$, $\rho = 0.5$, $\kappa = c \cdot r$ where r is the ratio between the number of seed node(s) and the size of the target cluster, and c is a tuning constant. For Amazon-reviews, we set $c = 0.025$ as suggested in [80]. For Microsoft-academic, Trivago-clicks, and Florida-Bay we also use $c = 0.025$ because it produces good results. For High-school-contact we select $c = 0.25$ after some tuning to make sure both *-LH-2.0 and *-LH-1.4 have good results. We set the parameters for ACL in exactly the same way as in [80]. We set $\delta = 1$ for U-LH-* and $\delta = \max_{e \in E} |e|$ for C-LH-*.

Experiments for unit and cardinality cut-costs. For each target cluster in each dataset, we run the methods multiple times, each time we use a different node as the

Table 5.2: Summary of ground-truth clusters used in the experiments

Dataset	Cluster	Size	Volume	Conductance
Amazon-reviews	1 - Amazon Fashion	31	3042	0.06
	2 - All Beauty	85	4092	0.12
	3 - Appliances	48	183	0.18
	12 - Gift Cards	148	2965	0.13
	15 - Industrial & Scientific	5334	72025	0.14
	17 - Luxury Beauty	1581	28074	0.11
	18 - Magazine Subs.	157	2302	0.13
	24 - Prime Pantry	4970	131114	0.10
	25 - Software	802	11884	0.14
Trivago-clicks	KOR - South Korea	945	3696	0.24
	ISL - Iceland	202	839	0.21
	PRI - Puerto Rico	144	473	0.25
	UA-43 - Crimea	200	1091	0.24
	VNM - Vietnam	832	2322	0.24
	HKG - Hong Kong	536	4606	0.24
	MLT - Malta	157	495	0.24
	GTM - Guatemala	199	652	0.24
	UKR - Ukraine	264	648	0.24
	SET - Estonia	158	850	0.23
Florida-Bay	Producers	17	10781	0.70
	Low-level consumers	35	173311	0.58
	High-level consumers	70	375807	0.54
Microsoft-academic	Data	15817	45060	0.06
	ML	10265	26765	0.16
	TCS	4159	10065	0.08
	CV	13974	38395	0.08
High-school-contact	Class 1	36	1773	0.25
	Class 2	34	1947	0.29
	Class 3	40	2987	0.20
	Class 4	29	913	0.41
	Class 5	38	2271	0.26
	Class 6	34	1320	0.26
	Class 7	44	2951	0.16
	Class 8	39	2204	0.19
	Class 9	33	1826	0.25

Table 5.3: Local clustering results for Amazon-reviews network

Metric	Method	Cluster								
		1	2	3	12	15	17	18	24	25
Cond.	U-HFD	**0.17**	**0.11**	**0.12**	**0.16**	**0.36**	**0.25**	**0.17**	**0.14**	**0.28**
	U-LH-2.0	0.42	0.50	0.25	0.44	0.74	0.44	0.57	0.58	0.61
	U-LH-1.4	0.33	0.44	0.25	0.36	0.81	0.40	0.51	0.54	0.59
	ACL	0.42	0.50	0.25	0.54	0.77	0.52	0.63	0.68	0.65
F1 score	U-HFD	**0.45**	**0.09**	**0.65**	**0.92**	0.04	**0.10**	**0.80**	**0.81**	**0.09**
	U-LH-2.0	0.23	0.07	0.23	0.29	**0.05**	0.06	0.21	0.28	0.05
	U-LH-1.4	0.23	**0.09**	0.35	0.40	0.00	0.07	0.31	0.35	0.06
	ACL	0.23	0.07	0.22	0.25	0.04	0.05	0.17	0.20	0.04

Table 5.4: Local clustering results for Microsoft-academic network

Metric	Method	Cluster			
		Data	ML	TCS	CV
Cond.	U-HFD	**0.03**	**0.06**	**0.06**	**0.03**
	U-LH-2.0	0.07	0.09	0.10	0.07
	U-LH-1.4	0.07	0.08	0.09	0.07
	ACL	0.08	0.11	0.11	0.09
F1 score	U-HFD	**0.78**	**0.54**	**0.86**	**0.73**
	U-LH-2.0	0.67	0.46	0.71	0.61
	U-LH-1.4	0.65	0.46	0.59	0.59
	ACL	0.64	0.43	0.70	0.57

Table 5.5: Local clustering results for Trivago-clicks network

Metric	Method	Cluster									
		KOR	ISL	PRI	UA-43	VNM	HKG	MLT	GTM	UKR	EST
Conductance	U-HFD	**0.010**	**0.023**	**0.014**	**0.011**	**0.018**	**0.017**	**0.010**	**0.007**	**0.016**	**0.012**
	U-LH-2.0	0.020	0.042	0.027	0.027	0.037	0.035	0.031	0.035	0.032	0.019
	U-LH-1.4	0.036	0.069	0.047	0.039	0.060	0.052	0.040	0.045	0.065	0.036
	ACL	0.027	0.050	0.034	0.031	0.042	0.043	0.047	0.039	0.043	0.026
	C-HFD	**0.007**	**0.016**	**0.007**	**0.005**	**0.009**	**0.011**	**0.007**	**0.003**	**0.010**	**0.009**
	C-LH-2.0	0.022	0.066	0.030	0.030	0.035	0.035	0.029	0.028	0.029	0.029
	C-LH-1.4	0.043	0.095	0.042	0.048	0.071	0.059	0.053	0.047	0.075	0.046
F1 score	U-HFD	0.75	**0.99**	0.89	0.85	0.28	0.82	**0.98**	0.94	0.60	**0.94**
	U-LH-2.0	0.70	0.86	0.79	0.70	0.24	0.92	0.88	0.82	0.50	0.90
	U-LH-1.4	0.69	0.84	0.80	0.75	0.28	0.87	0.92	0.83	0.47	0.90
	ACL	0.65	0.84	0.75	0.68	0.23	0.90	0.83	0.69	0.50	0.88
	C-HFD	**0.76**	**0.99**	**0.95**	**0.94**	**0.32**	0.80	**0.98**	**0.97**	**0.68**	**0.94**
	C-LH-2.0	0.73	0.90	0.84	0.78	0.27	**0.94**	0.96	0.88	0.51	0.83
	C-LH-1.4	0.71	0.88	0.84	0.78	0.27	0.88	0.93	0.85	0.50	0.85

Table 5.6: Local clustering results for High-school-contact network

Metric	Method	Cluster								
		Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
Conductance	U-HFD	**0.25**	**0.29**	**0.13**	**0.42**	**0.21**	**0.26**	**0.16**	**0.19**	**0.25**
	U-LH-2.0	0.31	0.36	0.23	0.63	0.33	0.36	0.18	0.21	0.30
	U-LH-1.4	0.29	0.32	0.21	0.54	0.29	0.37	**0.16**	0.22	0.29
	ACL	0.62	0.64	0.61	0.98	0.61	0.60	0.59	0.55	0.59
	C-HFD	**0.25**	**0.28**	**0.20**	**0.41**	**0.24**	**0.26**	**0.16**	**0.19**	**0.25**
	C-LH-2.0	0.27	0.33	**0.20**	0.57	0.29	0.32	**0.16**	0.20	0.27
	C-LH-1.4	0.28	0.32	**0.20**	0.52	0.28	0.33	**0.16**	0.21	0.28
F1 score	U-HFD	**0.99**	**1.00**	0.59	**0.96**	0.73	**1.00**	0.88	**1.00**	**0.99**
	U-LH-2.0	0.91	0.83	0.93	0.66	**0.84**	0.88	0.96	0.96	0.90
	U-LH-1.4	0.93	0.78	0.90	0.78	0.70	0.90	0.97	0.95	0.88
	ACL	0.72	0.73	0.73	0.06	0.70	0.76	0.77	0.78	0.76
	C-HFD	**0.99**	**1.00**	**1.00**	**0.96**	0.80	**1.00**	**1.00**	**1.00**	**0.99**
	C-LH-2.0	0.93	0.82	0.92	0.74	0.84	0.93	0.97	0.97	0.91
	C-LH-1.4	0.94	0.74	0.69	0.84	0.76	0.94	0.96	0.96	0.85

singe seed node. We report the median conductance and F1 score of the output cluster in Table 5.3 for Amazon-reviews networks, Table 5.4 for Microsoft-academic network, Table 5.6 for High-school-contact network, Table 5.5 for Trivago-clicks network. For Amazon-reviews, we only compare the unit cut-cost because it is both shown in [80] and verified by our experiments that unit cut-cost is more suitable for this dataset. Observe that U-HFD obtains the highest F1 scores for nearly all clusters. In particular, U-HFD significantly outperforms other methods for clusters 12, 18, 24, where we see an increase in F1 score by up to 52%. For Microsoft-academic, we stick with the unit cut-cost for the same reason, and for all clusters U-HFD yields the best result in terms of both conductance and F1. For Trivago-clicks, we consider both unit and cardinality cut-costs. C-HFD has the best performance for all but one clusters. Among the rest of all other methods, U-HFD has the second highest F1 scores for nearly all clusters. Moreover, observe that for each method (i.e., HFD, LH-2.0, LH-1.4), cardinality cut-cost leads to higher F1 than its unit cut-cost counterpart. For High-school-contact, C-HFD still gives the best overall performance.

Experiments for general submodular cut-cost. In order to understand the importance of specialized general submodular hypergraphs we study the node ranking problem for the Florida Bay food network using hypergraph modelling shown in Figure 1.2b. We compare HFD using unit (U-HFD, $\gamma_1 = \gamma_2 = 1$), cardinality-based (C-HFD, $\gamma_1 = 1/2$ and $\gamma_2 = 1$) and submodular (S-HFD, $\gamma_1 = 1/2$ and $\gamma_2 = 0$) cut-costs. Our goal is to search the most similar species of a queried species based on the food-network structure. Table 5.7 shows that S-HFD provides the only meaningful node ranking results. Intuitively, when $\gamma_2 = 0$, separating the preys v_1, v_2 from the predators v_3, v_4 incurs 0 cost. This encourages S-HFD to diffuse mass among preys or predators only and not to cross from a predator to a prey or vice versa. As a result, similar species receive similar amount of mass and thus are ranked similarly. In the local clustering setting, Table 5.8 compares HFD using different cut-costs. By exploiting specialized higher-order relations, S-HFD further improves F1 scores by up to 20% over U-HFD and C-HFD. This is not surprising given the poor node ranking results of other cut-costs.

Table 5.7: Node ranking results in Florida Bay food network using different cut-costs

	Top-2 node-ranking results	
Method	Query: Raptors	Query: Gray Snapper
U-HFD	Epiphytic Gastropods, Detriti. Gastropods	Meiofauna, Epiphytic Gastropods
C-HFD	Epiphytic Gastropods, Detriti. Gastropods	Meiofauna, Epiphytic Gastropods
S-HFD	Gruiformes, Small Shorebirds	Snook, Mackerel

Table 5.8: Local clustering results in Florida Bay food network using different cut-costs

	Cluster		
Method	Producers	Low-level consumers	High-level consumers
U-HFD	**0.69**	0.47	0.64
C-HFD	0.67	0.47	0.64
S-HFD	**0.69**	**0.62**	**0.84**

5.5 Proofs of Results

5.5.1 Technical Lemmas

We start with a few technical lemmas which will be used later in our proofs.

Lemma 5.13. *Let $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ be an optimal solution for Problem (5.3) and $\hat{x}$ an optimal solution for Problem (5.4). We have that*

$$\hat{\phi}_e^p = f_e(\hat{x})^q, \ \forall e \in E.$$

Moreover, if $\sigma > 0$, then

$$\hat{z}_v^p = \hat{x}_v^q, \ \forall v \in V.$$

Proof. It follows directly from Equation (5.13) and Equation (5.14) and strong duality that $\hat{\phi}, \hat{z}$ must satisfy, for each $e \in E$ and $v \in V$,

$$\hat{\phi}_e = f(\hat{x})^{q-1} = \operatorname*{argmax}_{\phi_e \geq 0} \ \phi_e f_e(\hat{x}) - \frac{1}{p}\phi_e^p \quad \text{and} \quad \hat{z}_v = \hat{x}_v^{q-1} = \operatorname*{argmax}_{z_v \geq 0} \ z_v \hat{x}_v - \frac{1}{p} z_v^p.$$

□

Lemma 5.14 (Proposition 4.2 in [15])**.** *Let w be a submodular function such that $w(\emptyset) = 0$. Let $x \in \mathbb{R}^{|V|}$, with unique values $a_1 > \cdots > a_m$, taken at sets $A_1, \ldots, A_m$ (i.e., $V = A_1 \cup \cdots \cup A_m$ and $\forall i \in \{1, \ldots, m\}$, $\forall v \in A_i$, $x_v = a_v$). Let B be the associated base polytope. Then $\rho \in B$ is optimal for $\max_{\rho \in B} \rho^T x$ if and only if for all $i = 1, \ldots, m$, $\rho(A_1 \cup \cdots \cup A_i) = w(A_1 \cup \cdots \cup A_i)$.*

Lemma 5.15. *For $x \in \mathbb{R}^{|V|}$ and $p > 1$ we have that*

$$\left(\sum_{v \in V} d_v |x_v| \right)^p \leq \mathrm{vol}(\mathrm{supp}(x))^{p-1} \sum_{v \in V} d_v |x_v|^p.$$

Proof. Let $q = p/(p-1)$. Apply the generalized Hölder's inequality we have

$$\sum_{v\in V} d_v|x_v| = \sum_{v\in \mathrm{supp}(x)} |d_v^{1/q}||d_v^{1/p}x_v| \le \left(\sum_{v\in\mathrm{supp}(x)} d_v\right)^{1/q}\left(\sum_{v\in\mathrm{supp}(x)} d_v|x_v|^p\right)^{1/p}$$
$$= \mathrm{vol}(\mathrm{supp}(x))^{1/q}\left(\sum_{v\in V} d_v|x_v|^p\right)^{1/p}.$$

□

Lemma 5.16 (Lemma I.2 in [78])**.** *For any $x \in \mathbb{R}_+^{|V|}\setminus\{0\}$ and $q \ge 1$, one has*

$$\frac{\sum_{e\in E}\vartheta_e f_e(x)^q}{\sum_{v\in V} d_v x_v^q} \ge \frac{c(x)^q}{q^q},$$

where

$$c(x) = \min_{h\ge 0}\frac{\mathrm{vol}(\partial(\{v\in V \mid x_v^q > h\}))}{\mathrm{vol}(\{v\in V\mid x_v^q>h\})} = \min_{h\ge 0}\frac{\mathrm{vol}(\partial(\{v\in V \mid x_v > h\}))}{\mathrm{vol}(\{v\in V\mid x_v>h\})}.$$

Lemma 5.17. *Let $(\hat\phi, \hat z, \{\hat r_e\}_{e\in E})$ be an optimal solution for Problem (5.3) and $\hat x$ an optimal solution for Problem (5.4). Let $S_h = \{v\in V \mid \hat x_v \ge h\}$. Let $\hat h = \mathrm{argmin}_{h>0}\Phi(S_h)$ and denote $\hat S = S_{\hat h}$. Then*

$$\sum_{e\in E}\vartheta_e\hat\phi_e^p \ge \left(\frac{\Phi(\hat S)}{q}\right)^q \sum_{v\in V} d_v\hat z_v^p.$$

Proof. By Lemma 5.13, we have

$$\sum_{e\in E}\vartheta_e\hat\phi_e^p = \sum_{e\in E}\vartheta_e f_e(\hat x)^q \quad\text{and}\quad \sum_{v\in V} d_v\hat z_v^p = \sum_{v\in V} d_v\hat x_v^q,$$

and the result follows from applying Lemma 5.16. □

Lemma 5.18. *Let $x\in\mathbb{R}_+^{|V|}$ and $S_h = \{v\in V\mid x_v\ge h\}$. We have that*

$$\Delta^T x = \int_{h=0}^{+\infty}\Delta(S_h)dh, \quad d^T x = \int_{h=0}^{+\infty}\mathrm{vol}(S_h)dh, \quad f_e(x) = \int_{h=0}^{+\infty} w_e(S_h)dh.$$

Proof. Recall that for a vector $a \in \mathbb{R}^{|V|}$ and a set $S \subseteq V$, we defined $a(S) = \sum_{v \in S} a_v$. This actually corresponds to a modular set-function $a : 2^V \to \mathbb{R}$ taking input on subsets of V. The Lovász extension of the modular function a is simply $f(x) = a^T x$ [15]. Based on the above definition, both Δ and d can be treated as modular functions on 2^V, and by definition the function w_e is a submodular function on 2^V. The Lovász extension of Δ and d are $\Delta^T x$ and $d^T x$, respectively. The Lovász extension of w_e is $f_e(x)$. The results then follow immediately from representing the Lovász extensions using Choquet integrals. See, e.g., Proposition 3.1 in [15]. □

5.5.2 Proof of Lemma 5.1

Using convex conjugates, for $x \in \mathbb{R}_+^{|V|}$, we have

$$\frac{1}{q} f_e(x)^q = \max_{\phi_e \geq 0} \ \phi_e f_e(x) - \frac{1}{p}\phi_e^p, \ \forall e \in E, \tag{5.13}$$

$$\frac{1}{q} x_v^q = \max_{z_v \geq 0} \ z_v x_v - \frac{1}{p} z_v^p, \ \forall v \in V. \tag{5.14}$$

Apply the definition of $f_e(x)$, we can write Equation (5.13) as

$$\frac{1}{q} f_e(x)^q = \max_{\phi_e \geq 0} \ \phi_e f_e(x) - \frac{1}{p}\phi_e^p \ = \max_{\phi_e \geq 0, r_e \in \phi_e B_e} r_e^T x - \frac{1}{p}\phi_e^p.$$

Therefore,

$$\begin{aligned}
&\max_{x \in \mathbb{R}_+^{|V|}} \ (\Delta - d)^T x - \frac{1}{q}\sum_{e \in E} \vartheta_e f_e(x)^q - \frac{\sigma}{q}\sum_{v \in V} d_v x_v^q \\
= &\max_{x \in \mathbb{R}_+^{|V|}} \ (\Delta - d)^T x - \sum_{e \in E} \vartheta_e \left(\max_{\phi_e \geq 0, r_e \in \phi_e B_e} \ r_e^T x - \frac{1}{p}\phi_e^p \right) - \sigma \sum_{v \in V} d_v \left(\max_{z_v \geq 0} \ z_v x_v - \frac{1}{p} z_v^p \right) \\
= &\max_{x \in \mathbb{R}_+^{|V|}} \ (\Delta - d)^T x + \min_{\substack{\phi \in \mathbb{R}_+^{|E|} \\ r_e \in \phi_e B_e, \forall e \in E}} \ \sum_{e \in E} \left(\frac{1}{p}\vartheta_e \phi_e^p - \vartheta_e r_e^T x \right) + \min_{z \in \mathbb{R}_+^{|V|}} \ \sigma \sum_{v \in V} \left(\frac{1}{p} d_v z_v^p - d_v z_v x_v \right) \\
= &\min_{\substack{\phi \in \mathbb{R}_+^{|E|}, z \in \mathbb{R}_+^{|V|} \\ r_e \in \phi_e B_e, \forall e \in E}} \ \frac{1}{p}\sum_{e \in E} \vartheta_e \phi_e^p + \frac{\sigma}{p}\sum_{v \in V} d_v z_v^p + \max_{x \in \mathbb{R}_+^{|V|}} \left((\Delta - d)^T x - \sum_{e \in E} \vartheta_e r_e^T x - \sigma \sum_{v \in V} d_v z_v x_v \right) \\
= &\min_{\substack{\phi \in \mathbb{R}_+^{|E|}, z \in \mathbb{R}_+^{|V|} \\ r_e \in \phi_e B_e, \forall e \in E}} \ \frac{1}{p}\sum_{e \in E} \vartheta_e \phi_e^p + \frac{\sigma}{p}\sum_{v \in V} d_v z_v^p \quad \text{s.t.} \quad \Delta - d - \sum_{e \in E} \vartheta_e r_e - \sigma D z \leq 0.
\end{aligned}$$

In the above derivations, we may exchange the order of minimization and maximization and arrive at the second last equality, due to Proposition 2.2, Chapter VI, in [41]. The last equality follows from

$$\max_{x\in\mathbb{R}_+^{|V|}}\left((\Delta-d)^Tx-\sum_{e\in E}\vartheta_e r_e^Tx-\sigma\sum_{v\in V}d_vz_vx_v\right)=\begin{cases}0, & \text{if } \Delta-d-\sum\limits_{e\in E}\vartheta_e r_e-\sigma Dz\le 0,\\ +\infty, & \text{otherwise.}\end{cases}$$

5.5.3 Proof of Lemma 5.2

To see the first inequality, note that if $\hat{x}_v=0$ for every $v\in e$ for some e, then $f_e(\hat{x})=0$. By Lemma 5.13, this means $\hat{\phi}_e=0$. Thus, $\hat{\phi}_e\neq 0$ only if there is some $v\in e$ such that $\hat{x}_v\neq 0$. Therefore, we have that

$$\sum_{e\in\mathrm{supp}(\hat{\phi})}\vartheta_e\le\sum_{v\in\mathrm{supp}(\hat{x})}\sum_{e\in E:v\in e}\vartheta_e=\sum_{v\in\mathrm{supp}(\hat{x})}d_v=\mathrm{vol}(\mathrm{supp}(\hat{x})).$$

To see the last inequality, note that, by the first order optimality condition of Problem (5.4), if $\hat{x}_v\neq 0$ then we must have

$$\Delta_v-d_v=\sum_{e\in E}\vartheta_e f_e(\hat{x})^{q-1}\hat{\rho}_{e,v}+\sigma d_v\hat{x}_v^{q-1},\quad\text{for some}\quad\hat{\rho}_e\in\partial f_e(\hat{x})=\operatorname*{argmax}_{\rho_e\in B_e}\rho_e^T\hat{x}.\tag{5.15}$$

Denote $N=\mathrm{supp}(\hat{x})$ and $E[N]=\{e\in E\mid v\in N\text{ for all }v\in e\}$. Note that $E[N]\cap\partial(N)=\emptyset$, and $E[N]\cup\partial(N)=\{e\in E\mid v\in N\text{ for some }v\in e\}$, that is, $E[N]\cup\partial(N)$ contain all hyperedges that are incident to some node in N. Moreover, we have that for any $\hat{\rho}_e\in\mathrm{argmax}_{\rho_e\in B_e}\rho_e^T\hat{x}$,

$$\sum_{v\in N}\hat{\rho}_{e,v}=\hat{\rho}_e(N)=\begin{cases}w_e(N), & \text{if } e\in\partial(N),\\ 0, & \text{if } e\in E[N],\end{cases}$$

where $\hat{\rho}_e(N)=w_e(N)$ for $e\in\partial(N)$ follows from Lemma 5.14, since $\hat{x}_v>0$ for $v\in N$ and $\hat{x}_v=0$ for $v\notin N$. The equality $\hat{\rho}_e(N)=0$ for $e\in E[N]$ follows from $\hat{\rho}_e(N)=\hat{\rho}_e(e)=0$ because $e\subseteq N$ and $\hat{\rho}_{e,v}=0$ for all $v\notin e$.

Taking sums over $v \in N$ on both sides of equation (5.15) we obtain

$$\begin{aligned}
\Delta(N) - \mathrm{vol}(N) &= \sum_{v\in N}\sum_{e\in E} \vartheta_e f_e(\hat{x})^{q-1}\hat{\rho}_{e,v} + \sum_{v\in N}\sigma d_v \hat{x}_v^{q-1} \\
&= \sum_{v\in N}\sum_{e\in E[N]} \vartheta_e f_e(\hat{x})^{q-1}\hat{\rho}_{e,v} + \sum_{v\in N}\sum_{e\in\partial(N)} \vartheta_e f_e(\hat{x})^{q-1}\hat{\rho}_{e,v} + \sum_{v\in N}\sigma d_v \hat{x}_v^{q-1} \\
&= \sum_{e\in E[N]} \vartheta_e f_e(\hat{x})^{q-1}\sum_{v\in N}\hat{\rho}_{e,v} + \sum_{e\in\partial(N)} \vartheta_e f_e(\hat{x})^{q-1}\sum_{v\in N}\hat{\rho}_{e,v} + \sum_{v\in N}\sigma d_v \hat{x}_v^{q-1} \\
&= 0 + \sum_{e\in\partial(N)} \vartheta_e f_e(\hat{x})^{q-1} w_e(N) + \sum_{v\in N}\sigma d_v \hat{x}_v^{q-1} \\
&\geq 0.
\end{aligned}$$

The second equality follows from $\hat{\rho}_{e,v} = 0$ for all $v \notin e$. This proves $\mathrm{vol}(\mathrm{supp}(\hat{x})) \leq \Delta(\mathrm{supp}(\hat{x})) \leq \|\Delta\|_1$.

Finally, if $\sigma > 0$, then $\mathrm{vol}(\mathrm{supp}(\hat{z})) = \mathrm{vol}(\mathrm{supp}(\hat{x}))$ follows from Lemma 5.13 that $\hat{z}^p = \hat{x}^q$ for all $v \in V$.

5.5.4 Proof of Lemma 5.7

Denote $\hat{\nu} = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p$. We need to show that $\hat{\nu} \geq \mathrm{vol}(C)^p / \mathrm{vol}(\partial(C))^{p-1}$. The proof of this follows from a case analysis on the total amount of excess mass $\sigma \sum_{v\in V} d_v \hat{z}_v$ at optimality. Intuitively, if the excess is small, then naturally there must be a large amount of flow in order to satisfy the primal constraint; if the excess is large, then Lemma 5.17 and Lemma 5.15 guarantee that the flow is also large. We give details below.

Suppose that $\sigma \sum_{v\in V} d_v \hat{z}_v < \mathrm{vol}(C)$. Note that this also includes the case where $\sigma = 0$. By Assumption 5.4 there is at least $\Delta(C) \geq 3\mathrm{vol}(C)$ amount of source mass trapped in C at the beginning. Moreover, the primal constraint enforces the nodes in C can settle at most $\sum_{v\in C}(d_v + \sigma d_v \hat{z}_v) \leq \mathrm{vol}(C) + \sum_{v\in V} \sigma d_v \hat{z}_v < 2\mathrm{vol}(C)$ amount of mass. Therefore, the remaining at least $\mathrm{vol}(C)$ amount of mass needs to get out of C using the hyperedges in $\partial(C)$. That is, the net amount of mass that moves from C to $V \setminus C$ satisfies $\sum_{e\in\partial(C)} \vartheta_e \hat{r}_e(C) \geq \mathrm{vol}(C)$. We focus on the cost of $\hat{\phi}$ restricted to these hyperedges alone.

It is easy to see that

$$\sum_{e\in\partial C}\vartheta_e\hat{\phi}_e^p \;\geq\; \min_{\phi\in\mathbb{R}_+^{|\partial C|}} \sum_{e\in\partial C}\vartheta_e\phi_e^p \;\text{ subject to }\; \hat{r}_e\in\phi_e B_e,\;\forall e\in\partial C \tag{5.16a}$$

$$\geq \min_{\phi\in\mathbb{R}_+^{|\partial C|}} \sum_{e\in\partial C}\vartheta_e\phi_e^p \;\text{ subject to }\; \sum_{e\in\partial C}\vartheta_e\hat{r}_e(C)\leq\sum_{e\in\partial C}\vartheta_e\phi_e w_e(C) \tag{5.16b}$$

$$\geq \min_{\phi\in\mathbb{R}_+^{|\partial C|}} \sum_{e\in\partial C}\vartheta_e\phi_e^p \;\text{ subject to }\; \mathrm{vol}(C)\leq\sum_{e\in\partial C}\vartheta_e\phi_e w_e(C). \tag{5.16c}$$

The first inequality follows because $\hat{\phi}$ restricted to $\partial(C)$ is a feasible solution in Problem (5.16a). The second inequality follows because $\hat{r}_e\in\phi_e B_e$ implies $\hat{r}_e(C)\leq\phi_e w_e(C)$, therefore every feasible solution for (5.16a) is also a feasible solution for (5.16b). The third inequality follows because $\mathrm{vol}(C)\leq\sum_{e\in E}\vartheta_e\hat{r}_e(C)$. Let $\bar{\phi}\in\mathbb{R}_+^{|\partial(C)|}$ be an optimal solution of Problem (5.16c). The optimality condition of (5.16c) is given by (we may assume the p factor in the gradient of $\sum_{e\in\partial(C)}\vartheta_e\phi_e^p$ is absorbed into multipliers λ and η_e)

$$\begin{aligned}
&\vartheta_e\phi_e^{p-1}-\lambda\vartheta_e w_e(C)-\eta_e=0,\;\forall e\in\partial C\\
&\phi_e\geq 0,\;\eta_e\geq 0,\;\phi_e\eta_e=0,\;\forall e\in\partial C\\
&\mathrm{vol}(C)\leq\sum_{e\in\partial C}\vartheta_e\phi_e w_e(C)\\
&\lambda\geq 0,\;\lambda\left(\mathrm{vol}(C)-\sum_{e\in\partial(C)}\vartheta_e\phi_e w_e(C)\right)=0.
\end{aligned}\tag{5.17}$$

If $\lambda=0$, then the conditions in (5.17) imply that $\vartheta_e\phi_e^{p-1}=\eta_e$, but then by complimentary slackness we would obtain $\phi_e=\eta_e=0$ for all $e\in\partial(C)$ which will violate feasibility. Therefore we must have $\lambda>0$, and consequently, we have that

$$\sum_{e\in\partial(C)}\vartheta_e\bar{\phi}_e w_e(C)=\mathrm{vol}(C). \tag{5.18}$$

Moreover, the conditions in (5.17) imply that for $e\in\partial(C)$, $\bar{\phi}_e=0$ if and only if $w_e(C)=0$, and hence we have that

$$\vartheta_e\bar{\phi}_e^{p-1}=\lambda\vartheta_e w_e(C),\;\forall e\in\partial(C). \tag{5.19}$$

Rearrange (5.19) we get

$$\bar{\phi}_e w_e(C)=\lambda^{1/(p-1)}w_e(C)^{p/(p-1)},\;\forall e\in\partial(C).$$

Substitute the above into (5.18),

$$\mathrm{vol}(C) = \sum_{e\in\partial(C)} \vartheta_e \bar{\phi}_e w_e(C) = \sum_{e\in\partial(C)} \vartheta_e \lambda^{1/(p-1)} w_e(C)^{p/(p-1)},$$

this gives

$$\lambda^{1/(p-1)} = \frac{\mathrm{vol}(C)}{\sum_{e\in\partial(C)} \vartheta_e w_e(C)^{p/(p-1)}}.$$

Therefore, the solution $\bar{\phi}$ for (5.16c) is give by

$$\bar{\phi}_e = \lambda^{1/(p-1)} w_e(C)^{1/(p-1)} = \frac{\mathrm{vol}(C) w_e(C)^{1/(p-1)}}{\sum_{e'\in\partial(C)} \vartheta_{e'} w_{e'}(C)^{p/(p-1)}}, \quad \forall e \in \partial(C),$$

and hence,

$$\begin{aligned}
\hat{\nu} &= \sum_{e\in E} \vartheta_e \hat{\phi}_e^p \geq \sum_{e\in\partial(C)} \vartheta_e \hat{\phi}_e^p \geq \sum_{e\in\partial(C)} \vartheta_e \bar{\phi}_e^p \\
&= \sum_{e\in\partial(C)} \vartheta_e \frac{\mathrm{vol}(C)^p w_e(C)^{p/(p-1)}}{\left(\sum_{e'\in\partial(C)} \vartheta_{e'} w_{e'}(C)^{p/(p-1)}\right)^p} = \frac{\mathrm{vol}(C)^p \sum_{e\in\partial(C)} \vartheta_e w_e(C)^{p/(p-1)}}{\left(\sum_{e'\in\partial(C)} \vartheta_{e'} w_{e'}(C)^{p/(p-1)}\right)^p} \\
&= \frac{\mathrm{vol}(C)^p}{\left(\sum_{e'\in\partial(C)} \vartheta_{e'} w_{e'}(C)^{p/(p-1)}\right)^{p-1}} \geq \frac{\mathrm{vol}(C)^p}{\left(\sum_{e'\in\partial(C)} \vartheta_{e'} w_{e'}(C)\right)^{p-1}},
\end{aligned}$$

where the last inequality follows because $w_e(C) \in [0,1]$ and $p \geq 1$.

Suppose now that $\sigma \sum_{v\in V} d_v \hat{z}_v \geq \mathrm{vol}(C)$. Because $\Phi(C) \leq (\Phi(\hat{S})/q)^q$ (recall that we assumed this without loss of generality), by Assumption 5.5, we know that $\sigma < (\phi(\hat{S})/q)^q$. Therefore,

$$\begin{aligned}
\hat{\nu} = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p &\overset{(i)}{\geq} \sigma \sum_{v\in V} d_v \hat{z}_v^p \\
&\overset{(ii)}{\geq} \frac{\sigma \left(\sum_{v\in V} d_v \hat{z}_v\right)^p}{\mathrm{vol}(\mathrm{supp}(\hat{z}))^{p-1}} \\
&\overset{(iii)}{\geq} \frac{\sigma^p \left(\sum_{v\in V} d_v \hat{z}_v\right)^p}{\sigma^{p-1}(3\mathrm{vol}(C)/\beta)^{p-1}} \\
&\overset{(iv)}{\geq} \frac{\sigma^p \left(\sum_{v\in V} d_v \hat{z}_v\right)^p}{\mathrm{vol}(\partial(C))^{p-1}} \\
&\overset{(v)}{\geq} \frac{\mathrm{vol}(C)^p}{\mathrm{vol}(\partial(C))^{p-1}}.
\end{aligned}$$

In the above, (i) is due to Lemma 5.16, (ii) is due to Lemma 5.15, (iii) is due to Lemma 5.2 that $\mathrm{vol}(\mathrm{supp}(\hat{z})) \le \|\Delta\|_1$ and Assumption 5.4 that $\|\Delta\|_1 \le 3\mathrm{vol}(C)/\beta$, so for $p \ge 1$,

$$\mathrm{vol}(\mathrm{supp}(\hat{z}))^{p-1} \le (3\mathrm{vol}(C)/\beta)^{p-1}.$$

(iv) is due to Assumption 5.5 that $\sigma \le \frac{\beta \mathrm{vol}(\partial(C))}{3\mathrm{vol}(C)}$, so for $p \ge 1$,

$$(3\sigma\mathrm{vol}(C)/\beta)^{p-1} \le \mathrm{vol}(\partial(C))^{p-1}.$$

(v) is due to the assumption that $\sigma \sum_{v\in V} d_v \hat{z}_v \ge \mathrm{vol}(C)$. This proves Lemma 5.7.

5.5.5 Proof of Claim 5.8

Denote $\hat{\nu} = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p$. For $e \in E$, define $l(e) = f_e(\hat{x})/\hat{\nu}^{1/q}$. Then $l(e) \le \hat{l}(e)$. Moreover,

$$\sum_{e:l(e)<\hat{l}(e)} \vartheta_e \le \sum_{e\in\mathrm{supp}(\hat{\phi})} \vartheta_e \le \mathrm{vol}(\mathrm{supp}(\hat{x})) \le \|\Delta\|_1 = \frac{3}{\alpha}\mathrm{vol}(S) \le \frac{3}{\beta}\mathrm{vol}(C).$$

The first inequality follows from that $l(e) < \hat{l}(e)$ only if $l(e) \ne 0$, and by Lemma 5.13, $l(e) \ne 0$ if and only if $\hat{\phi}_e \ne 0$. The second and the third inequalities are due to Lemma 5.2. The second to last equality follows from the diffusion setup (5.5) and Assumption 5.4 that $\delta = 3/\alpha$. The last inequality follows from Assumption 5.3. Therefore,

$$\begin{aligned}
\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1} &= \sum_{e:l(e)=\hat{l}(e)} \vartheta_e f_e(\hat{x}) \frac{f_e(\hat{x})^{q-1}}{\hat{\nu}^{(q-1)/q}} + \sum_{e:l(e)<\hat{l}(e)} \vartheta_e f_e(\hat{x}) \frac{1}{\mathrm{vol}(C)^{(q-1)/q}} \\
&\le \sum_{e:l(e)=\hat{l}(e)} \vartheta_e f_e(\hat{x}) \frac{f_e(\hat{x})^{q-1}}{\hat{\nu}^{(q-1)/q}} + \sum_{e:l(e)<\hat{l}(e)} \vartheta_e \frac{\hat{\nu}^{1/q}}{\mathrm{vol}(C)^{1/q}} \frac{1}{\mathrm{vol}(C)^{(q-1)/q}} \\
&= \frac{1}{\hat{\nu}^{(q-1)/q}} \sum_{e:l(e)=\hat{l}(e)} \vartheta_e f_e(\hat{x})^q + \frac{\hat{\nu}^{1/q}}{\mathrm{vol}(C)} \sum_{e:l(e)<\hat{l}(e)} \vartheta_e \\
&\le \frac{1}{\hat{\nu}^{(q-1)/q}} \sum_{e\in E} \vartheta_e f_e(\hat{x})^q + \frac{\hat{\nu}^{1/q}}{\mathrm{vol}(C)} \frac{3\mathrm{vol}(C)}{\beta} \\
&= \frac{\hat{\nu}}{\hat{\nu}^{(q-1)/q}} + \frac{3\hat{\nu}^{1/q}}{\beta} \\
&\le \frac{4\hat{\nu}^{1/q}}{\beta}
\end{aligned}$$

where the last equality follows from Lemma 5.13 that $\hat{\nu} = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p = \sum_{e\in E} \vartheta_e f_e(\hat{x})^q$.

5.5.6 Proof of Theorem 5.6

Let $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ be an optimal solution for Problem (5.3) and $\hat{x}$ an optimal solution for Problem (5.4). For $h > 0$ define the level cuts

$$S_h = \{v \in V \mid \hat{x}_v \geq h\}.$$

Denote $\hat{\nu} = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p$. By the strong duality between Problem (5.3) and Problem (5.4), we know that

$$(\Delta - d)^T \hat{x} - \frac{1}{q}\sum_{e\in E} \vartheta_e f_e(\hat{x})^q - \frac{\sigma}{q}\sum_{v\in V} d_v \hat{x}_v^q = \frac{1}{p}\sum_{e\in E} \vartheta_e \hat{\phi}_e^p + \frac{\sigma}{p}\sum_{v\in V} d_v \hat{z}_v^p.$$

Hence, by Lemma 5.13, we get

$$(\Delta - d)^T \hat{x} \geq \frac{1}{q}\sum_{e\in E} \vartheta_e f_e(\hat{x})^q + \frac{1}{p}\sum_{e\in E} \vartheta_e \hat{\phi}_e^p = \sum_{e\in E} \vartheta_e \hat{\phi}_e^p = \hat{\nu}.$$

It then follows that

$$\frac{\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1}}{(\Delta - d)^T \hat{x}} \leq \frac{\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1}}{\hat{\nu}} \overset{(i)}{\leq} \frac{4\hat{\nu}^{1/q}}{\beta\hat{\nu}} = \frac{4}{\beta\hat{\nu}^{1/p}} \overset{(ii)}{\leq} \frac{4\mathrm{vol}(\partial C)^{1/q}}{\beta \mathrm{vol}(C)}, \tag{5.20}$$

where (i) is follows from Claim 5.8 and (ii) follows from Lemma 5.7.

We can write the left hand side of (5.20) in its integral form, as follows. By Lemma 5.18, we have

$$(\Delta - d)^T \hat{x} = \int_{h=0}^{\infty} (\Delta(S_h) - \mathrm{vol}(S_h))dh,$$

and

$$\begin{aligned}
\sum_{e\in E} \vartheta_e f_e(\hat{x})\hat{l}(e)^{q-1} &= \sum_{e\in E} \vartheta_e \int_{h=0}^{\infty} w_e(S_h)dh\ \hat{l}(e)^{q-1} \\
&= \int_{h=0}^{\infty} \sum_{e\in E} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}dh \\
&= \int_{h=0}^{\infty} \sum_{e\in \partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}dh,
\end{aligned}$$

where the last equality follows from the fact that $w_e(S_h) = 0$ for $e \notin \partial(S_h)$. Therefore, we get

$$\int_{h=0}^{\infty} \frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)}dh \leq \frac{4\mathrm{vol}(\partial(C))^{1/q}}{\beta \mathrm{vol}(C)},$$

which means that there must exist $h > 0$ with S_h non-empty, such that

$$\frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1}}{\Delta(S_h) - \mathrm{vol}(S_h)} \leq \frac{4\mathrm{vol}(\partial(C))^{1/q}}{\beta \mathrm{vol}(C)}. \tag{5.21}$$

We connect the left hand side of (5.21) to the conductance of S_h. For the denominator, by Assumption 5.4, we have

$$\Delta(S_h) - \mathrm{vol}(S_h) \leq \frac{3}{\alpha}\mathrm{vol}(S_h). \tag{5.22}$$

For the numerator, every hyperedge $e \in \partial(S_h)$ must contain some $u, v \in e$ such that $\hat{x}_u \neq \hat{x}_v$, thus $f_e(\hat{x}) > 0$, which means $\hat{l}(e) \geq 1/\mathrm{vol}(C)^{1/q}$. This gives

$$\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)\hat{l}(e)^{q-1} \geq \frac{\sum_{e\in\partial(S_h)} \vartheta_e w_e(S_h)}{\mathrm{vol}(C)^{(q-1)/q}} = \frac{\mathrm{vol}(\partial(S_h))}{\mathrm{vol}(C)^{(q-1)/q}}. \tag{5.23}$$

Put (5.21), (5.22) and (5.23) together, we have

$$\Phi(S_h) = \frac{\mathrm{vol}(\partial(S_h))}{\mathrm{vol}(S_h)} \leq \frac{12\mathrm{vol}(\partial(C))^{1/q}}{\alpha\beta\mathrm{vol}(C)^{1/q}} = \frac{12\Phi(C)^{1/q}}{\alpha\beta}.$$

5.5.7 Proof of Lemma 5.9 and Corollary 5.10

We will show the forward direction and the converse follows from the same reasoning. Let $\hat{\nu}_1$ and $\hat{\nu}_2$ denote the optimal objective value of Problem (5.3) and Problem (5.8), respectively. Let $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ be an optimal solution for Problem (5.3). Define

$$\hat{s}_e := \hat{r}_e + \sigma A_e \hat{z},\ \forall e \in E,$$

where $A_e \in \mathbb{R}^{|V|\times|V|}$ is a diagonal matrix such that $[A_e]_{v,v} = 1$ if $v \in e$ and 0 otherwise. We show that $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ is an optimal solution for Problem (5.8).

Because $\hat{r}_{e,v} = 0$ for all $v \notin e$, by the definition of A_e, we know that $\hat{s}_{e,v} = 0$ for all $v \notin e$. Moreover,

$$\sigma D\hat{z} = \sigma \sum_{e\in E} \vartheta_e A_e \hat{z} = \sum_{e\in E} \vartheta_e(\hat{s}_e - \hat{r}_e),$$

so

$$\Delta - \sum_{e\in E} \vartheta_e \hat{s}_e = \Delta - \sum_{e\in E} \vartheta_e \hat{r}_e - \sigma D\hat{z} \leq d.$$

Therefore, $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ is a feasible solution for Problem (5.8). Furthermore,

$$\begin{aligned}\sigma \sum_{v\in V} d_v \hat{z}_v^p &= \sigma \sum_{e\in E} \vartheta_e \sum_{v\in e} \hat{z}_v^p = \sigma \sum_{e\in E} \vartheta_e \|A_e \hat{z}\|_p^p \\ &= \frac{1}{\sigma^{p-1}} \sum_{e\in E} \vartheta_e \left\|\sigma A_e \hat{z}\right\|_p^p = \frac{1}{\sigma^{p-1}} \sum_{e\in E} \vartheta_e \left\|\hat{s}_e - \hat{r}_e\right\|_p^p.\end{aligned}$$

This means that $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ attains objective value $\hat{\nu}_1$ in Problem (5.8). Hence $\hat{\nu}_1 \geq \hat{\nu}_2$.

In order to show that $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ is indeed optimal for Problem (5.8), it left to show that $\hat{\nu}_2 \geq \hat{\nu}_1$. Let $(\phi', \{r'_e\}_{e\in E}, \{s'_e\}_{e\in E})$ be an optimal solution for Problem (5.8). Then we know that

$$s' = \operatorname*{argmin}_{s\in \bigotimes_{e\in E} \mathbb{R}^{|V|}} \sum_{e\in E} \vartheta_e \|s_e - r'_e\|_p^p, \text{ s.t. } \Delta - \sum_{e\in E} \vartheta_e s_e \leq d, \ s_{e,v} = 0 \ \forall v \notin e. \tag{5.24}$$

According to Lemma 5.12, we know that

$$s'_e = r'_e + A_e D^{-1} \left[\Delta - \sum_{e'\in E} \vartheta_{e'} r'_{e'} - d\right]_+, \ \forall e \in E. \tag{5.25}$$

Define

$$z' := \frac{1}{\sigma} D^{-1} \left[\Delta - \sum_{e\in E} \vartheta_e r'_e - d\right]_+.$$

Then $z' \geq 0$. Moreover, we have that

$$\sum_{e\in E} \vartheta_e s'_e - \sum_{e\in E} \vartheta_e r'_e = \sum_{e\in E} \vartheta_e A_e D^{-1} \left[\Delta - \sum_{e'\in E} \vartheta_{e'} r'_{e'} - d\right]_+ = \left[\Delta - \sum_{e'\in E} \vartheta_{e'} r'_{e'} - d\right]_+ = \sigma D z',$$

so

$$\Delta - \sum_{e\in E} \vartheta_e r'_e = \Delta - \sum_{e\in E} \vartheta_e s'_e + \sigma D z' \leq d + \sigma D z'.$$

Therefore, $(\phi', z', \{r'_e\}_{e\in E})$ is a feasible solution for Problem (5.3). Furthermore,

$$\begin{aligned}\frac{1}{\sigma^{p-1}} \sum_{e\in E} \vartheta_e \left\|s'_e - r'_e\right\|_p^p &= \frac{1}{\sigma^{p-1}} \sum_{e\in E} \vartheta_e \left\|\sigma A_e z'\right\|_p^p = \sigma \sum_{e\in E} \vartheta_e \|A_e z'\|_p^p \\ &= \sigma \sum_{e\in E} \vartheta_e \sum_{v\in e} z'^p_v = \sigma \sum_{v\in V} d_v z'^p_v.\end{aligned}$$

This means that $(\phi', z', \{r'_e\}_{e\in E})$ attains objective value $\hat{\nu}_2$ in Problem (5.3). Hence $\hat{\nu}_2 \geq \hat{\nu}_1$. This finishes the proof of Lemma 5.9.

Given an optimal solution $(\hat{\phi}, \{\hat{r}_e\}_{e\in E}, \{\hat{s}_e\}_{e\in E})$ for Problem (5.8), the above constructive proof shows that one can recover an optimal solution $(\hat{\phi}, \hat{z}, \{\hat{r}_e\}_{e\in E})$ for Problem (5.3) via $\hat{z} = \frac{1}{\sigma} D^{-1} \left[\Delta - \sum_{e\in E} \vartheta_e \hat{r}_e - d\right]_+$. It then follows from Lemma 5.13 that an optimal dual solution $\hat{x}$ is given by $\hat{x}_v = \hat{z}_v^{p-1}$ for all $v \in V$. This gives Corollary 5.10.

5.5.8 Proof of Theorem 5.11

In order to apply Theorem 3.2 in [20] we need to show that the objective function $g(\phi, r, s)$ of Problem (5.8) is block-wise Lipschitz smooth in the sub-level sets containing the iterates generated by Algorithm 4. Recall that $p \geq 2$.

Lemma 5.19 (Block-wise Lipschitz smoothness)**.** *The partial gradient $\nabla_{(\phi,r)} g(\phi, r, s)$ is Lipschitz continuous over the sub-level sets (given any fixed s)*

$$U_{\phi,r}(s) := \{(\phi, r) \in \mathbb{R}_+^{|V|} \times (\bigotimes_{e\in E} \mathbb{R}^{|V|}) \mid g(\phi, r, s) \leq g(\phi^{(0)}, r^{(0)}, s^{(0)})\}$$

with constant $L_{\phi,r}$ such that

$$L_{\phi,r} \leq (p-1) \frac{\vartheta_{\max}^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{p-1}},$$

where $\vartheta_{\max} := \max_{e\in E} \vartheta_e$ and $d_{\min} := \min_{v\in \mathrm{supp}(\Delta)} d_v$. The partial gradient $\nabla_s g(\phi, r, s)$ is Lipschitz continuous over the sub-level sets (given any fixed (ϕ, r))

$$U_s(\phi, r) := \{s \in \bigotimes_{e\in E} \mathbb{R}^{|V|} \mid g(\phi, r, s) \leq g(\phi^{(0)}, r^{(0)}, s^{(0)})\}$$

with constant $L_s \leq L_{\phi,r}$.

Proof. Fix $s \in \bigotimes_{e\in E} \mathbb{R}^{|V|}$ and consider

$$g_1(\phi, r) := g(\phi, r, s) = \frac{1}{p} \sum_{e\in E} \vartheta_e \phi_e^p + \frac{1}{p\sigma^{p-1}} \sum_{e\in E} \sum_{v\in V} \vartheta_e |r_{e,v} - s_{e,v}|^p.$$

The function $g_1(\phi, r)$ is coordinate-wise separable and hence its second order derivative $\nabla^2 g_1(\phi, r)$ is a diagonal matrix. Therefore, the largest eigenvalue of $\nabla^2 g_1(\phi, r)$ is the largest coordinate-wise second order partial derivative, that is,

$$L_{\phi,r} = \max_{(\phi,r)\in U_{\phi,r}(s)} \lambda_{\max}(\nabla^2 g_1(\phi, r)) = \max_{(\phi,r)\in U_{\phi,r}(s)} \max_{e\in E, v\in V} \{\nabla^2_{\phi_e} g_1(\phi, r), \nabla^2_{r_{e,v}} g_1(\phi, r)\}.$$

So it suffices to upper bound $\nabla^2_{\phi_e} G(\phi, r)$ and $\nabla^2_{r_{e,v}} G(\phi, r)$ for all $(\phi, r) \in U_{\phi,r}(s)$. We have that

$$g(\phi^{(0)}, r^{(0)}, s^{(0)}) = \frac{1}{p\sigma^{p-1}} \sum_{e\in E} \vartheta_e \sum_{v\in e} \frac{[\Delta_v - d_v]_+^p}{d_v^p} = \frac{1}{p\sigma^{p-1}} \sum_{v\in V} \frac{[\Delta_v - d_v]_+^p}{d_v^{p-1}} \le \frac{\|\Delta\|_p^p}{p\sigma^{p-1} d_{\min}^{p-1}}$$

where $d_{\min} = \min_{v\in \mathrm{supp}(\Delta)} d_v$. It follows that for all $(\phi, r) \in U_{\phi,r}(s)$,

$$\nabla^2_{\phi_e} g_1(\phi, r) = (p-1)\vartheta_e \phi_e^{p-2} \le \frac{(p-1)\vartheta_e^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{(p-1)(p-2)/p}} \le \frac{(p-1)\vartheta_e^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{p-1}}, \ \forall e \in E,$$

$$\nabla^2_{r_{e,v}} g_1(\phi, r) = (p-1)\frac{\vartheta_e}{\sigma^{p-1}} |s_{e,v} - r_{e,v}|^{p-2} \le \frac{(p-1)\vartheta_e^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{p-1}}, \ \forall e \in E, \ \forall v \in V,$$

because otherwise we would have $g(\phi, r, s) > g(\phi^{(0)}, r^{(0)}, s^{(0)})$. Hence,

$$L_{\phi,r} \le \max_{e\in E} \frac{(p-1)\vartheta_e^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{p-1}} = \frac{(p-1)\vartheta_{\max}^{2/p} \|\Delta\|_p^{p-2}}{d_{\min}^{(p-1)(p-2)/p} \sigma^{p-1}}.$$

Finally, by the symmetry between r and s in $F(\phi, r, s)$, we know that $L_s \le L_{\phi,r}$. □

Because the iterates generated by Algorithm 4 monotonically decrease the objective function value, in particular, we have that

$$g(\phi^{(0)}, r^{(0)}, s^{(0)}) \ge g(\phi^{(k+1)}, r^{(k+1)}, s^{(k)}) \ge g(\phi^{(k+1)}, r^{(k+1)}, s^{(k+1)})$$

for any $k \ge 0$. Therefore, the sequence of iterates live in the sub-level sets. As a result, for any $p \ge 2$, Lemma 5.19 gives the required block-wise Lipschitz smoothness within sub-level sets, and consequently, allows us to apply Theorem 3.2 in [20] to Algorithm 4 and obtain the rate of convergence as stated in Theorem 5.11.

5.5.9 Proof of Lemma 5.12

Rewrite Problem (5.9) as

$$\begin{aligned} \min_{s\in \bigotimes_{e\in E} \mathbb{R}^{|V|}} & \sum_{v\in V} \sum_{e\in E} \vartheta_e |s_{e,v} - r_{e,v}|^p \\ \text{s.t.} \quad & \Delta_v - \sum_{e\in E} \vartheta_e s_{e,v} \le d_v, \ \forall v \in V \\ & s_{e,v} = 0, \ \forall v \notin e. \end{aligned}$$

Then it is clear to see that Problem (5.9) decomposes into $|V|$ sub-problems indexed by $v \in V$,

$$\min_{\xi_v \in \mathbb{R}^{|E_v|}} \sum_{e \in E_v} \vartheta_e |\xi_{v,e} - r_{e,v}|^p, \text{ s.t. } \Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} \leq d_v, \tag{5.26}$$

where $E_v := \{e \in E \mid v \in e\}$ is the set of hyperedges incident to v, and we use $\xi_{v,e}$ for the entry in ξ_v that corresponds to $e \in E_v$. Let ξ_v^* denote the optimal solution for Problem (5.26). We have that $s_{e,v}^* = \xi_{v,e}^*$ if $v \in e$ and $s_{e,v}^* = 0$ otherwise. Therefore, it suffices to find ξ_v^* for $v \in V$. The optimality condition of Problem (5.26) is given by

$$\begin{aligned} &p\vartheta_e |\xi_{v,e} - r_{e,v}|^{p-1} \text{sign}(\xi_{v,e} - r_{e,v}) - \vartheta_e \lambda \ni 0, \ \forall e \in E_v, \\ &\lambda \geq 0, \ \Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} \leq d_v, \ \lambda\Big(\Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} - d_v\Big) = 0, \end{aligned}$$

where

$$\text{sign}(a) := \begin{cases} \{-1\}, & \text{if } a < 0, \\ \{1\}, & \text{if } a > 0, \\ [-1, 1] & \text{if } a = 0. \end{cases}$$

There are two cases about λ. We show that in both cases the solution given by Equation (5.10) is optimal.

Case 1. If $\lambda > 0$, then we must have that $p\vartheta_e |\xi_{v,e} - r_{e,v}|^{p-1} > 0$ for all $e \in E_v$ (otherwise, the stationarity condition would be violated). This means that $p|\xi_{v,e} - r_{e,v}|^{p-1} = \lambda$ for all $e \in E_v$, that is, $\xi_{v,e_1} - r_{e_1,v} = \xi_{v,e_2} - r_{e_2,v} > 0$ for every $e_1, e_2 \in E_v$. Denote $t_v := \xi_{v,e} - r_{e,v}$. Because $\lambda > 0$, by complementarity we have

$$\Delta_v - \sum_{e \in E_v} \vartheta_e (t_v + r_{e,v}) = \Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} = d_v,$$

which implies that $t_v = (\sum_{e \in E_v} \vartheta_e)^{-1} (\Delta_v - \sum_{e \in E_v} \vartheta_e r_{e,v} - d_v)$. Note that $\Delta_v - \sum_{e \in E_v} \vartheta_e r_{e,v} - d_v > 0$ because $\Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} - d_v = 0$ and $\xi_{v,e} > r_{e,v}$ for all $e \in E_v$. Therefore we have that

$$s_{e,v}^* = \xi_{v,e}^* = r_{e,v} + d_v^{-1} \Big[\Delta_v - \sum_{e \in E_v} \vartheta_e r_{e,v} - d_v\Big]_+.$$

Case 2. If $\lambda = 0$, then we have that $p\vartheta_e |\xi_{v,e} - r_{e,v}|^{p-1} \text{sign}(\xi_{v,e} - r_{e,v}) \ni 0$ for all $e \in E_v$, which implies $\xi_{v,e} - r_{e,v} = 0$ for all $e \in E_v$. Then we must have

$$\Delta_v - \sum_{e \in E_v} \vartheta_e r_{e,v} = \Delta_v - \sum_{e \in E_v} \vartheta_e \xi_{v,e} \leq d_v.$$

Therefore we still have that

$$s^*_{e,v} = \xi^*_{v,e} = r_{e,v} = r_{e,v} + d_v^{-1}\Big[\Delta_v - \sum_{e \in E_v} \vartheta_e r_{e,v} - d_v\Big]_+.$$

The required result then follows from the definition of A_e and D. This proves Lemma 5.12.

Chapter 6

Weighted Diffusion with Node Attributes

Our primary focus in this chapter is the flow diffusion process introduced in Chapter 4 and in particular its ℓ_2-norm version. We focus on flow diffusion due to its good empirical performance and its flexibility in initializing a diffusion process. The statistical recovery results in this chapter should easily extend to other graph diffusion processes such as (truncated) random walks and Approximate Personalized PageRank. In Section 6.1 we provide a very brief summary of the flow diffusion problem on a weighted graph $G = (V, E, w)$ and its algorithmic solutions. We refer the reader to Section 4.1 for a detailed introduction of the flow diffusion problem, to Section 4.3.1 for interpreting gradient-based optimization algorithms as an iterative diffusion process, to Section 4.3.2 for the special case $p = 2$.

6.1 The Weighted Flow Diffusion Problem

Given a weighted graph $G = (V, E, w)$ where $w \in \mathbb{R}^{|E|}$ specifies the edge weights (i.e., edge e has resistance $1/w_e$), two functions $\Delta, T : V \to \mathbb{R}_+$ which specify the source mass and sink capacity of each node, respectively, the p-norm flow diffusion problem introduced in Chapter 4 with $p = 2$ can be written as

$$\min_f \ \frac{1}{2}\sum_{e\in E} f_e^2/w_e, \quad \text{s.t. } B^T f \le T - \Delta, \tag{6.1}$$

and its dual formulation

$$\min_{x \geq 0} \ \frac{1}{2} x^T L x + x^T (T - \Delta), \tag{6.2}$$

where $B \in \mathbb{R}^{|E| \times |V|}$ is the signed incidence matrix under an arbitrary orientation of the graph, and $L = B^T W B$ is the (weighted) graph Laplacian matrix where $W = \mathrm{Diag}(w)$ is the diagonal matrix of w. Although the above formulations are straightforward generalization of Problem (4.9) and Problem (4.8) to weighted graphs while fixing $p = 2$, in order to emphasize the existence of edge weights (which is what matters for the subject of this chapter), we will refer to the above diffusion problem as the Weighted Flow Diffusion (WFD) problem.

In both Chapter 4 and Chapter 5, we have exclusively considered the sink capacity $T(i) = \deg(i)$ of each node i. In this chapter we extend from this and also consider $T(i) = 1$, which is fixed for every node. Adopting the same argument as in the proof of Lemma 4.1 we immediately obtain the following result on the sparsity of solutions which holds as long as the sink capacity is at least 1 on every node. We make the assumption on sink capacity explicit in Assumption 6.1.

Assumption 6.1. The sink function $T : V \to \mathbb{R}_+$ used in our diffusion Problem (6.1) and Problem (6.2) satisfies $T(i) \geq 1$ for all $i \in V$.

Lemma 6.2. *Let f^* and x^* denote the optimal solutions of Problem (6.1) and Problem (6.2), respectively. Suppose that $T(i) \geq 1$ for all $i \in V$. Then*

1. $|\mathrm{supp}(x^*)| \leq \|\Delta\|_1$;
2. $\mathrm{supp}(f^*) \subseteq \{(i,j) \in E \mid i \in \mathrm{supp}(x^*) \text{ or } j \in \mathrm{supp}(x^*) \text{ or both}\}$.

One may easily extend Algorithm 3 to work with edge weights and solve Problem (6.2). We provide one such algorithm in Algorithm 7. Note that Algorithm 7 directly updates the dual variables x rather than scaled variables z where $z(i) = x(i)/d(i)^{1/2}$ as in Algorithm 3. It is worth to point out that such difference is purely notational and has no effect on diffusion (nor the convergence properties of the algorithm).

Let x^* denote the optimal solution of Problem (6.2) and $\bar{S} = \mathrm{supp}(x^*)$.[1] Then the objective function of Problem (6.2) has coordinate-wise Lipchitz continuous gradient with

[1] One can verify that if Problem (6.2) is bounded, i.e. an optimal solution x^* exists, then $|\mathrm{supp}(x^*)| \leq |V| - 1$.

Algorithm 7 Weighted Flow Diffusion Algorithm

Input: graph $G = (V, E, w)$, source Δ, sink T.

1. Initially, $x(i) = 0$, $\text{mass}(i) = \Delta(i)$ and $\text{excess}(i) = \max\{0, \text{mass}(i) - T(i)\}$, $\forall i \in V$.
2. For $k = 1, 2, \ldots, K$ do
   (a) Define $A^k = \{i \in V \mid \text{excess}(i) > 0\}$.
   (b) For each $i \in A^k$, apply $\texttt{push}(i)$.
3. Return x.

$\texttt{push}(i)$:

Make the following updates:

1. $x(i) = x(i) + \text{excess}(i) / \sum_{j \sim i} w(i, j)$.
2. $\text{mass}(i) = T(i)$.
3. For each node $j \sim i$:

$$\text{mass}(j) = \text{mass}(j) + \text{excess}(i) \cdot \frac{w(i,j)}{\sum_{\ell \sim i} w(i, \ell)},$$
$$\text{excess}(j) = \max\{0, \text{mass}(j) - T(j)\}.$$

constant $d(i) = \sum_{j \sim i} w(i, j)$ for coordinate i, and it is strongly convex with parameter $\Phi^*(\bar{S})^2/2$ over the subset $\{x \in \mathbb{R}^{|V|} \mid \text{supp}(x) \subseteq \bar{S}\}$ of $\mathbb{R}^{|V|}$, where

$$\Phi^*(\bar{S}) = \min_{S' \subseteq \bar{S}} \Phi(S').$$

In the above, the strong convexity parameter follows from lower bounding the smallest Dirichlet eigenvalue $\lambda_{\min}(L_{\bar{S}})$, where $L_{\bar{S}}$ denotes the sub-matrix of the Laplacian matrix L restricted to those indexed by nodes in $\bar{S}$, in terms of the local Cheeger constant $\Phi^*(\bar{S})$, e.g., see Theorem 1 in [36]. Therefore, following the same argument as in Section 4.5.3, we obtain the following rate of convergence for Algorithm 7.

Theorem 6.3. *Let F denote the objective function of Problem (6.2). For any $\epsilon > 0$, after*

$$k = O\left(\frac{\max_{i \in \bar{S}} d(i)}{\Phi^*(\bar{S})^2} \log\left(\frac{\|\Delta\|_1}{\epsilon}\right)\right)$$

iterations, the vector x updated by Algorithm 7 satisfies $\text{supp}(x) \subseteq \text{supp}(x^*)$, and $F(x) - F(x^*) \leq \epsilon$. In addition, the function $\text{excess} : V \to \mathbb{R}_+$ updated by Algorithm 7 satisfies

$$\sum_{i \in V} \text{excess}(i) \leq \epsilon \|\Delta\|_1.$$

According to Lemma 6.2 and Theorem 6.3, if the maximum number of neighbors a node $i \in \text{supp}(x^*)$ does not exceed $\text{poly}(\|\Delta\|_1)$, e.g., there does not exist a node which connects to every other node in the graph, then each iteration of Algorithm 7 can be executed in time at most $O(\text{poly}(\|\Delta\|_1))$. Consequently, the rate of convergence given by Theorem 6.3 implies that that in time $O(\text{poly}(\|\Delta\|_1)\log(1/\epsilon))$, Problem (6.2) can be solved up to ϵ-additive error in function value and ϵ-multiplicative error in gradient norm (cf. Section 4.3.1 for interpreting gradient as excess in the context of diffusion). Since in practice most graphs have bounded node degrees, this means that Algorithm 7 solves Problem (6.2) in strongly local time.

6.2 Local Clustering with Node Attributes

We consider the single seed version of the local graph clustering problem, where we are given a seed node $s \in V$ and the goal is to identify a good cluster that contains the seed. Existing methods mostly focus on the setting where one only has access to the structural information, i.e. nodes and edges of the graph, and they do not take into account node attributes. However, it is reasonable to expect that informative node attributes should help improve the performance of a local clustering algorithm. For example, in Chapter 4 we employ p-norm flow diffusion to solve the local graph clustering problem by spreading source mass from seed node(s) to nearby nodes, and an output cluster is obtained based on where in the graph the mass spread to (recall that the sweep cut procedure is applied to the nonzero dual variables). In this case, node attributes may be used to guide the spread of mass so that more mass are trapped inside the ground-truth target cluster, and consequently, improve the clustering accuracy of the algorithm.

The idea to guide the diffusion by using node attributes can be easily realized by relating edge weights to node attributes. Given a graph $G = (V, E, w)$ with a set of node attributes $X_i \in \mathbb{R}^r$ for $i \in V$, and given a seed node s from an unknown target cluster K, the goal is to recover K. To do so, we construct a new graph $G' = (V, E, w')$ having the same structure but refined edge weights

$$w'(i, j) = \rho(X_i, X_j) \cdot w(i, j)$$

where $\rho(X_i, X_j)$ measures the proximity between X_i and X_j. In this case, for a weighted flow diffusion in G', if two adjacent nodes i and j have similar attributes, then it is easier to send a lot of mass along the edge (i, j). In particular, when one removes the excess mass from a node i by sending it to the neighbors, the amount of mass that a neighbor j receives is proportional to $w'(i, j)$ (cf. Step 3 of `push`(i) in Algorithm 7), and hence more

mass will be sent to a neighbor whose attributes also bear close proximity. Therefore, if nodes within the target cluster K share similar attributes, then a weighted flow diffusion in G', which starts from a seed node $s \in K$, would naturally force more mass to spread within K than a flow diffusion in the original graph G.

Algorithm 8 Local Graph Clustering with Node Attributes

Input: graph $G = (V, E, w)$, node attributes X_i for all $i \in V$, seed node $s \in V$, hyperparameter $\gamma \geq 0$.

Output: a cluster $K' \subseteq V$.

1. Define refined graph $G' = (V, E, w')$ whose edge weights are given by $w'(i, j) = w(i, j) \cdot \exp(-\gamma \|X_i - X_j\|_2^2)$.
2. Set source mass $\Delta(s) > 0$ and $\Delta(i) = 0$ for $i \neq s$. Set sink capacity $T(i)$.
3. Run Algorithm 7 with input G', Δ, T and obtain output x'.
4. Return $K' = \mathrm{supp}(x')$

In this work, we use the Gaussian kernel to measure the similarity between node attributes, that is, we consider

$$\rho(X_i, X_j) = \exp(-\gamma \|X_i - X_j\|_2^2)$$

where $\gamma \geq 0$ is a hyperparameter. The Gaussian kernel is one of the most widely used metrics of similarity and has proved useful in many applications such as spectral clustering. Later we will provide rigorous statistical guarantees on the performance of local graph clustering with node attributes by using the optimal solution of Problem (6.2), where edge weights are defined by the Gaussian kernel for an appropriately chosen $\gamma > 0$. We focus on the Gaussian kernel for its simplicity. Both the algorithm in this section and its analysis can be easily extended to work with other metrics such as the Laplacian kernel, polynomial kernel and cosine similarity.

We summarize the local clustering procedure in Algorithm 8. We will show that suitable choices for the sink function T include $T(i) = 1$ or $T(i) = \deg^\circ(i) = |\{j \in V \mid j \sim i\}|$ for all i, and one may correspondingly set the source mass $\Delta(s) = \alpha \sum_{i \in K} T(i)$ for $\alpha > 1$ where K is the target cluster. In practice, one does not need to know the exact value of $\sum_{i \in K} T(i)$. As we demonstrate in the experiments in Section 6.3, a rough estimate of the size of K (e.g. $|K|$ or $\mathrm{vol}(K)$) within a constant multiplicative factor already suffices for a good local clustering performance. Finally, note that Algorithm 8 can be implemented to maintain the locality nature of weighted flow diffusion: Starting from the seed node, executing Algorithm 8 requires access to a new node, its sink capacity and attributes only if they become necessary for subsequent computations. For example, one should never compute

an edge weight if that edge is not needed to diffuse mass. In theory, Algorithm 7 returns x^τ after τ number of iterations, and if τ is not large enough, it may happen that $\text{supp}(x^\tau)$ is a strict subset of $\text{supp}(x^*)$, where x^* denotes the optimal solution of Problem (6.2). However, due to convergence (cf. Proposition 4.16) we know that there always exists $\tau_0 \in \mathbb{N}$ such that $\text{supp}(x^\tau) = \text{supp}(x^*)$ for all $\tau \geq \tau_0$. We thus make the following assumption that for exponentially small tolerance ϵ we get $\text{supp}(x^\tau) = \text{supp}(x^*)$, which seems to hold for all practical purposes. Under this assumption, Algorithm 8 outputs $K' = \text{supp}(x^*)$ in time $O(\text{poly}(\|\Delta\|_1))$. Assumption 6.4 simplifies our analysis a bit because it allows us to directly analyze $\text{supp}(x^*)$ using the properties of x^*. On the other hand, Assumption 6.4 is not required for Algorithm 8 to produce good output. For an approximate solution x^τ where $\text{supp}(x^\tau) \neq \text{supp}(x^*)$, the same lines of argument used in our proofs can be applied to analyze $\text{supp}(x^\tau)$ and obtain similar results to Theorem 6.8 and Theorem 6.9.

Assumption 6.4. There is $\epsilon_0 = \Omega(e^{-\text{poly}(\|\Delta\|_1)})$ such that whenever $F(x^\tau) - F(x^*) \leq \epsilon_0$ then $\text{supp}(x^\tau) = \text{supp}(x^*)$, where x^* denotes the optimal solution of Problem (6.2) and x^τ denotes the dual variables obtained after τ iterations of Algorithm 7.

6.2.1 Statistical Recovery in Contextual Random Graphs

We assume that the node attributes and a target cluster are generated from the following random model. For simplicity in discussion we will assume that the random model generates unweighted graphs, although one may easily obtain identical results for weighted graphs whose edges weights do not scale with the number of nodes n.

Definition 6.5 (Contextual local random model)**.** Given a set of nodes V, let $K \subseteq V$ be a target cluster with cardinality $|K| = k$. For every pair of nodes i and j, if $i, j \in K$ then we draw an edge (i, j) with probability p; if $i \in K$ and $j \notin K$ then we draw an edge (i, j) with probability q; otherwise, we allow any (deterministic or random) model to draw an edge. The node attributes X_i for a node i are given as $X_i = \mu_i + Z_i$, where $\mu_i \in \mathbb{R}^r$ is a fixed signal vector and $Z_i \in \mathbb{R}^r$ is a random noise vector whose ℓ^{th} coordinate $Z_{i\ell}$ follows independent mean zero sub-Gaussian distribution with variance proxy σ_ℓ, i.e., for any $t \geq 0$ we have $\mathbb{P}(|Z_{i\ell}| \geq t) \leq 2\exp(-\frac{t^2}{2\sigma_\ell^2})$. Though not necessary, to simplify the discussion we require $\mu_i = \mu_j$ for $i, j \in K$.

This random model is fairly general. For example, if the edges that connect nodes in $V \backslash K$ have been generated from the Stochastic Block Model (SBM), $\mu_i = \mu_j$ for every i, j that belong to the same community, and all Z_i's follow the same isotropic Gaussian distribution, then we obtain the Contextual Stochastic Block Model (CSBM) which has

been extensively used in the analyses of algorithms for attributed graphs [39, 17, 134]. On the other hand, if the edges that connect nodes in $V \backslash K$ have been generated from the Erdős-Renyi model with probability q, $\mu_i = \mu_j \neq 0$ for $i, j \in K$ and $\mu_i = 0$ for $i \notin K$, and all Z_i's follow the same isotropic Gaussian distribution, then we obtain a natural coupling of the planted densest subgraph problem and the submatrix localization problem [29]. In terms of modelling the noise of node attributes, sub-Gaussian distributions include Gaussian, Bernoulli, and any other continuous or discrete distribution over finite domains. Therefore the random model allows different types of coordinate-wise noise (and varying levels of noise controlled by σ_ℓ) which could depend on the nature of the specific attribute. For example, the noise of a continuous attribute may be Gaussian or uniform, whereas the noise of a binary-encoded categorical attribute may be Bernoulli.

In order for node attributes to provide useful information, nodes inside K should have distinguishable attributes compared to nodes not in K. Denote

$$\hat{\mu} := \min_{i \in K, j \notin K} \|\mu_i - \mu_j\|_2, \quad \hat{\sigma} := \max_{1 \leq \ell \leq d} \sigma_\ell.$$

We make Assumption 6.6 which states that the relative signal $\hat{\mu}$ dominates the maximum coordinate-wise noise $\hat{\sigma}$, and that the sum of normalized noises does not grow faster than $\log n$. The latter assumption is easily satisfied, e.g., when the dimension r of node attributes does not scale with the number of nodes n. In practice, when the set of available or measurable attributes are fixed a priori, one always has $r = O_n(1)$. This is particularly relevant in the context of local clustering where it is desirable to have sublinear algorithms, since if $r = \Omega(n)$ then even computing a single edge weight $w(i, j) = \exp(-\gamma \|X_i - X_j\|_2^2)$ would take time at least linear in n.

Assumption 6.6. $\hat{\mu} = \omega(\hat{\sigma}\sqrt{\lambda \log n})$ for some $\lambda = \Omega_n(1)$; $\sum_{\ell=1}^{r} \sigma_\ell^2 / \hat{\sigma}^2 = O(\log n)$.

Before we move on to discuss how exactly node attributes help recover K, we need to talk about the signal and noise from the graph structure. For a node $i \in K$, the expected number of neighbors in K is $p(k-1)$, and the expected number of neighbors not in K is $q(n-k)$. Since mass spreads along edges, if there are too many edges connecting K to $V \backslash K$, it may become difficult to prevent a lot of mass from flowing out of K. The consequence of having too much mass which starts in K to leak out of K is that $\mathrm{supp}(x^*)$ may have little overlap with K, and consequently Algorithm 8 would have poor performance.

Fortunately, node attributes may be very helpful when the structural information is not strong enough, e.g., when $q(n-k) > p(k-1)$. As discussed earlier, informative node attributes should be able to guide the spread of mass in the graph. In a weighted flow diffusion, the location where mass will spreads to from the source node depends on the edge

weights. The higher weight an edge has, the easier to send mass along that edge. Therefore, in order to keep as much mass as possible inside the target cluster K, an ideal situation would be that edges inside K have significantly more weights than an edge that connects K to $V\backslash K$. It turns out that this is exactly the case when we have good node attributes. By applying concentration results on the sum of squares of sub-Gaussian random variables, Lemma 6.7 says that, with overwhelming probability, one obtains a desirable separation of edge weights as a consequence of node attributes having more signal than noise, i.e. when Assumption 6.6 holds. Lemma 6.7 follows from concentration inequalities for sums of sub-exponential and sub-Gaussian random variables. We leave details to Section 6.4.2.

Lemma 6.7. *Under Assumption 6.6, one may pick γ such that $\gamma\hat{\sigma}^2 = o(\log^{-1} n)$ and $\gamma\hat{\mu}^2 = \omega_n(\lambda)$. Consequently, with probability at least $1 - o_n(1)$, the edge weight $w_{ij} = \exp(-\gamma\|X_i - X_j\|_2^2)$ satisfies $w_{ij} \geq 1 - o_n(1)$ for all $i, j \in K$, and $w_{ij} \leq \exp(-\omega_n(\lambda))$ for all $i \in K, j \notin K$.*

Not surprisingly, Lemma 6.7 implies that the gap between edge weights is controlled by λ which, according to Assumption 6.6, measures how strong the attribute signal is. If λ is sufficiently large, then naturally one would expect an algorithm that uses the node attributes to nearly perfectly recover K, irrespective of how noisy the graph structure is. Otherwise, the performance to recover K would depend on a combination of both structural and attribute information. In what follows we present two recovery results which precisely correspond to these two scenarios. In all probability bounds, we keep explicit dependence on the cluster size k because, for local graph clustering, k may be a large constant and does not necessarily scale with n.

Theorem 6.8 (Recovery with very good node attributes)**.** *Under Assumption 6.6, for any γ satisfying $\gamma\hat{\sigma}^2 = o(\log^{-1} n)$ and $\gamma\hat{\mu}^2 = \omega_n(\lambda)$, with source mass $\Delta(s) = (1+\beta)\sum_{i\in K} T(i)$ for any $\beta > 0$,*

1. *if K is connected and $\lambda = \Omega_n(\log k + \log(1/\beta) + \log(q(n-k)))$, then with probability at least $1 - o_n(1) - k^{-1/3}$, for every seed node $s \in K$ we have $K \subseteq \mathrm{supp}(x^*)$ and $\sum_{i\in\mathrm{supp}(x^*)\backslash K} T(i) \leq \beta\sum_{i\in K} T(i)$;*
2. *if $p \geq \frac{(4+\epsilon)}{\delta^2}\frac{\log k}{k-1}$ for some $0 < \delta < 1$ and $\epsilon > 0$, and $\lambda = \Omega_n(\log k + \log(1/\beta) + \log(\frac{q(n-k)}{p(k-1)}) + \log(\frac{1}{1-\delta}))$, then with probability at least $1 - o_n(1) - k^{-1/3} - ek^{-\epsilon/2}$, for every seed node $s \in K$ we have $K \subseteq \mathrm{supp}(x^*)$ and $\sum_{i\in\mathrm{supp}(x^*)\backslash K} T(i) \leq \beta\sum_{i\in K} T(i)$.*

In particular, we obtain the following bounds on false positives: if $T(i) = 1$ for all $i \in V$ then

$$|\mathrm{supp}(x^*)\backslash K| \leq \beta|K|;$$

if $T(i) = \deg^\circ(i)$ *for all* $i \in V$ *then*

$$\mathrm{vol}^\circ(\mathrm{supp}(x^*)\backslash K) \leq \beta \mathrm{vol}^\circ(K).$$

Some discussions are in order. The first part of Theorem 6.8 does not assume anything about the internal connectivity of K. It applies as long as K is connected, and this includes the extreme case when the induced subgraph on K is a tree but each node in K is also connected to many other nodes not in K. The second part of Theorem 6.8 requires a weaker condition on the strength of attribute signal $\hat{\mu}$. The additive term $\log(q(n-k))$ from part 1 is weakened to $\log(\frac{q(n-k)}{p(k-1)})$ due to the improved connectivity of K, under the additional assumption that $p \geq \Omega(\log k/k)$. We consider two specific choices of T. The first choice gives the exact bound on the number of false positives, and the second choice bounds the size of false positives in terms of volume. Note that even in the case where the node attributes alone provide sufficient signal, the graph structure still plays a very important role as it allows the possibility that an algorithm could return a good output without having to explore all data points. For example, during the execution of Algorithm 8, one only needs to query the attributes of a node whenever they are required for subsequent computations.

Let us introduce one more notion before presenting the recovery guarantee with good, but not too good, node attributes. Given the contextual random model described in Definition 6.5, consider a "population" graph $\bar{G} = (V, \bar{E}, \bar{w})$ where $(i,j) \in \bar{E}$ for every pair i, j such that $i \neq j$, and the edge weight $\bar{w}_{ij}$ satisfies $\bar{w}_{ij} = p\exp(-\gamma\|\mathbb{E}[X_i] - \mathbb{E}[X_j]\|_2^2) = p$ if $i, j \in K$, $\bar{w}_{ij} = q\exp(-\gamma\|\mathbb{E}[X_i] - \mathbb{E}[X_j]\|_2^2) \leq qe^{-\gamma\hat{\mu}^2}$ if $i \in K, j \notin K$. A frequently used measure of cluster quality is conductance which quantifies the ratio between external and internal connectivity. For a set of nodes C in $\bar{G}$, its conductance is

$$\Phi_{\bar{G}}(C) = \frac{\sum_{i\in C, j\notin C} \bar{w}_{ij}}{\sum_{i\in C}\sum_{j\sim i} \bar{w}_{ij}}.$$

For $0 \leq c \leq 1$ denote

$$\eta(c) = \frac{p(k-1)}{p(k-1) + q(n-k)e^{-c\gamma\hat{\mu}^2}}.$$

For the target cluster K, one easily verifies that

$$\Phi_{\bar{G}}(K) \leq 1 - \eta(1) < 1 - \eta(0) = \Phi_G(K),$$

where $\Phi_G(K)$ is the conductance of K in the original unweighted graph G. Intuitively, a low conductance cluster is better connected internally than externally, and thus it should

be easier to detect. Therefore, the advantage of having node attributes is that they help reduce the conductance of the target cluster, making it easier to recover from the population graph. While in practice one never works with the population graph, our next theorem indicates that, with overwhelming probability, the recoverability of K in the population graph transfers to an realization of the random model in Definition 6.5. More specifically, Theorem 6.9 says that when the node attributes are good, i.e. Assumption 6.6 holds, but not too good, i.e. conditions required in Theorem 6.8 may not hold, then Algorithm 8 still fully recovers K as long as there is sufficient internal connection. Moreover, the relative size of false positives (compared to the size of K) is upper bounded by $O(1/\eta(c)^2) - 1$ for any $c < 1$. Denote

$$m(\delta_1, \delta_2) = \frac{(1 + 3\delta_1 + \frac{1}{p(k-1)})^2}{(1-\delta_1)(1-\delta_2)}, \qquad T_{\max} = \max_{i \in K} T(i).$$

Theorem 6.9 (Recovery with good node attributes)**.** *Under Assumption 6.6, if* $p \geq \max(\frac{(3+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-1}, \frac{(2+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-1}})$ *where* $0 < \delta_1, \delta_2 \leq 1$ *and* $\epsilon_1, \epsilon_2 > 0$*, then with probability at least* $1 - o_n(1) - 4k^{-\epsilon_1/3} - k^{-2\epsilon_2}$*, for every seed node* $s \in K$ *with initial seed mass*

$$\Delta(s) = c_1 T_{\max} \frac{m(\delta_1, \delta_2) k}{\eta(c_2)^2}$$

for any constants $c_1 > 1$ *and* $c_2 < 1$*, we have that* $K \subseteq \mathrm{supp}(x^*)$*. Moreover, if* $T(i) = 1$ *for all* $i \in V$ *then*

$$|\mathrm{supp}(x^*) \backslash K| \leq \left(\frac{c_1 m(\delta_1, \delta_2)}{\eta(c_2)^2} - 1 \right) |K|;$$

if $T(i) = \deg^\circ(i)$ *for all* $i \in V$

$$\mathrm{vol}^\circ(\mathrm{supp}(x^*) \backslash K) \leq \left(\frac{c_1 m(\delta_1, \delta_2)}{\eta(c_2)^2} \frac{(1+\delta_1)}{(1-\delta_1)} - 1 \right) \mathrm{vol}^\circ(K).$$

In the special case where there is no node attribute, we may simply take $\hat{\mu} = 0$ and Theorem 6.9 still holds. For this specific setting we obtain a nearly identical recovery result (i.e. same assumption and same result) that has been previously obtained for local graph clustering using PageRank vectors without node attributes [60], where the relative size of false positives is $O(1/\eta(0)^2 - 1)$. This comparison precisely quantifies the advantage of having good node attributes as they reduce the bound to $O(1/\eta(c)^2 - 1)$ for any $c < 1$, which can be substantially smaller. Note that the expression $1/\eta(c)^2$ is jointly controlled by the combinatorial conductance of K and the attribute signal $\hat{\mu}$. Theorem 6.9 follows from upper bounding the total amount of mass that leaks to the outside of the target cluster K during a diffusion process, while using minimal amount of source mass to saturate every node in K. We provide details of the proof in Section 6.4.4.

6.3 Empirical Results

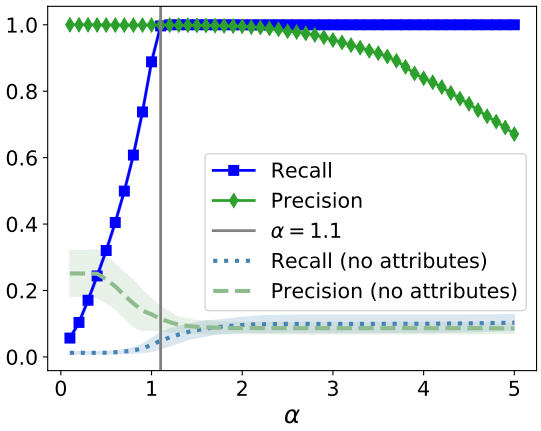
We evaluate the performance of Algorithm 8 for local clustering with node attributes.[2] First, we investigate empirically our theoretical results over synthetic data generated from a specification of the random model described in Definition 6.5. We use the synthetic experiments to demonstrate (i) the distinction between having weak and strong graph structural information, and (ii) the distinction between having very good and moderately good node attributes. In addition, the synthetic experiment indicates the necessity of Assumption 6.6 in order for Algorithm 8 to have notable performance improvement against method that does not use node attributes. Second, we carry out experiments using real-world data. We show that incorporating node attributes improves the F1 scores by an average of 4.3% over 20 clusters from two academic co-authorship networks.

6.3.1 Experiments on Synthetic Graphs

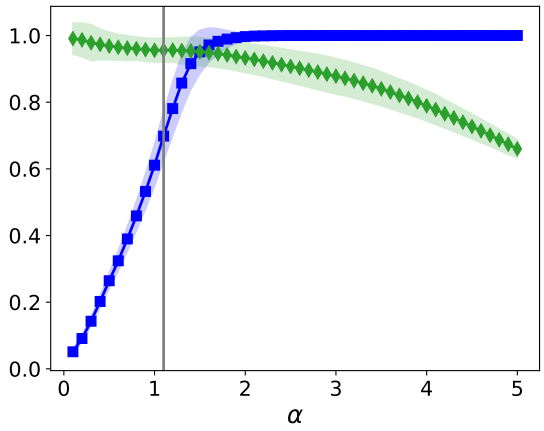
We generate random graphs using the stochastic block model with community size $k = 500$ and the total number of clusters $r = 20$. The total number of nodes is $n = kr = 10,000$. Two nodes within the same cluster are connected with probability p, and two nodes from different clusters are connected with probability q. We fix $q = 0.002$ and vary p to control the strength of the structural signal. We randomly pick one of the clusters as the target cluster K. The dimension of the node attributes is set to $d = 100$. For node attributes $X_i = \mu_i + Z_i$, we sample Z_i from Gaussian distribution with mean 0 and identity covariance. Therefore $\sigma_\ell = 1$ for all $\ell = 1, 2, \ldots, d$, and hence $\hat{\sigma} = 1$. We set $\mu_{i\ell} = a\hat{\sigma}\sqrt{\log n}/2\sqrt{d}$ for all ℓ if $i \in K$, and $\mu_{i\ell} = -a\hat{\sigma}\sqrt{\log n}/2\sqrt{d}$ for all ℓ if $i \notin K$. In this way, we get that $\hat{\mu} = \max_{i\in K, j\notin K} \|\mu_i - \mu_j\|_2 = a\hat{\sigma}\sqrt{\log n}$. We vary a to control the strength of node attribute signal.

We set the sink capacity $T_i = 1$ for all i. We set the source mass $\Delta_s = \alpha k$ and we allow α to vary. We set $\gamma = (\log^{-3/2} n)/4\hat{\sigma}^2$ so that $\gamma\hat{\sigma}^2 = o(\log^{-1} n)$ as required by Theorem 6.8 and Theorem 6.9. To measure the quality of an output cluster $C := \mathrm{supp}(x^\tau)$, we use precision and recall which are defined as $|C \cap K|/|C|$ and $|C \cap K|/|K|$, respectively. The F1 score is the harmonic mean of precision and recall given by $2/(\mathrm{Precision}^{-1} + \mathrm{Recall}^{-1})$. For comparison we also consider the performance of unweighted flow diffusion which does not use node attributes. There are other methods for local graph clustering without node attributes, such as the ℓ_1-regularized PageRank [8, 60]. We does not compare with these methods here as in Section 4.4 flow diffusion is shown to achieve better performance.

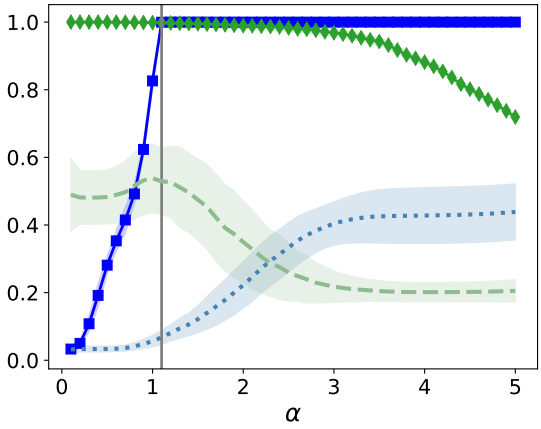
[2] Code is available at https://github.com/s-h-yang/WFD.



(a) $p = 0.01, q = 0.002, \hat{\mu} = 3\hat{\sigma} \log n$



(b) $p = 0.01, q = 0.002, \hat{\mu} = \frac{5}{2}\hat{\sigma} \log n$



(c) $p = 0.03, q = 0.002, \hat{\mu} = \frac{5}{2}\hat{\sigma} \log n$

Figure 6.1: Demonstration of Theorem 6.8. The lines show average performance over 100 trails. In each trial we randomly pick a seed node s from the target cluster K. The error bars show standard deviation. Figure 6.1a and Figure 6.1c show full recovery of K as soon as $\alpha > 1$ (i.e. as soon as $\beta > 0$, see first part of Theorem 6.8). The distinction between Figure 6.1b and Figure 6.1c demonstrate that the required threshold for $\hat{\mu}$ depends on p (cf. second part of Theorem 6.8). With very good node attributes, the performance of flow diffusion that uses node attributes is significantly better than the performance of flow diffusion that does not use node attributes.

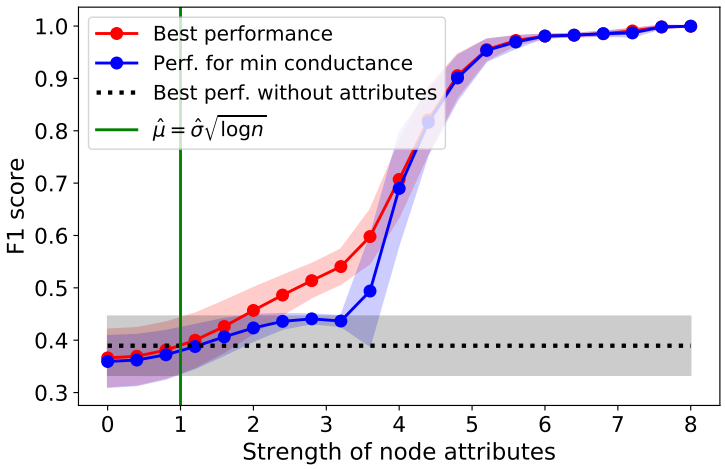


Figure 6.2: Performance of Algorithm 8 as $\hat{\mu}$ increases. $\hat{\mu}$ needs to be larger than $\hat{\sigma}\sqrt{\log n}$ in order for node attributes to be useful. The x-axis shows the value of a where $\hat{\mu} = a\hat{\sigma}\sqrt{\log n}$. We average over 100 trials, each trial uses a randomly selected seed node.

Moreover, the comparison between weighted and unweighted flow diffusion, which either use or does not use node attributes, allows us to obtain a fair estimate on the benefits of node attributes.

Figure 6.1 shows detailed views of the performance of Algorithm 8 as we vary α between $[0.1, 5]$ with 0.1 increments. It is used to demonstrate the two claims of Theorem 6.8. In Figure 6.1a, we set $p = 0.01 < \log k/k$, so the target cluster K is very sparse. On average, each node $i \in K$ only has 5 neighbors inside K while it has 19 neighbors outside of K. This means that the graph structural information alone is not very helpful for recovering K. On the other hand, we set $a = 3\sqrt{\log n}$ so $\hat{\mu} = 3\hat{\sigma}\log n$. This means that the node attributes contain very strong signal. In this case, observe that as soon as α becomes strictly larger than 1, the output cluster C fully recovers K, i.e. Recall $= 1$. This demonstrates the first claim of Theorem 6.8. As a comparison, the unweighted flow diffusion which does not use node attributes has very poor performance for every choice of α. This is expected because edge connectivity reveals very little clustering information. In Figure 6.1b, we keep the same graph structure but slightly weaken the node attributes to $\hat{\mu} = \frac{5}{2}\hat{\sigma}\log n$ by reducing a. This stops the output cluster C from fully recovering K for small α larger than 1. The algorithm still has a good performance if one chooses α properly. This scenario is covered by Theorem 6.9 and we will discuss more about it later. In Figure 6.1c, we keep the same node attributes as in Figure 6.1b but increase p from 0.01 to 0.03 which is slightly larger than $2\log k/k$. In this case, the output cluster C again fully recovers K as soon as α is

strictly larger than 1. The distinction between Figure 6.1b and Figure 6.1c means that the required threshold for $\hat{\mu}$ to fully recover K at any $\alpha > 1$ decreases as p increases. This demonstrates the second claim of Theorem 6.8.

In Figure 6.2 we consider a more realistic setting where one may not know the size of the target cluster K and the node attributes may be noisy. We keep the same graph connectivity (i.e. $p = 0.03$ and $q = 0.002$) and vary a between $[0, 8]$ with 0.5 increments. Recall that the node attributes are set in a way such that $\hat{\mu} = a\hat{\sigma}\sqrt{\log n}$, therefore the strength of node attributes increases as a increases. For each choice of a, given a seed node s, we run Algorithm 8 multiple times with source mass αk for $\alpha \in \{1.1, 1.6, \ldots, 10.1\}$. This gives multiple output clusters, one from each choice of α. We consider two cases for selecting a final cluster. The first case is a best-case scenario where we pick the cluster that achieves the best F1 score, the second case is a more realistic case where we pick the cluster that has the minimum conductance. Figure 6.2 illustrates the performance of Algorithm 8 in these two cases. The x-axis of Figure 6.2 is the value of a where $\hat{\mu} = a\hat{\sigma}\sqrt{\log n}$. Overall, the performance improves as $\hat{\mu}$ increases. When the node attributes are reasonably strong, e.g. $a \geq 4$, the scenario where we select a cluster based on minimum conductance matches with the best-case performance. Note that, the higher $\hat{\mu}$ is, the lower $\eta(c)$ is for any $0 < c \leq 1$, and according to Theorem 6.9, there should be less false positives and hence a higher F1 score. This is exactly what Figure 6.2 shows. In Figure 6.2 we also plot the best-case performance of unweighted flow diffusion without node attributes. When the node attributes are very noisy, and in particular, when $\hat{\mu} \leq \hat{\sigma}\sqrt{\log n}$ where Assumption 6.6 clearly fails, we see that using node attributes can be harmful as it can lead to worse performance than not using node attributes at all. On the other hand, once the node attributes become strong enough, e.g., $a \geq 4$, using node attributes start to yield much better outcome.

6.3.2 Experiments on Real-World Graphs

We evaluate the performance of Algorithm 8 on two co-authorship graphs based on the Microsoft Academic Graph from the KDD Cup 2016 challenge [106]. In these graphs, nodes are authors, and two nodes are connected by an edge if they have coauthored a paper. The clusters are defined according to the most active research field of each author. The node attributes represent paper keywords for each author's papers. The first graph consists of 18,333 computer science researchers and 81,894 connections among them. Each computer science researcher belongs to one of the 15 ground-truth clusters. The node attributes consists of 6,805 key words. The second graph consists of 34,493 physics researchers and

Table 6.1: Cluster statistics in co-authorship graphs

Network	Cluster	Number of nodes	Volume
Computer Science	Bioinformatics	708	3767
	Machine Learning	462	4387
	Computer Vision	2050	20384
	NLP	429	2476
	Graphics	1394	15429
	Networks	2193	18364
	Security	371	2493
	Databases	924	9954
	Data Mining	775	7573
	Game Theory	118	362
	HCI	1444	15145
	Information Theory	2033	16007
	Medical Informatics	420	3838
	Robotics	4136	33708
	Theoretical CS	876	9901
	TOTAL	18333	163788
Physics	Phys. Rev. A	5750	52151
	Phys. Rev. B	5045	54853
	Phys. Rev. C	17426	325475
	Phys. Rev. D	2753	40451
	Phys. Rev. E	3519	22994
	TOTAL	34493	495924

247,962 connections among them. Each physics researcher belongs to one of the 5 ground-truth clusters. The node attributes consists of 8,415 key words. The cluster sizes are given in Table 6.1.

For both graphs, we preprocess the node attributes by applying PCA to reduce the dimension to 128. In addition, for each node we enhance its attributes by taking a uniform average over its own attributes and the neighbors' attributes. Uniform averaging of neighborhood attributes has been shown to improve the signal-to-noise ratio in CSBM [17]. This operation does not break the local nature of Algorithm 8 because it only needs to be done whenever it becomes necessary for subsequent computations, i.e., when a node is looked at by Algorithm 8. We consider the ground-truth communities as the target clusters. We

Table 6.2: F1 scores for local clustering in co-authorship networks

		$T(i) = \deg^\circ(i)$ for all i			$T(i) = 1$ for all i		
Network	Cluster	No attr.	Ues attr.	Improv.	No attr.	Ues attr.	Improv.
Computer Science	Bioinformatics	32.1	39.3	7.2	23.5	31.7	8.2
	Machine Learning	30.9	37.3	6.4	27.5	34.4	6.9
	Computer Vision	37.6	35.5	-2.1	40.4	37.8	-2.6
	NLP	45.2	52.3	7.1	34.3	37.2	2.9
	Graphics	38.6	49.2	10.6	39.1	41.3	2.2
	Networks	44.1	47.0	2.9	43.0	44.1	1.1
	Security	29.9	35.7	5.8	23.0	26.2	3.2
	Databases	48.5	58.1	9.6	41.9	42.6	0.7
	Data Mining	27.5	28.8	1.3	26.2	28.6	2.4
	Game Theory	60.6	66.0	5.4	56.9	62.6	5.7
	HCI	70.0	77.6	7.6	44.0	63.1	19.1
	Information Theory	47.4	46.9	-0.5	41.6	41.4	-0.2
	Medical Informatics	65.7	70.3	4.6	62.7	68.1	5.4
	Robotics	59.9	59.9	0.0	58.8	55.9	-2.9
	Theoretical CS	66.3	70.7	4.4	54.9	59.1	4.2
Physics	Phys. Rev. A	69.4	70.9	1.5	53.5	60.9	7.4
	Phys. Rev. B	41.4	42.3	0.9	40.4	41.1	0.7
	Phys. Rev. C	79.3	82.1	2.8	84.9	85.9	1.0
	Phys. Rev. D	62.3	68.9	6.6	63.6	70.0	6.4
	Phys. Rev. E	49.5	53.7	4.2	30.1	34.9	4.8
	AVERAGE	50.3	54.6	4.3	44.5	48.3	3.8

consider two choices for the sink capacities T. The first is $T(i) = \deg^\circ(i)$ for all i and the second is $T(i) = 1$ for all i. For each target cluster K in a graph, given a seed node $s \in K$, we run Algorithm 8 with source mass $\Delta(s) = \alpha \sum_{i \in K} T(i)$ for $\alpha \in \{1.5, 1.75, 2, \ldots, 5\}$. We select the output cluster that has the minimum conductance and measure the recovery quality using the F1 score. For each of the 20 target clusters we run 100 trials and for each trial we use a different seed node. We report the average F1 scores (as percentage) in Table 6.2. For both graphs, we find that setting the sink capacities to be equal to the node degrees generally yields a better clustering result than setting the sink capacities to 1. In most cases, incorporating node attributes improves recovery accuracy. Over the total 20 clusters in the two co-authorship networks, using node attributes increases the F1 score by 4.3% on average.

6.4 Proofs of Results

6.4.1 Technical Lemmas

We start with some technical lemmas which will be used later in the proofs.

Lemma 6.10 (Lower bound of internal cut)**.** *For any $0 < \delta \leq 1$ and $\epsilon > 0$, if $p \geq \frac{(4+\epsilon)}{\delta^2} \frac{\log k}{k-1}$ and $k \geq 20$, then with probability at least $1 - ek^{-\epsilon/2}$ we have that*

$$|\partial_K(C)| = |\{(i,j) \in E \mid i \in K, j \in K \backslash C\}| \geq (1-\delta)p(k-1)$$

for all proper subsets $C \subset K$.

Proof. Consider integers j such that $1 \leq j \leq k/2$. First fix some j and let $C \subset K$ be such that $|C| = j$. Note that $|\partial_K(C)|$ is the sum of $j(k-j)$ independent Bernoulli random variables with expectation $\mathbb{E}(|\partial_K(C)|) = pj(k-j)$. Therefore we may apply the Chernoff bound and get

$$\mathbb{P}(|\partial_K(C)| \leq (1-\delta)p(k-1)) \leq e^{-pj(k-j)} \left(\frac{ej(k-j)}{(1-\delta)(k-1)} \right)^{(1-\delta)p(k-1)}.$$

By a union bound over all subsets $C \subset K$ such that $|C| = j$ we get that

$$\mathbb{P}\left(|\partial_K(C)| \leq (1-\delta)p(k-1), \forall C \subset K \text{ s.t. } |C| = j\right)$$

$$
\begin{aligned}
&\leq \binom{k}{j} e^{-pj(k-j)} \left(\frac{ej(k-j)}{(1-\delta)(k-1)} \right)^{(1-\delta)p(k-1)} \\
&\leq \left(\frac{ek}{j} \right)^{j} \exp \Bigg[-pj(k-j) + (1-\delta)p(k-1) + \\
&\qquad\qquad (1-\delta)p(k-1) \log \left(\frac{j(k-j)}{(1-\delta)(k-1)} \right) \Bigg] \\
&= \exp \Bigg[j + j \log \left(\frac{k}{j} \right) - pj(k-j) + (1-\delta)p(k-1) + \\
&\qquad\qquad (1-\delta)p(k-1) \log \left(\frac{j(k-j)}{(1-\delta)(k-1)} \right) \Bigg]. \qquad (6.3)
\end{aligned}
$$

Now consider the exponent in (6.3),

$$
\begin{aligned}
f(j) = j + j \log \left(\frac{k}{j} \right) - pj(k-j) + \\
(1-\delta)p(k-1) + (1-\delta)p(k-1) \log \left(\frac{j(k-j)}{(1-\delta)(k-1)} \right),
\end{aligned}
$$

we will show that $f(j) \leq -(1+\epsilon/2)\log k + 1$ for all $1 \leq j \leq k/2$ and $k \geq 20$. Let us first consider the interval $[1, 3k/8]$. The derivative of $f(j)$ with respect to j is

$$
f'(j) = -p(k-2j) + (1-\delta)p(k-1)\frac{(k-2j)}{j(k-j)} + \log \left(\frac{k}{j} \right),
$$

and we have that $f'(j) \leq 0$ for all $1 \leq j \leq 3k/8$. To see this, for $1 \leq j \leq k/2$ we have

$$
\begin{aligned}
\frac{(k-1)}{j(k-j)} \leq 1 &\iff \frac{(1-\delta)p(k-1)(k-2j)}{j(k-j)} \leq (1-\delta)p(k-2j) \\
&\iff -p(k-2j) + (1-\delta)p(k-1)\frac{(k-2j)}{j(k-j)} \leq -\delta p(k-2j),
\end{aligned} \qquad (6.4)
$$

moreover, since $p \geq \frac{(4+\epsilon)}{\delta^2}\frac{\log k}{k-1}$, for $1 \leq j \leq 3k/8$ and $k \geq 2$ we have

$$
-\delta p(k-2j) \leq -\frac{\delta pk}{4} \leq -\frac{(4+\epsilon)k}{4\delta(k-1)} \log k \leq -\log k \leq -\log(k/j), \qquad (6.5)
$$

and thus by combining (6.4) and (6.5) we get $f'(j) \le -\delta p(k-2j) + \log(k/j) \le 0$ for all $1 \le j \le 3k/8$. This implies that $f(j)$ achieves maximum at $j = 1$ over the interval $[1, 3k/8]$. Therefore, for all $1 \le j \le 3k/8$,

$$\begin{aligned} f(j) \le f(1) &= -p(k-1) + (1-\delta)p(k-1) - (1-\delta)p(k-1)\log(1-\delta) + 1 + \log k \\ &= -p(k-1)(\delta + (1-\delta)\log(1-\delta)) + 1 + \log k \\ &\le -p(k-1)\delta^2/2 + 1 + \log k \\ &\le -(2+\epsilon/2)\log k + 1 + \log k \\ &= -(1+\epsilon/2)\log k + 1 \end{aligned}$$

where the second inequality follows from the numeric inequality $\delta + (1-\delta)\log(1-\delta) \ge \delta^2/2$ for $\delta \in (0,1)$, and the third inequality follows from the assumption that $p \ge \frac{(4+\epsilon)}{\delta^2}\frac{\log k}{k-1}$.

Next, consider the value of $f(j)$ over the interval $[3k/8, k/2]$. We have that for $3k/8 \le j \le k/2$ and $k \ge 20$,

$$\begin{aligned} f(j) &\le -p\left(\frac{3k}{8}\right)\left(\frac{5k}{8}\right) + (1-\delta)p(k-1)\left(1 + \log\left(\frac{k^2/4}{(1-\delta)(k-1)}\right)\right) + \frac{k}{2} + \frac{3k}{8}\log\left(\frac{8}{3}\right) \\ &\le -\frac{15}{64}pk^2 + p(k-1)\left(1 + (1-\delta)\log\left(\frac{k^2/4}{k-1}\right)\right) + \frac{22}{25}k \\ &\le -pk\left(\frac{41}{256}k - 1 - \log\left(\frac{k^2/4}{k-1}\right)\right) - k\left(\frac{19}{256}pk - \frac{22}{25}\right) \\ &\le -\frac{1}{2}pk \le -(2+\epsilon/2)\log k. \end{aligned}$$

In the above, the first inequality follows from the fact that the term $j\log(k/j)$ is decreasing over the interval $[3k/8, k/2]$, the second inequality follows from the numeric inequality $(1-\delta) - (1-\delta)\log(1-\delta) \le 1$ for $\delta \in (0,1)$ which follows from the fact that $\log x \ge 1 - 1/x$ for $x > 0$, the forth inequality follows from $k \ge 20$.

Therefore, the exponent in (6.3) satisfies $f(j) \le -(1+\epsilon/2)\log k + 1$ for all $1 \le j \le k/2$ and $k \ge 20$. Finally, apply a union bound we get that

$$\begin{aligned} &\mathbb{P}(|\partial_K(C)| \le (1-\delta)p(k-1), \forall C \subset K \text{ s.t. } 1 \le |C| \le k-1) \\ &= \sum_{j=1}^{\lfloor k/2 \rfloor} \mathbb{P}(|\partial_K(C)| \le (1-\delta)p(k-1), \forall C \subset K \text{ s.t. } |C| = j) \\ &\le \exp(f(j) + \log k) \le \exp\left(-\frac{\epsilon}{2}\log k + 1\right) = ek^{-\epsilon/2} \end{aligned}$$

which proves the required result. □

Lemma 6.11 (Upper bound of external cut)**.** *For any* $0 < \delta \leq 1$ *and* $\epsilon > 0$ *with probability at least* $1 - k^{-\epsilon/3}$ *we have that* $|\partial_G(K)| \leq (1+\delta)qk(n-k) + (e\epsilon/\delta^2 + \epsilon/3)\log k$.

Proof. Note that $|\partial_G(K)|$ is the sum of $k(n-k)$ independent Bernoulli random variables with mean $\mathbb{E}[|\partial_G(K)|] = qk(n-k)$. We consider two cases depending on the value of $qk(n-k)$. If $qk(n-k) \geq \epsilon \log k/\delta^2$, then by the multiplicative Chernoff bound we have that,

$$\mathbb{P}(|\partial_G(K)| \geq (1+\delta)qk(n-k)) \leq \exp\left(-\frac{\delta^2}{3}qk(n-k)\right) \leq \exp\left(-\epsilon \log k/3\right). \tag{6.6}$$

Next consider the case $qk(n-k) \leq \epsilon \log k/\delta^2$. Denote $c(\epsilon,\delta) := e\epsilon/\delta^2 + \epsilon/3$ and observe that

$$\frac{\epsilon}{\delta^2} = \frac{c(\epsilon,\delta) - \epsilon/3}{e} = \left(1 - \frac{\epsilon/3}{c(\epsilon,\delta)}\right)\frac{c(\epsilon,\delta)}{e} \leq \exp\left(-\frac{\epsilon/3}{c(\epsilon,\delta)}\right)\frac{c(\epsilon,\delta)}{e}.$$

This means that

$$qk(n-k) \leq \frac{\epsilon}{\delta^2}\log k \leq \exp\left(-\frac{\epsilon/3}{c(\epsilon,\delta)} - 1\right)c(\epsilon,\delta)\log k,$$

and thus

$$\frac{qk(n-k)}{c(\epsilon,\delta)\log k} \leq \exp\left(-\frac{\epsilon/3}{c(\epsilon,\delta)} - 1\right) \iff c(\epsilon,\delta) + c(\epsilon,\delta)\log\left(\frac{qk(n-k)}{c(\epsilon,\delta)\log k}\right) \leq -\epsilon/3.$$

Therefore the Chernoff bound yields

$$\begin{aligned}
&\mathbb{P}\left(|\partial_G(K)| \geq c(\epsilon,\delta)\log k\right) \\
&\leq e^{-qk(n-k)}\left(\frac{eqk(n-k)}{c(\epsilon,\delta)\log k}\right)^{c(\epsilon,\delta)\log k} \\
&= \exp\left(-qk(n-k) + c(\epsilon,\delta)\log k\left(1 + \log\left(\frac{qk(n-k)}{c(\epsilon,\delta)\log k}\right)\right)\right) \\
&\leq \exp\left(\log k\left(c(\epsilon,\delta) + c(\epsilon,\delta)\log\left(\frac{qk(n-k)}{c(\epsilon,\delta)\log k}\right)\right)\right) \\
&\leq \exp(-\epsilon\log k/3).
\end{aligned} \tag{6.7}$$

Combining (6.6) and (6.7) gives the required result. □

Lemma 6.12 (Concentration of degrees). *If $p \geq \frac{(3+\epsilon)}{\delta^2}\frac{\log k}{k-1}$ for some $\epsilon > 0$ and $0 < \delta \leq 1$, then with probability at least $1 - 2k^{-\epsilon/3}$ we have that*

$$(1-\delta)p(k-1) \leq \deg_K^\circ(i) \leq (1+\delta)p(k-1), \forall i \in K.$$

Similarly, with probability at least $1 - 2k^{-\epsilon/3}$ we have that

$$(1-\delta)(p(k-1)+q(n-k)) \leq \deg_G^\circ(i) \leq (1+\delta)(p(k-1)+q(n-k)), \forall i \in K.$$

Proof. For each node $i \in K$, $\deg_K^\circ(i)$ is the sum of independent Bernoulli random variables with mean $\mathbb{E}[\deg_K^\circ(i)] = p(k-1)$, therefore, apply the multiplicative Chernoff bound we have

$$\mathbb{P}(|\deg_K^\circ(i) - p(k-1)| \geq \delta p(k-1)) \leq 2\exp(-\delta^2 p(k-1)/3) \leq 2\exp(-(1+\epsilon)\log k/3).$$

By taking a union bound over all $i \in K$ we obtain the required concentration result for $\deg_K^\circ(i)$ for all $i \in K$. The result for $\deg_G^\circ(i)$ for all $i \in K$ is obtained similarly. □

Lemma 6.13 (Well-connected cluster). *If $p \geq \max(\frac{(3+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-1}, \frac{(2+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-1}})$, then with probability at least $1 - 2k^{-\epsilon_1/3} - k^{-2\epsilon_2}$ we have that for all $s \in K$, for all $i \in K\backslash\{s\}$, there are at least $(1-\delta_1)(1-\delta_2)p^2(k-1)$ paths connecting node i to node s such that, the path lengths are at most 2 and the paths are mutually non-overlapping, i.e., an edge appears in at most one of the paths.*

Proof. Let $s \in K$ and denote F the set of neighbors of s in K. By Lemma 6.12 and our assumption on p we know that $|F| \geq (1-\delta_1)p(k-1)$ with probability at least $1 - 2k^{-\epsilon_1/3}$. Let us denote $E(A,B)$ the set of edges between $A \subseteq K$ and $B \subseteq K$. Let $i \in K\backslash\{s\}$. If $i \notin F$, then $|E(\{i\},F)|$ is the sum of independent Bernoulli random variables with mean $\mathbb{E}[|E(\{i\},F)|] = |F|p$. Apply the multiplicative Chernoff bound we get that

$$\begin{aligned}\mathbb{P}(|E(\{i\},F)| \leq (1-\delta_2)|F|p) &\leq \exp\left(-\frac{\delta_2^2}{2}|F|p\right) \\ &\leq \exp\left(-\frac{\delta_2^2(1-\delta_1)}{2}p^2(k-1)\right) \leq \exp(-(2+2\epsilon_2)\log k)\end{aligned}$$

where the last inequality is due to our assumption that $p \geq \frac{(2+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-1}}$. If $i \in F$, then the edge (i,s) is a path of length 1 between node i and node s, moreover,

$$\mathbb{P}(|E(\{i\},F\backslash\{i\})| + 1 \leq (1-\delta_2)|F|p) \leq \mathbb{P}(|E(i',F)| \leq (1-\delta_2)|F|p)$$

for any node $i' \in K\backslash F$ and $i' \neq s$. Note that, for $i \in K\backslash\{s\}$, each edge (i,j) in $E(\{i\}, F\backslash\{i\})$ identifies a unique path (i,j,s) and all these paths do not have overlapping edges. Therefore, denote $P(i,s)$ the set of mutually non-overlapping paths of length at most 2 between i and s. and take union bounds over all $i \in K\backslash\{s\}$ and then over all $s \in K$, we get that

$$\mathbb{P}(P(i,s) \leq (1-\delta_2)|F|p, \forall s \in K, \forall i \in K\backslash\{s\}) \leq k^{-2\epsilon_2}.$$

Finally, a union bound over the above event and the event that $|F| \leq (1-\delta_1)p(k-1)$ gives the required result. □

6.4.2 Proof of Lemma 6.7

We have that

$$\|X_i - X_j\|_2^2 = \begin{cases} \|Z_i - Z_j\|_2^2, & \text{if } i,j \in K, \\ \|Z_i - Z_j\|_2^2 + \|\mu_i - \mu_j\|_2^2 + (\mu_i - \mu_j)^T(Z_i - Z_j), & \text{if } i \in K, j \notin K. \end{cases} \tag{6.8}$$

Consider the random variable

$$\|Z_i - Z_j\|_2^2 - \mathbb{E}[\|Z_i - Z_j\|_2^2] = \sum_{\ell=1}^{r} \Big((Z_{i\ell} - Z_{j\ell})^2 - \mathbb{E}[(Z_{i\ell} - Z_{j\ell})^2] \Big).$$

Each term in the summation is sub-exponential and satisfies

$$\begin{aligned} \|(Z_{i\ell} - Z_{j\ell})^2 - \mathbb{E}[(Z_{i\ell} - Z_{j\ell})^2]\|_{\psi_1} &\leq C\|(Z_{i\ell} - Z_{j\ell})^2\|_{\psi_1} \\ &= C\|Z_{i\ell} - Z_{j\ell}\|_{\psi_2}^2 \leq 2C\|Z_{i\ell}\|_{\psi_2}^2 \leq C'\sigma_\ell^2 \end{aligned}$$

for some absolute constants C, C', where $\|\cdot\|_{\psi_1}$ and $\|\cdot\|_{\psi_2}$ denote the sub-exponential norm and the sub-Gaussian norm, respectively [123]. The first inequality follows from standard centering inequality for the sub-exponential norm (e.g. see Lemma 2.6.8 and Exercise 2.7.10 in [123]), and the second equality follows from Lemma 2.7.6 in [123]. Therefore, we may apply a Bernstein-type inequality for the sum of sub-exponential random variables (e.g. see Theorem 2.8.1 in [123]) and get

$$\begin{aligned} &\mathbb{P}\left(\Big| \|Z_i - Z_j\|_2^2 - \mathbb{E}\|Z_i - Z_j\|_2^2 \Big| > t \right) \\ \leq\ &\exp\left(-\min\left\{ \frac{t^2}{c\sum_{\ell=1}^{d} \|(Z_{i\ell} - Z_{j\ell})^2 - \mathbb{E}[(Z_{i\ell} - Z_{j\ell})^2]\|_{\psi_1}^2}, \right. \right. \end{aligned}$$

$$\left.\left.\frac{t}{c' \max_\ell \|(Z_{i\ell} - Z_{j\ell})^2 - \mathbb{E}[(Z_{i\ell} - Z_{j\ell})^2]\|_{\psi_1}}\right\}\right)$$

$$= \exp\left(-\min\left\{\frac{t^2}{c' \sum_{\ell=1}^d \sigma_\ell^4}, \frac{t}{c''\hat{\sigma}^2}\right\}\right)$$

for some absolute constants c, c'. Set $t = c''\hat{\sigma}^2 \log n$ for a large enough constant c'', use

$$\sum_{\ell=1}^r (\sigma_\ell/\hat{\sigma})^4 \le \sum_{\ell=1}^r (\sigma_\ell/\hat{\sigma})^2 = O(\log n)$$

which follows from Assumption 6.6, and take a union bound over all $i, j \in V$, we get that with probability at least $1 - o_n(1)$, for all $i, j \in V$ it holds that

$$\begin{aligned}
\|Z_i - Z_j\|_2^2 &\le \mathbb{E}\|Z_i - Z_j\|_2^2 + O(\hat{\sigma}^2 \log n) \\
&\le \tilde{c} \sum_{\ell=1}^r \|Z_{i\ell} - Z_{j\ell}\|_{\psi_2}^2 + O(\hat{\sigma}^2 \log n) \\
&\le \tilde{c}' \sum_{\ell=1}^r \sigma_\ell^2 + O(\hat{\sigma}^2 \log n) \\
&= O(\hat{\sigma}^2 \log n),
\end{aligned} \tag{6.9}$$

where $\tilde{c}, \tilde{c}'$ are absolute constants.

For $i \in K$ and $j \notin K$, the term $(\mu_i - \mu_j)^T(Z_i - Z_j) = \sum_{\ell=1}^d (\mu_{i\ell} - \mu_{j\ell})(Z_{i\ell} - Z_{j\ell})$ is a sum of independent and mean zero sub-Gaussian random variables. We may apply a general Hoeffding's inequality (see Lemma 2.6.3 in [123]) and get that

$$\begin{aligned}
\mathbb{P}(|(\mu_i - \mu_j)^T(Z_i - Z_j)| \ge t) &\le 2\exp\left(\frac{ct^2}{\max_\ell \|Z_{i\ell} - Z_{j\ell}\|_{\psi_2}^2 \|\mu_i - \mu_j\|_2^2}\right) \\
&\le 2\exp\left(-\frac{c't^2}{\hat{\sigma}^2 \|\mu_i - \mu_j\|_2^2}\right),
\end{aligned}$$

and hence by setting $t = c''\hat{\sigma}\sqrt{\log n}\|\mu_i - \mu_j\|_2$ for a large enough constant c'' we get that with probability at least $1 - o_n(1)$,

$$|(\mu_i - \mu_j)^T(Z_i - Z_j)| \le O(\hat{\sigma}\sqrt{\log n}\|\mu_i - \mu_j\|_2), \ \forall i \in K, j \notin K. \tag{6.10}$$

Combining (6.8), (6.9), (6.10), and using $\|\mu_i - \mu_j\|_2 \geq \hat{\mu} = \omega(\hat{\sigma}\sqrt{\log n})$, we get that with probability at least $1 - o_n(1)$,

$$\begin{aligned}
&\|X_i - X_j\|_2^2 \leq O(\hat{\sigma}^2 \log n), \ \forall i \in K, \forall j \in K, \\
&\|X_i - X_j\|_2^2 \geq \|\mu_i - \mu_j\|_2^2 - O(\hat{\sigma}\sqrt{\log n}\|\mu_i - \mu_j\|_2) \\
&\qquad\qquad = \|\mu_i - \mu_j\|_2^2(1 - o_n(1)) \geq \hat{\mu}^2(1 - o_n(1)), \ \forall i \in K, \forall j \notin K.
\end{aligned}$$

By Assumption 6.6, we may pick γ that satisfies $\gamma\hat{\sigma}^2 = o(\log^{-1} n)$ and $\gamma\hat{\mu}^2 = \omega_n(\lambda)$, and for any such γ we have

$$\begin{aligned}
&\exp(-\gamma\|X_i - X_j\|_2^2) \geq \exp(-o_n(1)), \ \forall i \in K, \forall j \in K, \\
&\exp(-\gamma\|X_i - X_j\|_2^2) \leq \exp(-\gamma\hat{\mu}^2(1 - o_n(1))), \ \forall i \in K, \forall j \notin K,
\end{aligned}$$

as required. This proves Lemma 6.7.

6.4.3 Proof of Theorem 6.8

We start with part 1 of the theorem. Without loss of generality let us assume that the node indices are such that $K = \{1, 2, \ldots, k\}$ and that $x_1^* \geq x_2^* \geq \ldots \geq x_k^*$. In order to show that $K \subseteq \text{supp}(x^*)$, it suffices to show that $x_k^* > 0$. Assume for the sake of contradiction that $x_k^* = 0$. Note that since the initial source mass is $(1+\beta)\sum_{i \in K} T_i$, in an optimal flow routing, the amount of mass that flows over an edge $e = (i, j)$ cannot be greater than $(1+\beta)\sum_{i \in K} T_i$. Using the relationship between primal and dual optimal solutions, this means that $w_{ij}|x_i^* - x_j^*| = |f_e^*| \leq (1+\beta)\sum_{i' \in K} T_{i'}$ for all $i, j \in V$. Therefore we have that

$$x_1^* \leq \sum_{i=1}^{k-1} \frac{(1+\beta)\sum_{i' \in K} T_{i'}}{w_{i(i+1)}} + x_k^* = \sum_{i=1}^{k-1} \frac{(1+\beta)\sum_{i' \in K} T_{i'}}{w_{i(i+1)}}.$$

It then follows from Lemma 6.7 that with probability at least $1 - o_n(1)$,

$$x_1^* \leq (1+\beta)k(1 + o_n(1)) \sum_{i \in K} T_i.$$

On the other hand, the total amount of mass that leaves K is

$$\sum_{i=1}^{k} \sum_{\substack{j \geq k+1 \\ j \sim i}} w_{ij}(x_i^* - x_j^*) \leq \sum_{i=1}^{k} x_i^* \sum_{\substack{j \geq k+1 \\ j \sim i}} w_{ij} \leq x_1^* \sum_{(i,j) \in \partial_G(K)} w_{ij}.$$

Apply Lemma 6.7 and Lemma 6.11 and pick $\epsilon = \delta = 1$ there, and use the above bound on x_1^*, we get that, with probability at least $1 - o_n(1) - k^{-1/3}$,

$$\sum_{i=1}^{k} \sum_{\substack{j \geq k+1 \\ j \sim i}} w_{ij}(x_i^* - x_j^*)$$
$$\leq (1+\beta)k^2(1+o_n(1))(2q(n-k) + 4\log k/k)\exp(-\gamma\hat{\mu}^2(1-o_n(1)))\sum_{i\in K} T_i.$$

Since we started with $(1+\beta)\sum_{i\in K} T_i$ initial mass inside K, nodes in K can settle at most $\sum_{i\in K} T_i$ units of mass, we know that at least $\beta\sum_{i\in K} T_i$ amount of mass must leave K. In what follows we show that this cannot be the case for appropriately chosen γ, and hence arriving at the desired contradiction. Since $\hat{\mu} = \omega(\hat{\sigma}\sqrt{\log n(1+\lambda)})$, we may pick γ such that $\gamma\hat{\sigma}^2 = o(\log^{-1} n)$ to satisfy the assumption required for Lemma 6.7, and at the same time $\gamma\hat{\mu}^2 = \omega(1+\lambda)$. Since $\lambda = \Omega(\log k + \log(q(n-k)) + \log(1/\beta))$, we know that for any terms $a_n = o_n(1)$ and $b_n = o_n(1)$ and for sufficiently large n,

$$\gamma\hat{\mu}^2(1-a_n) > 2\log k + \log(2q(n-k) + 4\log k/k) + \log(1/\beta + 1) + \log(1+b_n),$$

which implies that, for sufficiently large n,

$$(1+\beta)k^2(1+o_n(1))(2q(n-k) + 4\log k/k)\exp(-\gamma\hat{\mu}^2(1-o_n(1))) < \beta,$$

and hence

$$\sum_{i=1}^{k} \sum_{\substack{j \geq k+1 \\ j \sim i}} w_{ij}(x_i^* - x_j^*) < \beta\sum_{i\in K} T_i,$$

which is the desired contradiction. Therefore we must have that $x_k^* > 0$ and consequently $K \subseteq \mathrm{supp}(x^*)$. Now, since $x_i^* > 0$ for all $i \in K$, this means that nodes inside K settles exactly $\sum_{i\in K} T_i$ units mass, and hence exactly $\beta\sum_{i\in K} T_i$ mass leaves K. Because $x_i^* > 0$ only if node j is saturated with T_i unit mass, we get that $\sum_{i\in \mathrm{supp}(x_i^*)\setminus K} T_i \leq \beta\sum_{i\in K} T_i$.

Part 2 of the theorem is prove by following the same reasoning. Assume for the sake of contradiction that $x_k^* = 0$. Since $p \geq \frac{(4+\epsilon)}{\delta^2}\frac{\log k}{k-1}$, we apply Lemma 6.10 and get that with probability at least $1 - ek^{-\epsilon/2}$, $|\partial_K(C)| \geq (1-\delta)p(k-1)$ for every $C \subseteq K$ such that $1 \leq |C| \leq k-1$. We will assume that this event holds. Moreover, for any $1 \leq i \leq k-1$, the total amount of mass that moves from $\{1, 2, \ldots, i\}$ to $\{i+1, i+2, \ldots, k\}$ cannot be greater than $(1+\beta)\sum_{i\in K} T_i$. Since there are at least $(1-\delta)p(k-1)$ edges between $\{1, 2, \ldots, i\}$ and $\{i+1, i+2, \ldots, k\}$, we must have that

$$x_i^* - x_{i+1}^* \leq \frac{(1+\beta)\sum_{i'\in K} T_{i'}}{(1-\delta)p(k-1)\min_{j,j'\in K, j\sim j'} w_{jj'}}, \forall i = 1, 2, \ldots, k-1,$$

because, otherwise, there would be more than $(1+\beta)\sum_{i\in K} T_i$ mass that moves from $\{1,2,\ldots,i\}$ to $\{i+1,i+2,\ldots,k\}$. Apply Lemma 6.7 we have that, with probability at least $1-o_n(1)-ek^{-\epsilon/2}$,

$$x_1^* \le \sum_{i=1}^{k-1} \frac{(1+\beta)\sum_{i'\in K} T_{i'}}{(1-\delta)p(k-1)\min_{j,j'\in K, j\sim j'} w_{jj'}} \le \frac{(1+\beta)k(1+o_n(1))\sum_{i'\in K} T_{i'}}{(1-\delta)p(k-1)}.$$

The rest of the proof proceeds as the proof of part 1.

6.4.4 Proof of Theorem 6.9

To see that $K \subseteq \mathrm{supp}(x^*)$, let us assume for the sake of contradiction that $x_i^* = 0$ for some $i \in K$. This means that node i receives at most $T_i \le T_{\max}$ mass, because otherwise we would have $x_i^* > 0$. We also know that $i \neq s$ because $T_{\max} < \Delta_s$. Denote $F := \{j \in K : j \sim s\}$. We will consider two cases depending on if $i \in F$ or not. If $i \in F$, then we must have that, with probability at least $1-o_n(1)$,

$$w_{is}(x_s^* - x_i^*) \le T_{\max} \iff x_s^* \le T_{\max}/w_{is} + x_i^* = T_{\max}(1+a_n)$$

for some $a_n = o_n(1)$, where the last equality follows Lemma 6.7. Moreover, since $c_2 < 1$ we have that

$$\frac{p(k-1)}{\eta(c_2)} = p(k-1) + q(n-k)e^{-c_2\gamma\hat{\mu}^2} > p(k-1) + q(n-k)e^{-\gamma\hat{\mu}^2(1-b_n)} \tag{6.11}$$

for any $b_n = o_n(1)$ and for all sufficiently large n. Therefore, with probability at least $1-o_n(1)-4k^{-\epsilon_1/3}$ and for all sufficiently large n, the total amount of mass that is sent out from node s is

$$\begin{aligned}
\sum_{\ell\sim s} w_{is}(x_s^* - x_\ell^*) &= \sum_{\substack{\ell\sim s\\ \ell\in K}} w_{is}(x_s^* - x_\ell^*) + \sum_{\substack{\ell\sim s\\ \ell\notin K}} w_{is}(x_s^* - x_\ell^*) \\
&\overset{\text{(i)}}{\le} \sum_{\substack{\ell\sim s\\ \ell\in K}} x_s^* + \sum_{\substack{\ell\sim s\\ \ell\notin K}} e^{-\gamma\hat{\mu}^2(1-b_n)} x_s^* \qquad \text{for some } b_n = o_n(1) \\
&\overset{\text{(ii)}}{\le} (1+\delta_1)p(k-1)x_s^* + ((1+\delta_1)q(n-k) + 2\delta_1 p(k-1))e^{-\gamma\hat{\mu}^2(1-b_n)} x_s^* \\
&\le (1+3\delta_1)\left(p(k-1) + q(n-k)e^{-\gamma\hat{\mu}^2(1-b_n)}\right) x_s^*
\end{aligned}$$

$$\overset{\text{(iii)}}{<} (1+3\delta_1)\frac{p(k-1)}{\eta(c_2)}x_s^*$$
$$\leq (1+3\delta_1)\frac{p(k-1)}{\eta(c_2)}T_{\max}(1+a_n)$$
$$\overset{\text{(iv)}}{<} c_1(1+3\delta_1)\frac{p(k-1)}{\eta(c_2)}T_{\max},$$

where (i) follows from Lemma 6.7 and $x^* \geq 0$, (ii) follows from Lemma 6.12, (iii) follows from (6.11), (iv) follows from the assumption that $c_1 > 1$ and hence for all sufficiently large n we have $c_1 \geq (1+a_n)$ where $a_n = o_n(1)$. Since the initial mass equals the sum of T_s and the total amount of mass that is sent out from s, we get that the total amount of initial mass is

$$\Delta_s < c_1(1+3\delta_1)\frac{p(k-1)}{\eta(c_2)}T_{\max} + T_{\max} < c_1 T_{\max}\left(\frac{(1+3\delta_1)(1+\frac{1}{k-1})k}{\eta(c_2)}\right)$$
$$< c_1 T_{\max}\frac{m(\delta_1,\delta_2)k}{\eta(c_2)^2} = \Delta_s,$$

which is a contradiction. Therefore, we must have $i \notin F$.

Suppose now that $i \notin F$. Then we know that node i receives at most $T_i \leq T_{\max}$ mass from its neighbors. In particular, node i receives at most $T_{\max}$ mass from nodes in F, that is, $\sum_{\substack{j\in F\\ j\sim i}} w_{ij}x_j^* \leq T_{\max}$. By Lemma 6.13, we know that with probability at least $1 - 2k^{-\epsilon_1/3} - k^{-2\epsilon_2}$, node i has at least $(1-\delta_1)(1-\delta_2)p^2(k-1)$ neighbors in F. Apply Lemma 6.7 we get that, with probability at least $1 - o_n(1) - 2k^{-\epsilon_1/3} - k^{-2\epsilon_2}$,

$$\sum_{\substack{j\in F\\ j\sim i}} w_{ij}x_j^* \leq T_i \implies (1-\delta_1)(1-\delta_2)p^2(k-1)\cdot\min_{\substack{j\in F\\ j\sim i}} x_j^* \leq T_{\max}\cdot\max_{\substack{j\in F\\ j\sim i}}\frac{1}{w_{ij}}$$
$$\implies \min_{\substack{j\in F\\ j\sim i}} \leq \frac{T_{\max}(1+a_n)}{(1-\delta_1)(1-\delta_2)p^2(k-1)}$$
$$\implies \min_{j\in F} \leq \frac{T_{\max}(1+a_n)}{(1-\delta_1)(1-\delta_2)p^2(k-1)}$$

for some $a_n = o_n(1)$. Let $j \in F$ a node such that $x_j^* \leq x_\ell^*$ for all $\ell \in F$, then

$$x_j^* \leq \frac{T_{\max}(1+a_n)}{(1-\delta_1)(1-\delta_2)p^2(k-1)}. \tag{6.12}$$

By Lemma 6.13, with probability at least $1 - 2k^{-\epsilon_1/3} - k^{-2\epsilon_2}$, node j has at least $(1-\delta_1)(1-\delta_2)p^2(k-1) - 1$ neighbors in F. Since $x_j^* \le x_\ell^*$ for all $\ell \in F$ and $x_j^* \le x_s^*$, we know that

$$|\{\ell \in K : x_\ell^* \ge x_j^*\}| \ge (1-\delta_1)(1-\delta_2)p^2(k-1) \tag{6.13}$$

Therefore, for all sufficiently large n, with probability at least $1 - o_n(1) - 4k^{-\epsilon_1/3} - k^{-2\epsilon_2}$, the maximum amount of mass that node j can send out is

$$\begin{aligned}
&\sum_{\ell\sim j} w_{j\ell}(x_j^* - x_\ell^*) \\
&= \sum_{\substack{\ell\sim j\\ \ell\in K}} w_{j\ell}(x_j^* - x_\ell^*) + \sum_{\substack{\ell\sim j\\ \ell\notin K}} w_{j\ell}(x_j^* - x_\ell^*) \\
&\overset{\text{(i)}}{\le} \sum_{\substack{\ell\sim j\\ \ell\in K}} w_{j\ell}(x_j^* - x_\ell^*) + \sum_{\substack{\ell\sim j\\ \ell\notin K}} e^{-\gamma\hat{\mu}^2(1-b_n)}(x_j^* - x_\ell^*) \qquad \text{for some } b_n = o_n(1) \\
&\overset{\text{(ii)}}{\le} \Big((1+\delta_1)p(k-1) - (1-\delta_1)(1-\delta_2)p^2(k-1)\Big)x_j^* \\
&\qquad + \Big((1+\delta_1)q(n-k) + 2\delta_1 p(k-1)\Big)e^{-\gamma\hat{\mu}^2(1-b_n)}x_j^* \\
&\le \left[(1+3\delta_1)\left(p(k-1) + q(n-k)e^{-\gamma\hat{\mu}^2(1-b_n)}\right) - (1-\delta_1)(1-\delta_2)p^2(k-1)\right]x_j^* \\
&\overset{\text{(iii)}}{\le} \left[(1+3\delta_1)\frac{p(k-1)}{\eta(c_2)} - (1-\delta_1)(1-\delta_2)p^2(k-1)\right]x_j^* \\
&\overset{\text{(iv)}}{\le} \left[(1+3\delta_1)\frac{p(k-1)}{\eta(c_2)} - (1-\delta_1)(1-\delta_2)p^2(k-1)\right]\frac{T_{\max}(1+a_n)}{(1-\delta_1)(1-\delta_2)p^2(k-1)} \\
&\le T_{\max}(1+a_n)\frac{(1+3\delta_1)}{(1-\delta_1)(1-\delta_2)}\frac{1}{p\eta(c_2)} - T_{\max},
\end{aligned}$$

where (i) follows from Lemma 6.7, (ii) follows from Lemma 6.12 and (6.13), (iii) follows from (6.11) and (iv) follows from (6.12). Now, since node j settles at most $T_j \le T_{\max}$ mass, the maximum amount of mass that node j receives is

$$T_{\max}(1+a_n)\frac{(1+3\delta_1)}{(1-\delta_1)(1-\delta_2)}\frac{1}{p\eta(c_2)} - T_{\max} + T_{\max} = T_{\max}(1+a_n)\frac{(1+3\delta_1)}{(1-\delta_1)(1-\delta_2)}\frac{1}{p\eta(c_2)}.$$

This means that

$$w_{js}(x_s^* - x_j^*) \le T_{\max}(1+a_n)\frac{(1+3\delta_1)}{(1-\delta_1)(1-\delta_2)}\frac{1}{p\eta(c_2)}$$

$$\implies x_s^* \leq \frac{T_{\max}(1+a_n')}{(1-\delta_1)(1-\delta_2)}\left(\frac{1}{p^2(k-1)} + \frac{(1+3\delta_1)}{p\eta(c_2)}\right)$$

for some $a_n' = o_n(1)$, where we have applied Lemma 6.7 for w_{js}. Apply the same reasoning as before, we get that with probability at least $1 - o_n(1) - 4k^{-\epsilon_1/3} - k^{-2\epsilon_2}$ for all sufficiently large n, the total amount of mass that is sent out from node s is

$$\begin{aligned}
\sum_{\ell \sim s} w_{is}(x_s^* - x_\ell^*) &< (1+3\delta_1)\frac{p(k-1)}{\eta(c_2)}x_s^* \\
&\leq \frac{T_{\max}(1+a_n')}{(1-\delta_1)(1-\delta_2)}\left(\frac{(1+3\delta_1)}{p\eta(c_2)} + \frac{(1+3\delta_1)^2(k-1)}{\eta(c_2)^2}\right) \\
&\leq c_1 T_{\max}\frac{(1+3\delta_1)}{(1-\delta_1)(1-\delta_2)}\frac{(1+3\delta_2+\frac{1}{p(k-1)})}{\eta(c_2)^2}(k-1) \\
&\leq c_1 T_{\max}\frac{m(\delta_1,\delta_2)(k-1)}{\eta(c_2)^2},
\end{aligned}$$

but then this means that the total amount of initial mass is

$$\Delta_s < c_1 T_{\max}\frac{m(\delta_1,\delta_2)(k-1)}{\eta(c_2)^2} + T_{\max} < c_1 T_{\max}\frac{m(\delta_1,\delta_2)k}{\eta(c_2)^2} = \Delta_s$$

which is a contradiction. Therefore we must have $i \notin K$, but then this contradicts our assumption that $i \in K$. Since our choice of $i, s \in K$ were arbitrary, this means that $x_i^* > 0$ for all $i \in K$ and for all $s \in K$.

Finally, the upper bound on the false positives follows directly from the fact that $x_i^* > 0$ only if node i is saturated with exactly T_i mass. When $T_i = 1$ for all i the result follows directly from $\Delta_s = c_1 m(\delta_1,\delta_2)k/\eta(c_2)^2$. When $T_i = \deg^\circ(i)$ for all i, we may apply Lemma 6.12 and get that

$$\Delta_s \leq \frac{c_1 m(\delta_1,\delta_2)}{\eta(c_2)^2}(1+\delta_1)k(p(k-1)+q(n-k)) \leq \frac{c_1 m(\delta_1,\delta_2)}{\eta(c_2)^2}\frac{(1+\delta_1)}{(1-\delta_1)}\mathrm{vol}^\circ(K)$$

from which the result follows.

Chapter 7

Weighted Diffusion with Node Labels

7.1 Diffusion with Noisy Node Labels: What and Why

The growing interest in machine learning problems over graphs with additional node information such as texts, images, or labels has popularized methods that require the costly operation of processing the entire graph. Yet, little effort has been made to the development of fast local methods (i.e. without accessing the entire graph) that extract useful information from such data. In Chapter 6 we introduced and analyzed the weighted flow diffusion algorithm for local graph clustering with node attributes. It is the first local diffusion method which takes into account node attributes in a surprisingly simple yet principled way. However, establishing a good statistical recovery guarantee for Algorithm 8 requires assuming that the underlying data model obeys a strong homophily assumption. In particular, Assumption 6.6 requires that the node attributes are linearly separable with high probability, i.e., there is $h \in \mathbb{R}^r$ and $c \in \mathbb{R}$ such that, with probability at least $1 - o_n(1)$, one has $h^T X_i < c$ if $i \in K$ and $h^T X_i > c$ if $i \notin K$, for all $i \in V$. Although it is reasonable to expect that some real-world datasets contain very informative node attributes, such an assumption does not necessarily hold in general. Furthermore, in many practical settings we also have access to other forms of node information, for example, the ground-truth labels of a small set of nodes which reveal their cluster affiliation, such as when it is known that certain nodes do not belong to certain community.

This chapter studies diffusion with noisy labels in the context of local clustering. The theoretical setting that we introduce below removes the homophily assumption (i.e. Assumption 6.6) required by Algorithm 8 from Chapter 6, and additionally allows for working

with other forms of node information such as the ground-truth node labelling information. Formally, we consider the following abstract setting:

Diffusion with noisy node labels: Assume existence of a target cluster K inside a given graph $G = (V, E)$. Initially, each node receives a binary label based on its affiliation with K: 1 if it belongs to K and 0 otherwise. Then, a fraction of node labels is flipped. Given a seed node from K, how and when can these noisy node labels be used in a diffusion method and help recover K more accurately?

One can view the noisy node labels as an abstract aggregation of all additional sources of information at hand. The level of label noise, i.e. the fraction of flipped labels, controls the quality of the additional data sources that we have access to. From a practical point of view, noisy labels may be seen as the output of an oracle which takes input raw information about a node and generates a prediction on whether or not that node should belong to the target cluster K. For example, in semi-supervised node classification settings where only a constant number ground-truth node labels is available, one can still train a classifier to classify the unlabelled nodes based on their available feature information (e.g. raw texts, images, or numerical embeddings which describe properties of the node). In such cases, neither training the classifier nor making inference on an unlabelled node requires accessing the graph structure, and hence obtaining and using the classifier as an oracle to generate noisy node labels is inherently a local operation. Therefore, understanding how diffusion can exploit noisy node labels and recover a target cluster more accurately, without having to explore the entire graph, is of particular practical interest.

In this chapter, we still focus on the weighted flow diffusion problem as formulated in Problem (6.1) and Problem (6.2). Again, we choose flow diffusion due to its good empirical performance and its flexibility at initializing a diffusion process. However, the results in this chapter with respect to the weighted flow diffusion process may be easily extended to other diffusion processes such that (truncated) random walks and Personalized Approximate PageRank (APPR). In Section 7.3 we empirically show that the incorporating noisy node labels in PageRank dramatically improves its local clustering performance over both synthetic and real-world graphs.

7.2 Local Clustering with Noisy Node Labels

In this section we discuss the problem of local graph clustering with noisy node labels. Given a graph $G = (V, E)$ and noisy node labels $\tilde{y}_i \in \{0, 1\}$ for $i \in V$, the goal is to identify an unknown target cluster K around a set of seed nodes. We will focus on using

the solution of Problem (6.2) for local clustering, but the method and results should easily extend to other diffusion methods such as PageRank (APPR), which we verify empirically in Section 7.3. Let x^* denote the optimal solution of Problem (6.2), we adopt the same rounding strategy as we did in Chapter 6. That is, we consider $\mathrm{supp}(x^*)$ as the output cluster and compare it against the target K. We will discuss specific diffusion setup with regard to the choice of source mass Δ and sink capacity T in Section 7.2.1. Occasionally we write $x^*(T, \Delta)$ or $x^*(\Delta)$ to emphasize its dependence on the input source mass and sink capacity, and we omit them when they are clear from the context. Whenever deemed necessary for the sake of clarity, we use different superscripts x^* and $x^\dagger$ to distinguish solutions of Problem (6.2) obtained under different edge weights.

We will denote

$$a_1 = |K \cap \tilde{Y}_1|/|K|, \quad a_0 = |K^{\mathsf{c}} \cap \tilde{Y}_0|/|K^{\mathsf{c}}|, \tag{7.1}$$

which quantify the accuracy of labels within K and outside K, respectively. If $a_0 = a_1 = 1$, then the labels perfectly align with cluster affiliation. We say that the labels are noisy if at least one of a_0 and a_1 is strictly less than 1. In this case, we are interested in how and when the labels can help diffusion obtain a more accurate recovery of the target cluster. In order for the labels to provide any useful information at all, we will assume that the accuracy of these labels is at least $1/2$.

Assumption 7.1. The label accuracy satisfies $a_0 \geq 1/2$ and $a_1 \geq 1/2$.

To exploit the information provided by noisy node labels, we consider a straightforward way to weight the edges of the graph $G = (V, E)$ based on the given labels. Denote $G^w = (V, E, w)$ the weighted graph obtained by assigning edge weights according to $w : E \to \mathbb{R}$. We set

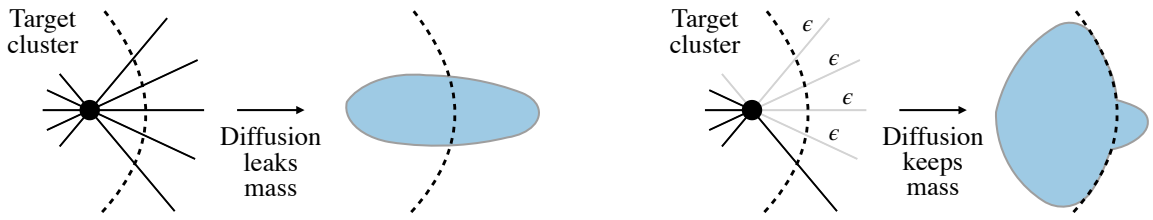
$$w(i,j) = 1 \ \text{ if } \ \tilde{y}_i = \tilde{y}_j, \quad \text{and} \quad w(i,j) = \epsilon \ \text{ if } \ \tilde{y}_i \neq \tilde{y}_j, \tag{7.2}$$

for some small $\epsilon \in [0, 1)$. This assigns a small weight over cross-label edges and maintains unit weight over same-label edges. The value of ϵ interpolates between two special scenarios. If $\epsilon = 1$ then G^w reduces to G, and if $\epsilon = 0$ then all edges between $\tilde{Y}_1$ and $\tilde{Y}_0$ are removed. In principle, the latter case can be very helpful if the labels are reasonably accurate or the target cluster is well-connected. For example, in Section 7.2.1 we will show that solving Problem (6.2) over the weighted graph G^w by setting $\epsilon = 0$ can lead to a more accurate recovery of the target cluster than solving it over the original graph G. In practice, one may also choose a small nonzero ϵ to improve robustness against very noisy labels. In our experiments, we find that the results are not sensitive to the choice of ϵ, as we obtain similar clustering accuracy for $\epsilon \in [10^{-2}, 10^{-1}]$ over different datasets with varying cluster size and label accuracy. We use the F1 score to measure the accuracy of cluster recovery.

Suppose that a local clustering algorithm returns a cluster $C \subset V$, recall that the F1 score is defined as

$$\mathrm{F1}(C) = \frac{|C \cap K|}{|C \cap K| + |C \backslash K|/2 + |K \backslash C|/2}.$$

Even though the edge weights defined in Equation (7.2) are fairly simple and straightforward, they lead to surprisingly good local clustering result over real-world data, as we will show in Section 7.3. Before we formally analyze diffusion over G^w, let us start with an informal and intuitive discussion on how such edge weights can be beneficial. Consider a step during a generic local diffusion process where mass is spread from a node within the target cluster to its neighbors. Suppose this node has a similar number of neighbors within and outside the target cluster. An illustrative example is shown in Figure 7.1a. In this case, since all edges are treated equally, diffusion will spread a lot of mass to the outside. This makes it very difficult to accurately identify the target cluster without suffering from excessive false positives and false negatives. On the other hand, if the labels have good initial accuracy, for example, if $a_1 > a_0$, then weighting the edges according to Equation (7.2) will generally make a lot of boundary edges smaller while not affecting as many internal edges. This is illustrated in Figure 7.1b. Since a diffusion step spreads mass proportionally to the edge weights (cf. Algorithm 7), a neighbor that is connected via a lower edge weight will receive less mass than a neighbor that is connected via a higher edge weight. Consequently, diffusion in such a weighted setting forces more mass to be spread within the target cluster, and hence less mass will leak out. This generally leads to a more accurate recovery of the target cluster.



(a) Diffusion in the original graph

(b) Diffusion in the weighted graph

Figure 7.1: Label-based edge weights such as those define in Equation (7.2) can avoid mass leakage by attenuating more boundary edges than internal edges. This helps local diffusion more accurately recover the target cluster.

7.2.1 Statistical Recovery in Random Graphs with Noisy Labels

Similar to our analysis in Section 7.2.1, we assume that the graph and the target cluster are generated from the following random model, which can be seen as a localized stochastic block model.

Definition 7.2. [Local random model] Given a set of nodes V and a target cluster $K \subset V$. For every pair of nodes i and j, if $i, j \in K$ then we draw an edge (i, j) with probability p; if $i \in K$ and $j \in K^{\mathsf{c}}$ then we draw an edge (i, j) with probability q; otherwise, if both $i, j \in K^{\mathsf{c}}$ then we allow any (deterministic or random) model to draw an edge.

Let $k = |K|$ and $n = |V|$. For a node $i \in K$, the expected number of internal connections is $p(k-1)$ and the expected number of external connections is $q(n-k)$. We will denote their ratio by

$$\gamma := \frac{p(k-1)}{q(n-k)}.$$

The value of γ can be seen as a measure of the structural signal of the target cluster K. When γ is small, a node within K is likely to have more external connections than internal connections; when γ is large, a node within K is likely to have more internal connections than external connections.

Recall our definition of the weighted graph G^w whose edge weights are given in Equation (7.2) based on node labels. We will study the effect of incorporating noisy labels by comparing the clustering accuracy obtained by the dual flow diffusion Problem (6.2) over the original graph G and the weighted graph G^w, respectively. For the purpose of the analysis, we simply set $\epsilon = 0$ as this is enough to demonstrate the advantage of diffusion over G^w. Determining an optimal and potentially nonzero $\epsilon \in [0, 1]$ that maximizes accuracy is an interesting question and we provide an informal discussion in Section 7.2.2. From the perspective of local clustering by minimizing conductance, if p and q are reasonably large, e.g., $p \geq 4 \log k/k$ and $q \geq 4 \log k/(n-k)$, then a simple computation invoking the Chernoff bound yields that

$$\begin{aligned} |\partial_{G^w}(K)| &\asymp (a_1(1-a_0) + a_0(1-a_1)) \cdot |\partial_G(K)|, \quad \text{with high probability,} \\ \mathrm{vol}_{G^w[K]}(K) &\asymp (a_1^2 + (1-a_1)^2) \cdot \mathrm{vol}_{G[K]}(K), \quad \text{with high probability,} \end{aligned}$$

where $G[S]$ denotes the subgraph induced on $S \subseteq V$, so $\mathrm{vol}_{G[K]}(K)$ measures the overall internal connectivity of K. Since $a_1(1-a_0) + a_0(1-a_1) < a_1^2 + (1-a_1)^2$ as long as $a_0, a_1 > 1/2$, the target cluster K will have a smaller conductance in G^w as long as the label accuracy is larger than $1/2$. As a result, this potentially improves the detectability of

K in G^w. Of course, a formal argument requires careful treatments of diffusion dynamics in G and G^w, respectively.

We consider local clustering with a single seed node $s \in K$ using the flow diffusion process where the sink capacity is set to $T(i) = 1$ for all $i \in V$. Although discussed earlier, we summarize below the steps of local clustering with noisy labels using flow diffusion in

Algorithm 9 Local Graph Clustering with Noisy Node Labels

Input: graph $G = (V, E)$, seed node $s \in V$, noisy labels $\tilde{y}_i \in \{0, 1\}$ for $i \in V$, source mass θ for some $\theta > 0$.

Output: a cluster $K' \subseteq V$.

1. Define weighted graph $G^w = (V, E, w)$ whose edge weights are given by $w(i, j) = 1$ if $\tilde{y}_i = \tilde{y}_j$ and 0 otherwise.
2. Set source mass $\Delta(i) = \theta$ if $i = s$ and 0 otherwise. Set sink capacity $T(i) = 1$ for all i.
3. Run Algorithm 7 with input G^w, Δ, T and obtain output x'.
4. Return $K' = \mathrm{supp}(x')$

In a practical implementation of Algorithm 9, Steps 1-3 can be carried out without accessing the full graph. As discussed earlier in Chapter 6, this is because computing the solution x^* only requires access to nodes (and their labels) that either belong to $\mathrm{supp}(x^*)$ or are neighbors of a node in $\mathrm{supp}(x^*)$. Recall that by Lemma 6.2 the total amount of source mass controls the size of $\mathrm{supp}(x^*)$. In the above setup, θ specifies the initial source mass at the seed node, and since $T(i) = 1$ for all i, we have $|\mathrm{supp}(x^*)| \leq \theta$. Applying Theorem 6.3, we get that an ϵ accurate solution can be computed in time $O(\mathrm{poly}(\theta) \log(1/\epsilon))$ (assuming that the number of neighbors each node has in the graph is bounded by $O(\mathrm{poly}(\theta))$). Here, we make the same assumption as stated in Assumption 6.4. That is, we assume that after τ iterations, if the dual variables x^τ generated by Algorithm 7 incur exponentially small error ϵ, then $\mathrm{supp}(x^\tau) = \mathrm{supp}(x^*)$. Under Assumption 6.4, Algorithm 9 returns $K' = \mathrm{supp}(x^*)$ in time $O(\mathrm{poly}(\theta))$.

Given source mass $\theta > 0$ at the seed node, let $x^*(\theta)$ and $x^\dagger(\theta)$ denote the solutions obtained from solving Problem (6.2) over G and G^w, respectively. Theorem 7.3 provides a lower bound on the F1 score obtained by $\mathrm{supp}(x^\dagger(\theta^\dagger))$ with appropriately chosen source mass $\theta^\dagger$. In addition, it gives a sufficient condition on the label accuracy a_0 and a_1 such that flow diffusion over G^w with source mass $\theta^\dagger$ at the seed node results in a more accurate recovery of K than flow diffusion over G with any possible choice of source mass. Theorem 7.3 involves many parameters and may be difficult to parse on the first pass. Therefore we take a special case and state a simplified version of Theorem 7.3 in Corollary 7.4 under

a slightly stronger assumption on the density of edges. We recommend the reader directly to Corollary 7.4 and come back to Theorem 7.3 for details and references.

Theorem 7.3. *Suppose that* $p \geq \max(\frac{(6+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-2}, \frac{(\sqrt{8}+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-2}})$ *and* $q \geq \frac{(3+\epsilon_3)}{\delta_3^2}\frac{\log k}{n-k}$ *for some* $\epsilon_1, \epsilon_2, \epsilon_3 > 0$ *and* $0 < \delta_1, \delta_2, \delta_3 \leq 1$. *Then with probability at least* $1 - 3k^{-\epsilon_1/6} - k^{-\epsilon_2} - k^{-\epsilon_3/3}$, *there is a set* $S \subseteq K$ *with cardinality at least* $|K|/2$ *and a choice of source mass* $\theta^\dagger$, *such that for every seed node* $s \in S$ *with source mass* $\theta^\dagger$ *at the seed node, we get*

$$\mathrm{F1}(\mathrm{supp}(x^\dagger(\theta^\dagger))) \geq \left[1 + \frac{1}{2}\left(\left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}}\right)^2 r - 1\right) + \frac{1-a_1}{2a_1}\right]^{-1}. \tag{7.3}$$

In this case, if the accuracy of noisy labels satisfies,

$$a_0 \geq 1 - \frac{(k-2)}{(k-1)}\left(\sqrt{\frac{p/\gamma}{r'} + \frac{2a_1 - 1}{ra_1}} - 1\right)a_1\gamma, \tag{7.4}$$

where

$$r = \frac{(1+\delta_1)(1+\delta_1+\frac{2}{p(k-1)}))}{(1-\delta_1)(1-\delta_2)} \quad \textit{and} \quad r' = \frac{r}{(1-\delta_3)}\frac{k}{(k-1)},$$

then we have $\mathrm{F1}(\mathrm{supp}(x^\dagger(\theta^\dagger))) > \max_{\theta\geq 0}\mathrm{F1}(\mathrm{supp}(x^*(\theta)))$.

Corollary 7.4 (Simplified version of Theorem 7.3)**.** *Suppose that* $p = \omega(\frac{\sqrt{\log k}}{\sqrt{k}})$ *and* $q = \omega(\frac{\log k}{n-k})$. *With probability at least* $1 - 1/k$, *there is a set* $S \subseteq K$ *with cardinality at least* $|K|/2$ *and a choice of source mass* $\theta^\dagger$, *such that for every seed node* $s \in S$ *we have*

$$\mathrm{F1}(\mathrm{supp}(x^\dagger(\theta^\dagger))) \geq \left[1 + \frac{(1-a_1)}{2a_1} + \frac{(1-a_0)}{\gamma a_1} + \frac{(1-a_0)^2}{2\gamma^2 a_1^2}\right]^{-1} - o_k(1). \tag{7.5}$$

Furthermore, if the accuracy of noisy labels satisfies

$$a_0 \geq 1 - \left(\sqrt{p/\gamma + 2 - 1/a_1} - 1\right)a_1\gamma + o_k(1), \tag{7.6}$$

then we have $\mathrm{F1}(\mathrm{supp}(x^\dagger(\theta^\dagger))) > \max_{\theta\geq 0}\mathrm{F1}(\mathrm{supp}(x^*(\theta)))$.

Corollary 7.4 requires additional discussion. First, the lower bound in (7.5) increases with a_0, a_1 and γ. This naturally corresponds to the intuition that, as the label accuracy a_0 and a_1 become larger, we can expect a more accurate recovery of K; while on the other

hand, as the local graph structure becomes noisy, i.e. as γ becomes smaller, it generally becomes more difficult to accurately recover K. Note that when $a_0 = a_1 = 1$, the labels perfectly align with cluster affiliation, and in this special case the lower bound on the F1 score naturally becomes $1 - o_k(1)$. This means that the solution $x^\dagger$ over the weighted graph G^w fully leverages the perfect label information. Finally, notice that in (7.5) the F1 is lower bounded by a constant as long as $\gamma = \Omega_n(1)$, even if a_0 is as low as $1/2$. In comparison, under a typical local clustering context where $k \ll n$, the F1 score obtained from directly using the noisy labels can be arbitrarily close to 0, i.e. we have $\mathrm{F1}(\tilde{Y}_1) \leq o_n(1)$, as long as a_0 is bounded away from 1. This demonstrates the importance of employing local diffusion. Even when the initial labels are deemed fairly accurate based on a_0 and a_1, e.g. $a_1 = 1$, $a_0 = 0.99$, the F1 score of the labels can still be very low due to $n - k \gg k$. In Section 7.3, over both synthetic and real-world data, we show empirically that flow diffusion over the weighted graph G^w can result in surprisingly better F1 even when the F1 of labels is very poor.

Second, if $a_1 = 1$ then (7.6) becomes

$$a_0 \geq 1 - \left(\sqrt{1 + p/\gamma} - 1\right)\gamma + o_k(1). \tag{7.7}$$

Observe that: (i) The function $\sqrt{\gamma^2 + p\gamma} - \gamma$ is increasing with γ, therefore the left hand side of (7.7) increases as γ decreases. This corresponds to the intuition that, as the external connectivity of the target cluster becomes larger (i.e. as γ decreases), we need more accurate labels to prevent a lot of mass from leaking out. (ii) When $q \geq \Omega(\frac{k}{n-k})$, we have $p/\gamma \geq \Omega(1)$, and (7.7) further simplifies to $a_0 \geq 1 - \Omega(\gamma)$. In this case, if γ is also constant, we can expect that flow diffusion over G^w to give a better result even if a constant fraction of labels is incorrect. Here, the required conditions on q and γ may look a bit strong because we did not assume anything about the graph structure outside K. One may expect to obtain much weaker conditions than (7.6) or (7.7) under additional assumptions on K^{c}. Finally, we leave the proof of Theorem 7.3 to Section 7.4.2.

7.2.2 Beyond the Trivial Edge Weight

In this section we informally discuss the choice of ϵ in Equation (7.2). Given a graph $G = (V, E)$ generated from the local random model in Definition 7.2, noisy labels $\tilde{y}_i$ for node $i \in V$, in Algorithm 9 we defined the edge weights in G^w such that $w(i, j) = 1$ if $\tilde{y}_i = \tilde{y}_j$ and $w(i, j) = \epsilon$ otherwise, where we set $\epsilon = 0$. While understanding diffusion in G^w with $\epsilon = 0$ already provides us with some insights with regard to how noisy labels can

be useful, a natural question is to ask for an "optimial" ϵ given model parameters n, k, p, q and label accuracy a_0, a_1.

To see why this is an interesting problem, consider the case when a_1 is low. In this case, if we set $\epsilon = 0$ and start diffusing mass from a seed node $s \in K$ with $\tilde{y}_s = 1$, then diffusion in G^w cannot reach a node $i \in K$ such that $\tilde{y}_i = 0$, because the graph G^w is disconnected with two components $\tilde{Y}_1 = \{i \in V : \tilde{y}_i = 1\}$ and $\tilde{Y}_0 = \{i \in V : \tilde{y}_i = 0\}$. Consequently, the recall (i.e. the accuracy metric) of the output cluster is at most a_1. On the other hand, if we instead set $\epsilon > 0$, this allows diffusion in G^w to reach node $i \in K$ whose label is $\tilde{y}_i = 0$, however, at the same time, diffusion in G^w incurs the risk of reaching a node $i \notin K$ whose label is $\tilde{y}_i = 0$. Therefore, setting $\epsilon > 0$ allows for discovering more true positives at the expense of incurring more false positives. Whether one should set $\epsilon = 0$ will depend on n, k, p, q, a_0, a_1 and the accuracy metric (e.g. precision, recall, or the F1) one aims to maximize. If the objective is to maximize recall, then clearly one should set $\epsilon > 0$ to allow recovering nodes in K that receive different noisy labels. In general, it turns out that rigorously characterizing an "optimal" ϵ that maximizes other accuracy metrics such as the F1 is nontrivial. In what follows we discuss intuitively the potential conditions under which one should set $\epsilon = 0$ or $\epsilon > 0$ to obtain a better clustering result which balances precision and recall, i.e. attains a higher F1. In Section 7.3.1 we empirically demonstrate these conditions over synthetic data.

Conjecture 1: *A sufficient condition to favor $\epsilon > 0$ is $(1 - a_1)pk > a_0q(n - k)$.*

Conjecture 2: *A sufficient condition to favor $\epsilon = 0$ is $(1 - a_1)p^2k < a_0q^2(n - k)$.*

We provide an informal explanation for these conditions. Note that if a seed node s is drawn uniformly at random from K, then with probability $a_1 \geq 1/2$ we get that $s \in K \cap \tilde{Y}_1$. Therefore let us assume that a seed node is selected from $K \cap \tilde{Y}_1$. In this case, since the diffusion of mass starts from within $K \cap \tilde{Y}_1$, excess mass needs to get out of $K \cap \tilde{Y}_1$ along the cut edges of $K \cap \tilde{Y}_1$. Let us focus on the edges between $K \cap \tilde{Y}_1$ and $\tilde{Y}_0$ since these are the edges affected by ϵ. The edges between $K \cap \tilde{Y}_1$ and $K^{\mathsf{c}} \cap \tilde{Y}_1$ are not affected by ϵ. In expectation, every node in $K \cap \tilde{Y}_1$ has $(1 - a_1)pk$ number of neighbors in $K \cap \tilde{Y}_0$ and $a_0q(n - k)$ number of neighbors in $K^{\mathsf{c}} \cap \tilde{Y}_0$. Therefore, in a diffusion step when we push excess mass from a node in $K \cap \tilde{Y}_1$ to its neighbors, for every $(1 - a_1)pk$ unit mass that is pushed into $K \cap \tilde{Y}_0$, on average $a_0q(n - k)$ unit mass is pushed into $K^{\mathsf{c}} \cap \tilde{Y}_0$. If $(1 - a_1)pk > a_0q(n - k)$, then this means that $K \cap \tilde{Y}_0$ receives more mass than $K^{\mathsf{c}} \cap \tilde{Y}_0$. Consequently, as a result of $K \cap \tilde{Y}_0$ receiving more mass than $K^{\mathsf{c}} \cap \tilde{Y}_0$ from a diffusion step within $K \cap \tilde{Y}_1$, we can expect that by setting $\epsilon > 0$, we obtain more true positives as the diffusion process covers nodes in $K \cap \tilde{Y}_0$ at the expense of fewer false positives as the diffusion process covers less nodes in $K^{\mathsf{c}} \cap \tilde{Y}_0$. This leads to our first conjecture on the

condition to favor $\epsilon > 0$ over $\epsilon = 0$.

For the condition in our second conjecture, note that even when $K \cap \tilde{Y}_0$ receives less mass from $K \cap \tilde{Y}_1$ than the amount of mass that $K^{\mathsf{c}} \cap \tilde{Y}_0$ receives from $K \cap \tilde{Y}_1$, it does not necessarily imply that we would get more number of false negatives from $K^{\mathsf{c}} \cap \tilde{Y}_0$ and fewer number of true positives from $K \cap \tilde{Y}_0$. Recall that we use $\mathrm{supp}(x^*)$ as the output cluster where x^* is the optimal solution of Problem (6.2). For a node $i \in V$, we know that $x_i^* > 0$ only if node i receives more than 1 unit mass. Consider the following two average diffusion dynamics. First, as discussed before, for every $(1-a_1)pk$ unit mass that is pushed into $K \cap \tilde{Y}_0$ from the a_1pk nodes in $K \cap \tilde{Y}_1$, on average (i.e. averaged over multiple nodes) $a_0q(n-k)$ unit mass is pushed into $K^{\mathsf{c}} \cap \tilde{Y}_0$. Second, in expectation, every node in $K \cap \tilde{Y}_0$ has a_1pk neighbors in $K \cap \tilde{Y}_1$ and every node in $K^{\mathsf{c}} \cap \tilde{Y}_0$ has a_1qk neighbors in $K \cap \tilde{Y}_1$. For every unit mass that moves from $K \cap \tilde{Y}_1$ to $K \cap \tilde{Y}_0$, a node in $K \cap \tilde{Y}_0$ on average (i.e. averaged over multiple nodes and multiple diffusion steps) receives $pa_1k/a_1k = p$ unit and a node in $K^{\mathsf{c}} \cap \tilde{Y}_0$ on average receives $qa_1k/a_1k = q$ unit. Combining the above two points, we get that on average, for every $(1-a_1)p^2k$ unit mass received by a node in $K \cap \tilde{Y}_0$, a node in $K^{\mathsf{c}} \cap \tilde{Y}_0$ receives $a_0q^2(n-k)$ unit mass. Therefore, if $(1-a_1)p^2k < a_0q^2(n-k)$, then setting $\epsilon > 0$ would make a node $i \in K^{\mathsf{c}} \cap \tilde{Y}_0$ generally receive less mass than a node in $j \in K^{\mathsf{c}} \cap \tilde{Y}_0$. Consequently, a node $j \in K^{\mathsf{c}} \cap \tilde{Y}_0$ is more likely to receive more than 1 unit mass. This implies that, by setting $\epsilon > 0$ we would get fewer number of true positives from $K \cap \tilde{Y}_0$ at the expense of incurring more number of false positives from $K^{\mathsf{c}} \cap \tilde{Y}_0$. This leads to our second conjecture.

Of course, a rigorous argument to justify both conjectures will require a much more careful analysis of the diffusion dynamics and additional assumptions on p, q so that the average behaviors described in the above hold with high probability. In addition, there is a gap of order p/q between the two conditions. It is an open question to determine a good strategy to set ϵ in that "gap regime".

7.3 Empirical Results

In this section, we evaluate the effectiveness of employing flow diffusion over the label-weighted graph G^w whose edge weights are given in Equation (7.2) for local clustering. That is, we evaluate Algorithm 9 for local clustering. We will refer to Algorithm 9 as **Label-based Flow Diffusion (LFD)** due to the fact that it employs flow diffusion in G^w whose edges are weighted according to node labels. We compare the results with employing flow diffusion directly on the original unweighted graph G, which we will abbreviate as **FD**. Whenever a dataset includes node attributes, we also compare with Algorithm 8

which uses the Gaussian kernel of node attributes to define edge weights. We will refer to Algorithm 8 as **WFD**. Additionally, we carry out experiments comparing **Label-based PageRank (LPR)** on the weighted graph G^w with vanilla **PageRank (PR)** on the original graph G. The results are similar: Diffusion over the label-weighted graph G^w consistently outperforms diffusion over the original graph G.

We use both synthetic and real-world data to evaluate the methods. The synthetic data is used to demonstrate our theory and show how local clustering performance improves as label accuracy increases in a controlled environment. For the real-world data, we consider both supervised (i.e. we have access to both node attributes and some ground-truth labels) and unsupervised (i.e. we have access to only node attributes) settings. We show that in both settings, one may easily obtain reasonably good node labels such that, leveraging these labels via diffusion over G^w leads to consistently better results across all 6 datasets, improving the F1 score by up to 13%.

7.3.1 Experiments on Synthetic Graphs

We generate a synthetic graph using the stochastic block model with cluster size $k = 500$ and number of clusters $c = 20$. The number of nodes in the graph equals $n = kc = 10,000$. Two nodes within the same cluster are connected with probability $p = 0.05$, and two nodes from different clusters are connected with probability $q = 0.025$. For edge weights in G^w we set $\epsilon = 0.05$, see Equation (7.2). We vary the label accuracy a_0 and a_1 as defined in Equation (7.1) to demonstrate the effects of varying label noise. We consider 3 settings: (i) fix $a_0 = 0.7$ and vary $a_1 \in [1/2, 1)$; (ii) fix $a_1 = 0.7$ and vary $a_0 \in [1/2, 1)$; (iii) vary both $a_0, a_1 \in [1/2, 1)$ at the same time. For each pair of (a_0, a_1), we run 100 trials. For each trial, we randomly select one of the 20 ground-truth clusters as the target cluster. Then we generate noisy labels according to a_0 and a_1. For each trial, we randomly select a node from the target cluster as the seed node. We set the sink capacity $T(i) = 1$ for all nodes. For the source mass at the seed node, we set it to αk for $\alpha = 2, 2.25, \ldots, 4$, out of which we select the one that results in the highest F1 score based on $\text{supp}(x^*)$, i.e. the output of Algorithm 9. We do this for the experiment on the synthetic graph to illustrate how label accuracy affects local clustering performance. In practice, without the ground-truth information, one may fix a reasonable α, e.g. $\alpha = 2$, and then apply a the sweep cut procedure on x^*. We adopt the latter approach for experiments on real-world data.

In Figure 7.2 we compare the F1 scores achieved by FD and LFD over varying levels of label accuracy. Recall that FD does not use and hence is not affected by the labels at all, whereas LFD uses label-based edge weights from Equation (7.2). In addition to the

results obtained from flow diffusion, we also include the F1 scores obtained from the labels alone (**Labels**), i.e. we compare $\tilde{Y}_1 = \{i \in V : \tilde{y}_i = 1\}$ against K. Not surprisingly, as predicted by (7.5), the F1 of LFD increases as at least one of a_0, a_1 increases. Moreover, LFD already outperforms FD at reasonably low label accuracy, e.g. when $a_0, a_1 = 0.7$ and the F1 of the labels alone is as low as 0.2. This shows the effectiveness of incorporating noisy labels and employing diffusion over the label-weighted graph G^w. Even fairly noisy node labels can boost local clustering performance. In Figure 7.3 we compare the F1 scores achieved by employing the ℓ_1-regularized PageRank [50] over the original graph G and the label-weighted graph G^w, under the same setting as above. The results are similar.

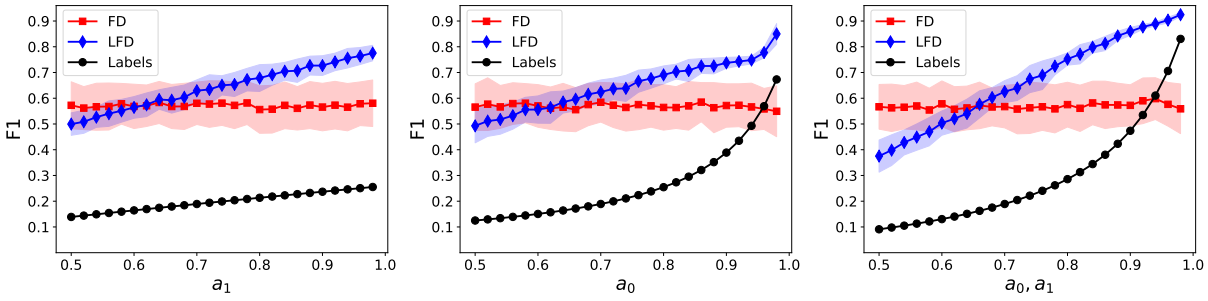


Figure 7.2: F1 scores obtained by employing flow diffusion over the original graph (FD) and the label-weighted graph (LFD). For comparison, we also plot the F1 obtained by the noisy labels (Labels). The solid line and error bar show mean and standard deviation over 100 trials, respectively. As discussed in Section 7.2.1, even fairly noisy labels can already help boost local clustering performance.

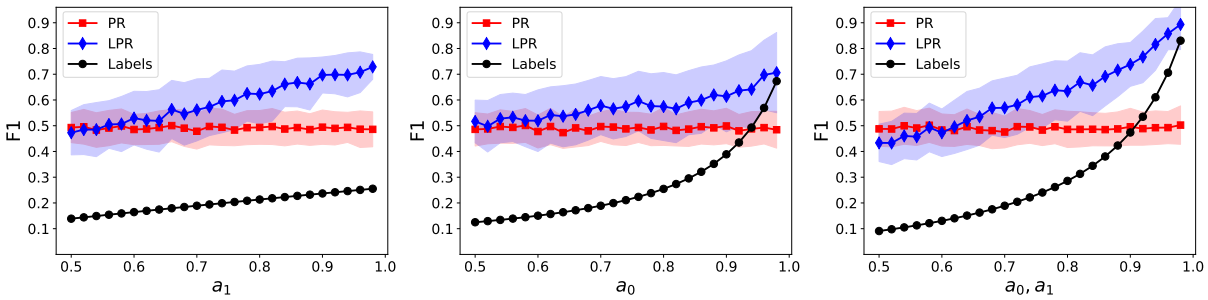


Figure 7.3: F1 scores obtained by employing ℓ_1-regularized PageRank [50] over the original graph (PR) and the label-weighted graph (LPR). For comparison, we also plot the F1 obtained by the noisy labels (Labels). The solid line and error bar show mean and standard deviation over 100 trials, respectively.

Empirical demonstration of conjectures in Section 7.2.2

We demonstrate our conjectures in Section 7.2.2 about when to set $\epsilon = 0$ versus $\epsilon > 0$. As before, we generate synthetic graphs using the stochastic block model with cluster size $k = 500$ an number of clusters $c = 20$. The number of nodes in the graph equals $n = kc = 10,000$. Two nodes within the same cluster are connected with probability p and two nodes from different clusters are connected with probability q. We consider different choices for $q, a_0, a1$ such that the condition in either Conjecture 1 or Conjecture 2 is satisfied.

In Table 7.1 we report the F1 scores obtained by setting $\epsilon = 0$ and $\epsilon = 0.2$, respectively. We average over 100 trials for each setting. For comparison purposes we also include the results obtained by employing Flow Diffusion (FD) over the original graph G. Note that FD is equivalent to setting $\epsilon = 1$. Observe that, when $(1-a_1)pk > a_0q(n-k)$ as required by Conjecture 1, setting $\epsilon = 0.2$ leads to a higher F1, whereas when $(1-a_1)p^2k < a_0q^2(n-k)$ as required by Conjecture 2, setting $\epsilon = 0$ leads to a higher F1. This demonstrates both conjectures.

Table 7.1: Empirical demonstration of our conjectures: F1 scores for local clustering in the local random graph model with different model parameters and label accuracy

	Empirical Setting	FD	LFD (ϵ=0)	LFD (ϵ=0.2)
Conjecture 1	p=0.05, q=0.0015, a_0=0.7, a_1=0.6	69.2	64.5	**77.8**
	p=0.05, q=0.0015, a_0=0.6, a_1=0.65	69.2	64.2	**74.6**
Conjecture 2	p=0.05, q=0.0075, a_0=0.8, a_1=0.7	9.7	**48.8**	37.3
	p=0.05, q=0.0075, a_0=0.9, a_1=0.9	9.7	**76.7**	61.1

From a practical point of view, we would like to point out that real networks often have much more complex structures than the synthetic graphs. Therefore the same conditions may not generalize to the real networks that one works with in practice. To that end, in Section 7.3.2 we provide an empirical study on the robustness of our method with respect to different values of ϵ. The empirical results in Section 7.3.2 indicate that, over real networks, the local clustering accuracy remains similar for different choices of ϵ, ranging from 0.01 to 0.2.

7.3.2 Experiments on Real-World Graphs

We carry out experiments over the following 6 real-world attributed graph datasets. We include the two datasets used in Section 6.3.2, namely Coauthor CS [106] and Coauthor Physics [106]. Additionally, we use 4 well-established graph machine learning benchmarks: Amazon Photo [92], Amazon Computers [106], Cora [93] and Pubmed [104]. We provide a description of the four new datasets below.

- Amazon Photo is a co-purchasing graph from Amazon ([92]), where nodes represent products and an edge indicates whether two products are frequently bought together. Labels of the nodes are determined by the product's category, while node attributes are bag-of-word encodings of product reviews. The dataset consists of 7,487 photographic equipment products, 119,043 co-purchasing connections, and 8 categories
- Amazon Computers is another co-purchasing graph extracted from Amazon ([106]), with the same structure as Amazon Photo. It has 13,752 computer equipment products, 245,861 connections, and 10 categories.
- Cora ([93]) is a citation network where each node denotes a scientific publication in Computer Science. An edge from node A to B indicates a citation from work A to work B. Despite their directed nature, we utilize an undirected version of these graphs for our analysis. The graph includes 2,708 publications, 5,429 edges, and 7 classes denoting the paper categories. The node features are bag-of-words encodings of the paper abstract.
- Pubmed ([104]) is a citation network with a similar structure as Cora. We also adopt an undirected version of the graph. The dataset categorizes medical publications into one of 3 classes and comprises 19,717 nodes and 44,338 edges. Node features are TF/IDF encodings from a selected dictionary.

We divide the experiments into two settings. In the first, we assume access to a selected number of ground-truth labels, evenly sampled from both the target and non-target classes. These nodes are utilized to train a classifier (without graph information). The predictions of the classifier are then used as noisy labels to construct the weighted graph G^w as defined in Equation (7.2), and we set $\epsilon = 0.05$ as in the synthetic experiments. We use all the positive nodes, i.e. nodes that belong to the target cluster based on the given ground-truth labels, as seed nodes during the diffusion process. For each cluster in each dataset, we compare LFD against FD and WFD over 100 trials. For each trial, a classifier is trained using randomly sampled positive and negative nodes which we treat as ground-truth information.

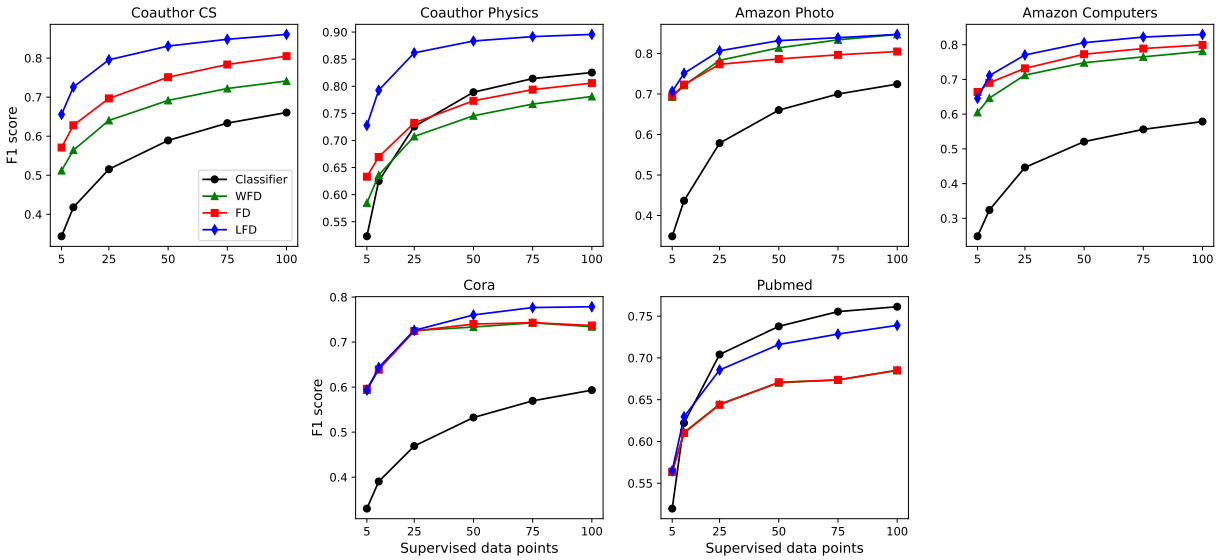


Figure 7.4: F1 scores for local clustering using Flow Diffusion (FD), Weighted Flow Diffusion (WFD), Label-based Flow Diffusion (LFD), and Logistic Regression (Classifier) with an increasing number of positive and negative ground-truth samples.

Figure 7.4 shows the average F1 obtained by each method versus the number of samples used for training the classifier. As illustrated in Figure 7.4, using the outputs from a weak classifier (e.g. with an F1 score as low as 40%) as noisy labels already enhances the diffusion process, obtaining an improvement as high as 13% over other methods (e.g. see Coauthor Physics with 25 positive and negative nodes). The increasing availability of ground-truth positive nodes typically benefits all diffusion processes. However, as seen in Cora, additional seed nodes can also increase the risk of mass leakage to the outside of the target class and hence result in lower accuracy. In such cases, the learned classifier mitigates this problem by reducing inter-edge weights. In Figure 7.5 we compare local clustering performance by employing PageRank over G and G^w, respectively. The results are similar.

In the second set of experiments, we consider the setting where we are only given a **single seed node** with no access to ground-truth labels or a pretrained classifier. To demonstrate the effectiveness of our method, a heuristic approach is adopted. First, we solve the flow diffusion problem over the original graph G and get a solution x^*. Then, we select 100 nodes with the highest and lowest values in x^*, which are designated as positive and negative nodes, respectively. We use these nodes to train a binary classifier. The idea

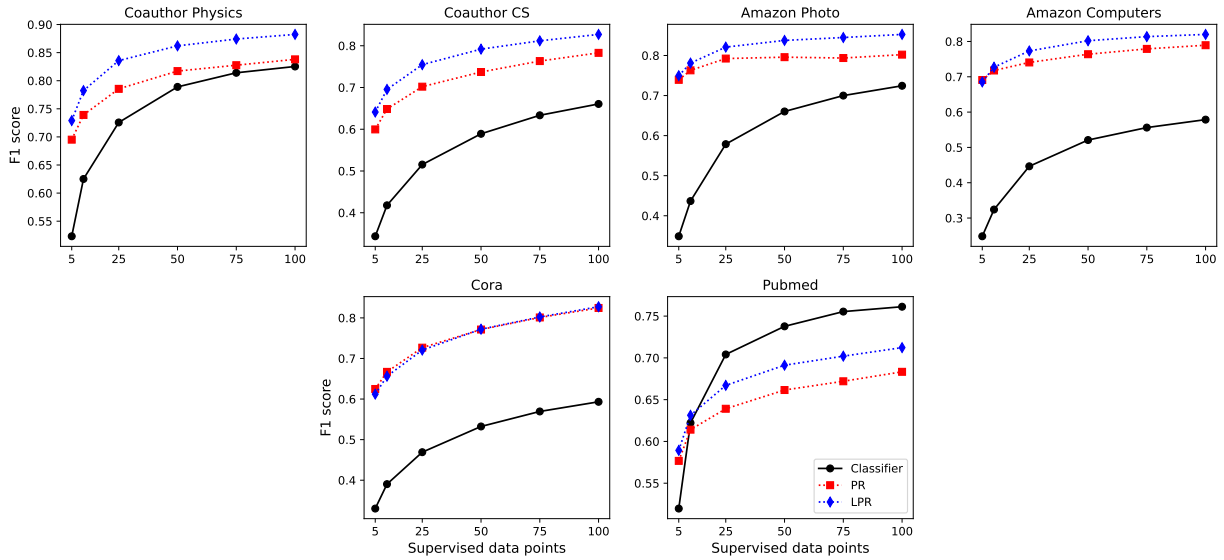


Figure 7.5: F1 scores for local clustering using ℓ_1-regularized PageRank (PR), Label-based PageRank (LPR), and Logistic Regression (Classifier) with an increasing number of positive and negative ground truth samples.

is that nodes with the highest values in x^* typically belong to the target cluster, whereas nodes with the lowest values in x^* — typically zero — are outside of the target cluster. We use the outputs of the classifier as noisy node labels to construct the weighted graph G^w. We test this approach against FD (i.e. flow diffusion over the original graph) and WFD (i.e. weighted flow diffusion from Chapter 6 where edge weights are defined based on the Gaussian kernel), both in the single and multi-seed settings. In the multi-seed setting, the 100 (pseudo-)positive nodes are used as seed nodes. Additionally, for each dataset, we compare LFD with the best-performing baseline and report the improvement in Table 7.2. The results demonstrate a consistent improvement of LFD over other methods across all datasets.

Hyperparameter analysis

We test the robustness of our method against various choices of hyperparameters. There are 2 hyperparameters in our method. The first hyperparameter is the edge weight $\epsilon \in [0, 1)$ from Equation (7.2), and the second hyperparameter is the total amount of source mass $\theta > 0$ at the seed node to initialize the flow diffusion process.

Table 7.2: Comparison of F1 scores across datasets for Flow Diffusion (FD), Weighted Flow Diffusion (WFD), and Label-based Flow Diffusion (LFD) in the absence of ground-truth information

Dataset	FD (single-seed)	WFD (single-seed)	FD (multi-seed)	WFD (multi-seed)	LFD	Improv. (±)	Improv. (%)
Coauthor CS	43.8	39.9	50.5	47.1	**63.1**	+12.6	+24.9
Coauthor Physics	62.8	57.0	55.5	51.1	**72.9**	+10.1	+16.1
Amazon Photo	54.5	57.4	62.1	62.6	**66.8**	+4.2	+6.7
Amazon Computers	56.2	53.3	58.2	54.6	**60.4**	+2.2	+3.8
Cora	33.3	33.7	55.4	55.4	**56.5**	+1.1	+1.9
Pubmed	53.0	53.2	53.9	53.9	**55.3**	+1.4	+2.7
AVERAGE	50.6	49.1	55.9	54.1	**62.5**	+5.3	+9.3

We conduct a detailed case study using the Coauthor CS dataset. Similar trends and results are seen in the experiments using other datasets. We focus on the one of the empirical setting where we are given 10 positive and 10 negative ground-truth node labels. Apart from the choices for ϵ and θ, we keep all other empirical settings the same and report the average local clustering result over 100 trails.

We vary the total amount of source mass θ as follows. Set $\theta = \alpha \text{vol}(K)$ and we let $\alpha \in \{2, 3, 4, 5\}$. Recall that K denotes the target cluster. Therefore picking α in the range of $[2, 5]$ results in very large variations in θ. Our experiments on real-world graphs use $\alpha = 2$ and $\epsilon = 0.05$. Here, we present results using different combinations of values for α and ϵ. Observe that for a fixed $\alpha \in \{2, 3, 4\}$, the maximum change in the F1 score across $\epsilon \in [10^{-2}, 10^{-1}]$ is 1.2. Moreover, for all combinations of ϵ and α, our method has a much higher F1, highlighting the effectiveness and robustness to incorporate noisy labels for local clustering.

Runtime analysis

We report the running time of our Label-based Flow Diffusion (LFD) along with other flow diffusion-based local methods. The experiments are run on an Intel i9-13900K CPU with 36MB Cache and 2 x 48GB DDR5 RAM. We highlight the fast running time of LFD. Fast running times are typically seen in local methods and are due to the fact that these methods do not require processing the entire graph. The runtimes reported in Table 7.4

Table 7.3: F1 scores for different values of source mass and inter-edge weight

			LFD					
α	FD	WFD	$\epsilon = 0.01$	$\epsilon = 0.025$	$\epsilon = 0.05$	$\epsilon = 0.075$	$\epsilon = 0.1$	$\epsilon = 0.2$
2	62.8	56.4	73.0	**73.1**	72.6	72.4	72.2	70.8
3	67.5	58.3	**74.8**	74.1	74.1	74.0	74.0	73.0
4	68.1	57.6	**73.4**	72.7	72.1	72.3	72.2	72.0
5	66.1	55.4	**71.8**	70.8	70.0	69.5	69.3	68.7

are based on the experiments using the Coauthor CS dataset, averaged over 10 trials across 15 clusters.

Table 7.4: Average runtimes for different local diffusion methods

	FD	WFD	LFD
Train model	-	-	0.09 ± 0.01 s
Calculate weights	-	1.11 ± 0.77 s	0.03 ± 0.02 s
Diffusion process	0.01 ± 0.01 s	0.01 ± 0.01 s	0.01 ± 0.01 s
TOTAL	0.01 ± 0.01 s	1.12 ± 0.78 s	0.13 ± 0.03 s

Comparison with Graph Convolutional Networks (GCNs)

Within the task of clustering or node classification in the presence of ground-truth node labels, it is natural to extend the comparison to other types of methods which exploit both the graph structure and node information. To that end, we provide an empirical comparison with the performance of GCNs in our problem setting. Note that both the training and the inference stages of GCNs require accessing every node in the graph, and this makes GCNs (and more generally other global methods that require full graph processing) unsuitable for local graph clustering. In contrast, local methods only explore a small portion of the graph around the seed node(s).

To highlight the strengths of our local method against global methods such as GCNs, we carried out additional experiments to compare both runtime and accuracy. Again, we fix the same empirical setting as before, that is, we use the Coauthor CS dataset and select 10 nodes each from positive and negative ground-truth categories. Let us remind the reader that, here, a positive ground-truth label means that a node selected from the

target cluster K is given a label 1. Similarly, a negative ground-truth label means that a node selected from the rest of the graph is given a label 0.

Table 7.5: Comparison between Label-based Flow Diffusion (LFD) and Graph Convolutional Network (GCN)

	LFD	GCN
F1 score	73.0	46.9
Runtime	0.13 ± 0.03 s	3.68 ± 0.31 s

We use a two-layer GCN architecture with a hidden layer size of 16. When training the GCN model, we terminate the training process after 100 epochs. In our approach, each class is treated separately in a one-vs-all classification framework during training. We replicate this procedure for each class across 10 independent trials. The results are shown in Table 7.5. Observe that our method not only runs substantially faster (i.e. 28 times faster) than a GCN but also obtains a much higher F1. The poor performance of GCNs is due to the scarcity of ground-truth data in our setting, where we only have 20 samples. GCNs generally require a much greater number of ground-truth labels to work well. In order to make GCN achieve a better accuracy than LFD, we had to increase the number of ground-truth labels to 600 samples, which is not very realistic for local clustering contexts.

7.4 Proofs of Results

7.4.1 Concentration Results

We will use some concentration results concerning the connectivity of the random graphs G and G^w. These results are mostly derived from straightforward applications of the Chernoff bound. For completeness we state these results and provide their proofs below.

Lemma 7.5 (External degree in G)**.** *If $q \geq \frac{(3+\epsilon)}{\delta^2}\frac{\log k}{n-k}$ for some $\epsilon > 0$ and $0 < \delta \leq 1$, then with probability at least $1 - k^{-\epsilon/3}$ we have that for all $i \in K$,*

$$|E(\{i\}, K^{\mathsf{c}})| \geq (1-\delta)q(n-k).$$

Proof. This follows directly by noting that, for each $i \in K$, $|E(\{i\}, K^{\mathsf{c}})|$ is the sum of independent Bernoulli random variables with mean $q(n-k)$. Applying a multiplicative Chernoff bound on $|E(\{i\}, K^{\mathsf{c}})|$ and then a union bound over $i \in K$ gives the result. □

Lemma 7.6 (Node degree in G^w)**.** *If $p \geq \frac{(6+\epsilon)}{\delta^2}\frac{\log k}{k-2}$ for some $\epsilon > 0$ and $0 < \delta \leq 1$, then with probability at least $1 - k^{-\epsilon/6}$ we have that for all $i \in K \cap \tilde{Y}_1$,*

$$\deg_{G^w}(i) \leq (1+\delta)(p(a_1 k - 1) + (1-a_0)q(n-k)).$$

Proof. For each node $i \in K \cap \tilde{Y}_1$, since $K \cap \tilde{Y}_1 = a_1 k$ and $K^{\mathsf{c}} \cap \tilde{Y}_1 = (1-a_0)(n-k)$, its degree in G^w, that is $\deg_{G^w}(i)$, is the sum of independent Bernoulli random variables with mean $\mathbb{E}(\deg_{G^w}(i)) = p(a_1 k - 1) + (1-a_0)q(n-k) \geq p(a_1 k - 1) \geq \frac{(3+\epsilon/2)}{\delta^2}\log k$. Apply the Chernoff bound we get

$$\mathbb{P}\left(\deg_{G^w}(i) \geq (1+\delta)\mathbb{E}(\deg_{G^w}(i))\right) \leq \exp(-\delta^2 \mathbb{E}(\deg_{G^w}(i))/3) \leq \exp(-(1+\epsilon/6)\log k).$$

Taking a union bound over all $i \in K \cap \tilde{Y}_1$ gives the result. □

Lemma 7.7 (Internal connectivity in G^w)**.** *If $p \geq \max(\frac{(6+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-2}, \frac{(\sqrt{8}+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-2}})$, then with probability at least $1 - 2k^{-\epsilon_1/6} - k^{-\epsilon_2}$, we have that for all $i, j \in K \cap \tilde{Y}_1$ where $i \neq j$, there are at least $(1-\delta_1)(1-\delta_2)p^2(a_1 k - 1)$ distinct paths connecting node i to node j such that, each of these paths consists of at most 2 edges, and each edge from G^w appears in at most one of these paths.*

Proof. Let F_i denote the set of neighbors of a node i in $K \cap \tilde{Y}_1$. By our assumption that $p \geq \frac{(6+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-2}$, we may take a Chernoff bound on the size of F_i and a union bound over all $i \in K \cap \tilde{Y}_1$ to get that, with probability at least $1 - 2k^{-\epsilon_1/6}$,

$$(1-\delta_1)p(a_1 k - 1) \leq |F_i| \leq (1+\delta_1)p(a_1 k - 1), \ \forall i \in K \cap \tilde{Y}_1.$$

If $j \notin F_i$, then since $|E(\{j\}, F_i)|$ is a sum of independent Bernoulli random variables with mean $|F_i|p$, we may apply the Chernoff bound and get that, with probability at least $1 - 2k^{-\epsilon_1/6}$ (under the event that $(1-\delta_1)p(a_1 k - 1) \leq |F_i| \leq (1+\delta_1)p(a_1 k - 1)$),

$$\begin{aligned}\mathbb{P}(|E(\{j\}, F_i)| \leq (1-\delta_2)|F_i|p) &\leq \exp(-\delta_2^2 |F_i|p/2) \\ &\leq \exp(-\delta_2^2(1-\delta_1)p^2(a_1 k - 1)/2)) \leq \exp(-(2+\epsilon_2)\log k). \end{aligned} \tag{7.8}$$

The last inequality in the above follows from our assumption that $p \geq \frac{(\sqrt{8}+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-2}}$. If $j \in F_i$, then the edge (i, j) is a path of length 1 connecting j to i, and moreover, let $\ell \in K \cap \tilde{Y}_1$ be such that $\ell \notin F_i$ and $\ell \neq i$, we have that

$$\begin{aligned}\mathbb{P}(|E(\{j\}, F_i \backslash \{j\})| + 1 \leq (1-\delta_2)|F_i|p) &\leq \mathbb{P}(|E(\{\ell\}, F_i)| \leq (1-\delta_2)|F_i|p) \\ &\leq \exp(-(2+\epsilon_2)\log k),\end{aligned}$$

where the last inequality follows from (7.8). Note that, for a node j in $K \cap \tilde{Y}_1$ such that $j \neq i$, each edge $(j, \ell) \in E(\{j\}, F_i \backslash \{j\})$ identifies a unique path (j, ℓ, i) and none of these paths has overlapping edges. Therefore, denote $P(i, j)$ the set of mutually non-overlapping paths of length at most 2 between i and j, and take union bound over all $i, j \in K \cap \tilde{Y}_1$, we get that

$$\mathbb{P}(\exists i, j \in K \cap \tilde{Y}_1, i \neq j, \text{s.t. } P(i, j) \leq (1 - \delta_2)|F_i|p) \leq k^{-\epsilon_2}.$$

Finally, taking a uninon bound over the above event and the event that there is $i \in K \cap \tilde{Y}_1$ such that $|F_i| < (1 - \delta_1)p(a_1 k - 1)$ gives the required result. □

7.4.2 Proof of Theorem 7.3

Recall that x^* denotes the optimal solution of Problem (6.2) over the original unweighted graph G and $x^\dagger$ denotes the optimal solution of Problem (6.2) over the weighted graph G^w. The proof is based on (1) lower bounding the number of false positives incurred by $\text{supp}(x^*)$, which we show in Proposition 7.8, (2) upper bounding the number of false positives incurred by $\text{supp}(x^\dagger)$, which we show in Proposition 7.9, and (3) combine both lower and upper bounds.

Let $\hat{x}$ denote a generic optimal solution of Problem (6.2), which is obtained over either G or G^w. We will heavily use the following two important properties of $\hat{x}$ (along with its physical interpretation). These properties follow from optimality conditions of Problem (6.2).

1. The solution $\hat{x} \in \mathbb{R}^n$ defines a flow diffusion over the underlying graph such that, for all $i, j \in V$, the amount of mass that node i sends to node j along edge $e(i, j)$ is given by $\hat{f}_e = w(i, j) \cdot (\hat{x}_i - \hat{x}_j)$, where $\hat{f}$ denotes the optimal solution for Problem (6.1).

2. For all $i \in V$, $\hat{x}_i > 0$ only if the total amount of mass at node i equals $T(i)$, i.e., $\Delta(i) + \sum_{j \sim i} w(i, j) \cdot (\hat{x}_j - \hat{x}_i) = T(i)$ (cf. Lemma 4.1).

3. For all $i \in V$, $\hat{x}_i > 0$ if and only if the total amount of mass that node i receives exceeds $T(i)$, i.e., $\Delta(i) + \sum_{j \sim i, \hat{x}_j > \hat{x}_i} w(i, j) \cdot (\hat{x}_j - \hat{x}_i) > T(i)$.

Using these properties we may easily obtain a lower bound on the number of false positives incurred by $\text{supp}(x^*)$. Recall that x^* denotes the optimal solution of Problem (6.2) over G, with sink capacity $T(i) = 1$ for all i. We state the result in Proposition 7.8.

Proposition 7.8. *If* $q \geq \frac{(3+\epsilon)}{\delta^2}\frac{\log k}{n-k}$ *for some* $\epsilon > 0$ *and* $0 < \delta \leq 1$*, then with probability at least* $1 - k^{-\epsilon/3}$*, for every seed node* $s \in K$*, if* $|\mathrm{supp}(x^*)| \geq 2$ *then we have that*

$$|\mathrm{supp}(x^*) \cap K^{\mathsf{c}}| > (1-\delta)\frac{p}{\gamma}(k-1).$$

Proof. If $|\mathrm{supp}(x^*)| \geq 2$ it means that $x^*_s > 0$ as otherwise we must have $x^* = 0$. Moreover, let $i \in V$ be such that $i \neq s$ and $x^*_i \geq x^*_j$ for all $j \neq s$. Then we must have that i is a neighbor of s. Because $\Delta_i = 0$, $x^*_i > 0$ and $x^*_i \geq x^*_j$ for all $j \neq s$, we know that the amount of mass that node s sends to node i is strictly larger than 1, and hence $x^*_s > x^*_i + 1 > 1$. But then this means that we must have $x^*_\ell > 0$ for all $\ell \sim s$. By Lemma 7.5 we know that with probability at least $1 - k^{-\epsilon/3}$, every node $i \in K$ has more than $(1-\delta)q(n-k)$ neighbors in K^{c}. This applies to s which was chosen arbitrarily from K. Therefore we have that with probability at least $1 - k^{-\epsilon/3}$, for every seed node $s \in K$, if $|\mathrm{supp}(x^*)| \geq 2$ then $|\mathrm{supp}(x^*) \cap K^{\mathsf{c}}| > (1-\delta)q(n-k)$. The required result then follows from our definition that $\gamma = \frac{p(k-1)}{q(n-k)}$. □

In the next, Proposition 7.9 provides an upper bound on the number of false positives incurred by $\mathrm{supp}(x^\dagger)$ under appropriately chosen source mass $\theta^\dagger$ at the seed node. Its proof is based on upper bounding the total amount of mass that leaks to the outside of the target cluster during a diffusion process, and it follows from similar reasoning as the proof of Theorem 6.9 from Chapter 6.

Proposition 7.9. *If* $p \geq \max(\frac{(6+\epsilon_1)}{\delta_1^2}\frac{\log k}{k-2}, \frac{(\sqrt{8}+\epsilon_2)}{\delta_2\sqrt{1-\delta_1}}\frac{\sqrt{\log k}}{\sqrt{k-2}})$ *for some* $0 < \delta_1, \delta_2 \leq 1$ *and* $\epsilon_1, \epsilon_2 > 0$*, then with probability at least* $1 - 3k^{-\epsilon_1/6} - k^{-\epsilon_2}$*, for every seed node* $s \in K \cap \tilde{Y}_1$ *with source mass*

$$\theta^\dagger = \left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}}\right)^2 r a_1 k,$$

we have that $K \cap \tilde{Y}_1 \subseteq \mathrm{supp}(x^\dagger)$ *and*

$$|\mathrm{supp}(x^\dagger) \cap K^{\mathsf{c}}| \leq \left(\left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}}\right)^2 r - 1\right) a_1 k.$$

Proof. To see that $K \cap \tilde{Y}_1 \subseteq \mathrm{supp}(x^\dagger)$, let us assume for the sake of contradiction that $x^\dagger_i = 0$ for some $i \in K \cap \tilde{Y}_1$. This means that node i receives at most 1 unit mass, because

otherwise we would have $x_i^\dagger > 0$. We also know that $i \neq s$ because $\Delta_s > 1$. Denote $F := \{j \in K \cap \tilde{Y}_1 : j \sim s\}$. We will consider two cases depending on if $i \in F$ or not.

Suppose that $i \in F$. Then we have that $x_s^\dagger - x_i^\dagger \leq 1$ because node i receives at most 1 unit mass from node s. This means that $x_s^\dagger \leq 1 + x_i^\dagger = 1$. It follows that the total amount of mass which flows out of node s is

$$\sum_{\ell \sim s}(x_s^\dagger - x_\ell^\dagger) \leq \sum_{\ell \sim s} x_s^\dagger \leq \deg_{G^w}(s) \leq (1+\delta)(p(a_1 k - 1) + (1-a_0)q(n-k)),$$

where the last inequality follows from Lemma 7.6. Therefore, we get that the total amount of source mass is at most

$$\begin{aligned}
\theta^\dagger &\leq (1+\delta)(p(a_1 k - 1) + (1-a_0)q(n-k)) + 1 \\
&= (1+\delta)p(a_1 k - 1)\frac{a_1 p(k - 1/a_1) + (1-a_0)q(n-k)}{a_1 p(k-1/a_1)} + 1 \\
&= (1+\delta)p(a_1 k - 1)\frac{a_1 \gamma^{\frac{(k-1/a_1)}{(k-1)}} + (1-a_0)}{a_1 \gamma^{\frac{(k-1/a_1)}{(k-1)}}} + 1 \\
&\leq (1+\delta)p(a_1 k - 1)\frac{a_1 \gamma^{\frac{(k-2)}{(k-1)}} + (1-a_0)}{a_1 \gamma^{\frac{(k-2)}{(k-1)}}} + 1 \\
&\leq (1+\delta)(1+2/k)\left(\frac{a_1 \gamma^{\frac{(k-2)}{(k-1)}} + (1-a_0)}{a_1 \gamma^{\frac{(k-2)}{(k-1)}}}\right) a_1 k < \theta^\dagger,
\end{aligned}$$

where the second last inequality follows from $a_1 \geq 1/2$. This is a contradiction, and hence we must have $i \notin F$.

Now, suppose that $i \notin F$. Then we know that the total amount of mass that node i receives from its neighbors is at most 1. In particular, node i receives at most 1 unit mass from nodes in F. This means that

$$\sum_{\substack{j \sim i \\ j \in F}} x_j^\dagger = \sum_{\substack{j \sim i \\ j \in F}} (x_j^\dagger - x_i^\dagger) \leq 1.$$

By Lemma 7.7, we know that with probability at least $1 - 2k^{-\epsilon_1/6} - k^{-\epsilon_2}$, node i has at least $(1-\delta_1)(1-\delta_2)p^2(a_1 k - 1)$ neighbors in F, and thus

$$\sum_{\substack{j \in F \\ j \sim i}} x_j^\dagger \leq 1 \implies \min_{j \in F} x_j^\dagger \leq \frac{1}{(1-\delta_1)(1-\delta_2)p^2(a_1 k - 1)}$$

Therefore, let $j \in F$ a node such that $x_j^\dagger \le x_\ell^\dagger$ for all $\ell \in F$, then with probability at least $1 - 2k^{-\epsilon_1/6} - k^{-\epsilon_2}$,

$$x_j^\dagger \le \frac{1}{(1-\delta_1)(1-\delta_2)p^2(a_1k-1)}. \tag{7.9}$$

By Lemma 7.7, with probability at least $1-2k^{-\epsilon_1/6}-k^{-\epsilon_2}$, node j has at least $(1-\delta_1)(1-\delta_2)p^2(a_1k-1)-1$ neighbors in F. Since $x_j^\dagger \le x_\ell^\dagger$ for all $\ell \in F$ and $x_j^\dagger \le x_s^\dagger$, we know that

$$|\{\ell \in V : \ell \sim j \text{ and } x_\ell^\dagger \ge x_j^\dagger\}| \ge (1-\delta_1)(1-\delta_2)p^2(a_1k-1). \tag{7.10}$$

Therefore, with probability at least $1-3k^{-\epsilon_1/3}-k^{-\epsilon_2}$, the total amount of mass that node j sends out to its neighbors is at most

$$\begin{aligned}
&\sum_{\ell\sim j}(x_j^\dagger - x_\ell^\dagger) \le \sum_{\substack{\ell\sim j\\ x_\ell^\dagger \le x_j^\dagger}}(x_j^\dagger - x_\ell^\dagger) \le \sum_{\substack{\ell\sim j\\ x_\ell^\dagger \le x_j^\dagger}} x_j^\dagger \\
&\overset{\text{(i)}}{\le} \Big((1+\delta_1)(p(a_1k-1)+(1-a_0)q(n-k)) - (1-\delta_1)(1-\delta_2)p^2(a_1k-1)\Big)x_j^\dagger \\
&\overset{\text{(ii)}}{\le} \frac{(1+\delta_1)}{(1-\delta_1)(1-\delta_2)}\left(\frac{p(a_1k-1)+(1-a_0)q(n-k)}{p^2(a_1k-1)}\right) - 1.
\end{aligned}$$

where (i) follows from Lemma 7.6 and (7.10), and (ii) follows from (7.9). Since node j settles 1 unit mass, the total amount of mass that node j receives from its neighbors is therefore at most

$$\frac{(1+\delta_1)}{(1-\delta_1)(1-\delta_2)}\left(\frac{p(a_1k-1)+(1-a_0)q(n-k)}{p^2(a_1k-1)}\right).$$

Recall that the amount of mass that node j receives from node s is given by $x_s^\dagger - x_j^\dagger$, and hence we get

$$x_s^\dagger \le \frac{(1+\delta_1)}{(1-\delta_1)(1-\delta_2)}\left(\frac{p(a_1k-1)+(1-a_0)q(n-k)}{p^2(a_1k-1)}\right) + x_j^\dagger. \tag{7.11}$$

Apply the same reasoning as before, we get that with probability at least $1-3k^{-\epsilon_1/6}-k^{-\epsilon_2}$, the total amount of mass that is sent out from node s is

$$\sum_{\ell\sim s}(x_s^\dagger - x_\ell^\dagger) \;<\; \deg_{G^w}(s)\cdot x_s^\dagger \;\overset{\text{(i)}}{\le}\; (1+\delta_1)(p(a_1k-1)+(1-a_0)q(n-k))\cdot x_s^\dagger$$

$$
\begin{aligned}
&\overset{\text{(ii)}}{\leq} \frac{(1+\delta_1)}{(1-\delta_1)(1-\delta_2)} \left((1+\delta_1) \frac{(p(a_1k-1)+(1-a_0)q(n-k))^2}{p^2(a_1k-1)^2} \right. \\
&\qquad\qquad \left. + \frac{p(a_1k-1)+(1-a_0)q(n-k)}{p^2(a_1k-1)^2} \right) (a_1k-1) \\
&\overset{\text{(iii)}}{\leq} \frac{(1+\delta_1)(1+\delta_1+\frac{2}{p(k-1)})}{(1-\delta_1)(1-\delta_2)} \left(\frac{p(a_1k-1)+(1-a_0)q(n-k)}{p(a_1k-1)} \right)^2 (a_1k-1) \\
&\overset{\text{(iv)}}{=} \frac{(1+\delta_1)(1+\delta_1+\frac{2}{p(k-1)})}{(1-\delta_1)(1-\delta_2)} \left(\frac{a_1\gamma\frac{(k-1/a_1)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-1/a_1)}{(k-1)}} \right)^2 (a_1k-1) \\
&\overset{\text{(v)}}{\leq} \frac{(1+\delta_1)(1+\delta_1+\frac{2}{p(k-1)})}{(1-\delta_1)(1-\delta_2)} \left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}} \right)^2 (a_1k-1),
\end{aligned}
$$

where (i) follows from Lemma 7.6, (ii) follows from (7.9) and (7.11), (iii) and (v) follow from $a_1 \geq 1/2$, and (iv) follows from the definition $\gamma = \frac{p(k-1)}{q(n-k)}$. This implies that the total amount of source mass is

$$
\theta^\dagger < \frac{(1+\delta_1)(1+\delta_1+\frac{2}{p(k-1)})}{(1-\delta_1)(1-\delta_2)} \left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}} \right)^2 a_1k = \theta^\dagger
$$

which is a contradiction. Therefore we must have $i \notin K \cap \tilde{Y}_1$, but then this contradicts our assumption that $i \in K \cap \tilde{Y}_1$. Since our choice of $i, s \in K_1$ were arbitrary, this means that $x_i^\dagger > 0$ for all $i \in K_1$ and for all $s \in K_1$.

Finally, the upper bound on $|\text{supp}(x^\dagger) \cap K^\complement|$ follows directly from the fact that $x_i^\dagger > 0$ only if node i settles 1 unit mass. □

By Proposition 7.8, the F1 score for $\text{supp}(x^*)$ is at most

$$
\text{F1}(\text{supp}(x^*)) < \frac{2k}{2k + (1-\delta_3)p(k-1)/\gamma}.
$$

By Proposition 7.9, the F1 score for $\text{supp}(x^\dagger)$ is at least

$$
\text{F1}(\text{supp}(x^\dagger)) \geq \frac{2a_1k}{2a_1k + \left(\left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}} \right)^2 r - 1 \right) a_1k + (1-a_1)k}.
$$

Therefore, a sufficient condition for $\mathrm{F1}(\mathrm{supp}(x^\dagger)) \geq \mathrm{F1}(\mathrm{supp}(x^*))$ is

$$\begin{aligned}
&\left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}}\right)^2 r - 1 + \frac{(1-a_1)}{a_1} \leq (1-\delta_3)\frac{p}{\gamma}\frac{(k-1)}{k} \\
\iff &\left(\frac{a_1\gamma\frac{(k-2)}{(k-1)} + (1-a_0)}{a_1\gamma\frac{(k-2)}{(k-1)}}\right)^2 \leq \frac{1}{r}\left((1-\delta_3)\frac{p}{\gamma}\frac{(k-1)}{k} + \frac{2a_1-1}{a_1}\right) \\
\iff &\ a_1\gamma\frac{(k-2)}{(k-1)} + 1 - a_0 \leq a_1\gamma\frac{(k-2)}{(k-1)}\sqrt{\frac{1}{r}\left((1-\delta_3)\frac{p}{\gamma}\frac{(k-1)}{k} + \frac{2a_1-1}{a_1}\right)} \\
&\qquad = a_1\gamma\frac{(k-2)}{(k-1)}\sqrt{\frac{p/\gamma}{r'} + \frac{2a_1-1}{ra_1}} \\
\iff &\ a_0 \geq 1 - \frac{(k-2)}{(k-1)}\left(\sqrt{\frac{p/\gamma}{r'} + \frac{2a_1-1}{ra_1}} - 1\right)a_1\gamma.
\end{aligned}$$

Finally, setting $S = K \cap \tilde{Y}_1$ completes the proof of Theorem 7.3.

Chapter 8

Conclusion and Future Work

This thesis introduces several new perspectives to graph diffusion, covering computation, clustering, statistical recovery, and applications to local community detection and node classification problems on graphs and hypergraphs. There are several limitations of this work, for example, the overlap assumption (cf. Assumption 4.2 and Assumption 5.3) in the context of local partitioning, the density assumption (cf. Theorem 6.9 and Theorem 7.3 and requirement for p therein) in the context of statistical cluster recovery, are rather strong. I hope that future research in graph diffusion methods can extend the ideas presented in this thesis, in both theoretical and practical aspects.

On the theoretical side, and in particular for the graph setting, the overlap assumption (cf. Assumption 4.2) between the initial seed set and a low-conductance cluster is typically assumed by flow-based methods [95, 49] which achieve tighter approximation guarantee with a better running time complexity. Although our current theory indicates a natural tradeoff between clustering quality and computational cost (cf. Theorem 4.8), it is apparently not tight. Since our empirical results show that p-norm flow diffusion can work extremely well even in the single seed node setting, it is not unreasonable to expect that the same $\Phi(C)^{1/q}$ bound holds even with diffusion from a single seed node. Therefore, an exciting future work is to adopt a different technical approach and obtain a tight approximation guarantee in this setting. This will likely result in a local diffusion algorithm with the best known bound. On the other hand, one may consider cost functions other than the p-norm to go beyond the ℓ_2-norm which induces the random walk-like diffusion process. For example, adopting a similar approach as presented in Section 4.2.2, one may show that penalizing the cost $\|f\|_2 + \lambda\|f\|_\infty$ (as opposed to $\|f\|_p$) in Problem (4.1) can lead to constant approximation $\Phi(C)/\alpha$, for appropriately chosen λ, where α is the ratio of the overlap. This matches the approximation guarantee of flow-based methods. How-

ever, it has the additional advantage that it can also work with a single seed node. It seems true that the objective function $\|f\|_2 + \lambda\|f\|_\infty$ gives rise to a single method which interpolates between spectral methods (which work on single seed but are subject to the Cheeger's barrier) and flow-based methods (which do not work on single seed, but have good approximation power if there is a good initial overlap), depending on the size of the seed set. Proving such a result is not only interesting on its own, but also will remove practitioner's need to having to choose between diffusion-based methods and flow-based methods.

On the application side, newer and more advanced (and hence more complex) diffusion methods require a lot of efforts to be eventually deployed in practice. We need real applications (not just experiments on real data or benchmarks) to validate the advantage of newer methods. To accomplish this it is imperative to work with domain experts on specific domain problems. To that end, we made an attempt to work with epidemiologist and successfully demonstrated a good use case of p-norm flow diffusion in the context of epidemic intervention [137]. However, it is far from enough. An important future direction is to take advanced diffusion and local clustering methods from the computer science community and the theory community, and apply them to solve real problems, e.g. in the analysis of galaxy data or particle physics, beyond the usual context of social and internet networks. I hope that future research in this field become more interdisciplinary in nature and foster collaborations across different branches of science and social science, in which a fundamental understanding of the structure of networks is vital for applications of wider interest.

References

[1] E. Abbe. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2018.

[2] E. Abbe, J. Fan, and K. Wang. An ℓ_p theory of pca and spectral clustering. *The Annals of Statistics*, 50(4):2359–2385, 2022.

[3] V. L. Alev and L. C. Lau. Improved analysis of higher order random walks and applications. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, 2020.

[4] Z. Allen-Zhu, S. Lattanzi, and S. M. Vahab. A local algorithm for finding well-connected clusters. In *International Conference on Machine Learning (ICML)*, 2013.

[5] I. Amburg, N. Veldt, and A. R. Benson. Clustering in graphs and hypergraphs with categorical edge labels. In *The Web Conference (WWW)*, 2020.

[6] K. Ameranis, A. F. Depavia, L. Orecchia, and E. Tani. Fast algorithms for hypergraph PageRank with applications to semi-supervised learning. In *International Conference on Machine Learning (ICML)*, 2024.

[7] R. Andersen, C. Borgs, J. Chayes, J. Hopcroft, V. Mirrokni, and S.-H. Teng. Local computation of pagerank contributions. *Internet Mathematics*, 5(1-2):23–45, 2008.

[8] R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

[9] R. Andersen, S. O. Gharan, Y. Peres, and L. Trevisan. Almost optimal local graph clustering using evolving sets. *Journal of the ACM*, 2016.

[10] R. Andersen and K. J. Lang. An algorithm for improving graph partitions. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008.

[11] E. Arias-Castro, E. J. Candès, and A. Durand. Detection of an anomalous cluster in a network. *The Annals of Statistics*, pages 278–304, 2011.

[12] E. Arias-Castro, E. J. Candès, H. Helgason, and O. Zeitouni. Searching for a trail of evidence in a maze. *The Annals of Statistics*, 36(4):1726–1757, 2008.

[13] K. Avrachenkov, A. Kadavankandy, and N. Litvak. Mean field analysis of personalized pagerank with implications for local graph clustering. *Journal of Statistical Physics*, 2018.

[14] K. Avrachenkov, N. Litvak, D. Nemirovsky, and N. Osipova. Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM Journal on Numerical Analysis*, 45(2):890–904, 2007.

[15] F. Bach. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends in Machine Learning*, 6(2-3):145–373, 2013.

[16] B. Bahmani, K. Chakrabarti, and D. Xin. Fast personalized pagerank on mapreduce. In *ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2011.

[17] A. Baranwal, K. Fountoulakis, and A. Jagannath. Graph convolution for semi-supervised classification: Improved linear separability and out-of-distribution generalization. In *International Conference on Machine Learning (ICML)*, 2021.

[18] A. Baranwal, K. Fountoulakis, and A. Jagannath. Effects of graph convolutions in multi-layer networks. In *International Conference on Learning Representations (ICLR)*, 2023.

[19] A. Baranwal, A. Jagannath, and K. Fountoulakis. Optimality of message-passing architectures for sparse graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[20] A. Beck. On the convergence of alternating minimization for convex programming with applications to iteratively reweighted least squares and decomposition schemes. *SIAM Journal on Optimization*, 25(1):185–209, 2015.

[21] A. R. Benson, D. F. Gleich, and J. Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

[22] N. Binkiewicz, J. T. Vogelstein, and K. Rohe. Covariate-assisted spectral clustering. *Biometrika*, 104(2):361–377, 03 2017.

[23] C. Borgs, M. Brautbar, J. Chayes, and S.-H. Teng. Multiscale matrix sampling and sublinear-time pagerank computation. *Internet Mathematics*, 10(1-2):20–48, 2014.

[24] G. Braun, H. Tyagi, and C. Biernacki. An iterative clustering algorithm for the contextual stochastic block model with optimality guarantees. In *International Conference on Machine Learning (ICML)*, 2022.

[25] S. D. Brown, J. A. Gerlt, J. L. Seffernick, and P. C. Babbitt. A gold standard set of mechanistically diverse enzyme superfamilies. *Genome Biology*, 7(1), 2006.

[26] T. Bühler and M. Hein. Spectral clustering based on the graph p-laplacian. In *International Conference on Machine Learning (ICML)*, 2009.

[27] S. O. Chan, T. C. Kwok, and L. C. Lau. Random walks and evolving sets: Faster convergences and limitations. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017.

[28] L. Chen, R. Peng, and D. Wang. 2-norm flow diffusion in near-linear time. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2022.

[29] Y. Chen and J. Xu. Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. *The Journal of Machine Learning Research*, 17(1):882–938, 2016.

[30] E. Chien, P. Li, and O. Milenkovic. Landing probabilities of random walks for seed-set expansion in hypergraphs. In *IEEE Information Theory Workshop (ITW)*, 2021.

[31] U. Chitra, K. Ding, J. C. H. Lee, and B. J. Raphael. Quantifying and reducing bias in maximum likelihood estimation of structured anomalies. In *International Conference on Machine Learning (ICML)*, 2021.

[32] U. Chitra and B. Raphael. Random walks on hypergraphs with edge-dependent vertex weights. In *International Conference on Machine Learning (ICML)*, 2019.

[33] P. S. Chodrow, N. Veldt, and A. R. Benson. Generative hypergraph clustering: From blockmodels to modularity. *Science Advances*, 7(28), 2021.

[34] K. Choromanski. Taming graph kernels with random features. In *International Conference on Machine Learning (ICML)*, 2023.

[35] P. Christiano, J. A. Kelner, A. Madry, D. A. Spielman, and S.-H. Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *ACM Symposium on Theory of Computing (STOC)*, 2011.

[36] F. Chung. Random walks and local cuts in graphs. *Linear Algebra and its Applications*, 423(1):22–32, 2007.

[37] F. Chung. A local graph partitioning algorithm using heat kernel pagerank. *Internet Mathematics*, 6(3):315–330, 2009.

[38] A. Back de Luca, K. Fountoulakis, and S. Yang. Local graph clustering with noisy labels. In *International Conference on Learning Representations (ICLR)*, 2024.

[39] Y. Deshpande, S. Sen, A. Montanari, and E. Mossel. Contextual stochastic block models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[40] M. Dreveton, F. S. Fernandes, and D. R. Figueiredo. Exact recovery and bregman hard clustering of node-attributed stochastic block model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[41] I. Ekeland and R. Témam. *Convex Analysis and Variational Problems*. Society for Industrial and Applied Mathematics, 1999.

[42] C. Eksombatchai, P. Jindal, J. Z. Liu, Y. Liu, R. Sharma, C. Sugnet, M. Ulrich, and J. Leskovec. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *The Web Conference (WWW)*, 2018.

[43] C. Eksombatchai, J. Leskovec, R. Sharma, C. Sugnet, and M. Ulrich. Node graph traversal methods. U.S. Patent 10 762 134 B1, Sep. 2020.

[44] I. Falih, N. Grozavu, R. Kanawati, and Y. Bennani. Community detection in attributed network. In *The Web Conference (WWW)*, 2018.

[45] D. Fogaras, B. Rácz, K. Csalogány, and T. Sarlós. Towards scaling fully personalized pagerank: Algorithms, lower bounds, and experiments. *Internet Mathematics*, 2(3):333–358, 2005.

[46] K. Fountoulakis, D. F. Gleich, and M. W. Mahoney. An optimization approach to locally-biased graph algorithms. *Proceedings of the IEEE*, 105(2):256–272, 2017.

[47] K. Fountoulakis, A. Levi, S. Yang, A. Baranwal, and A. Jagannath. Graph attention retrospective. *The Journal of Machine Learning Research*, 24, 2023.

[48] K. Fountoulakis, P. Li, and S. Yang. Local hyper-flow diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[49] K. Fountoulakis, M. Liu, D. F. Gleich, and M. W Mahoney. Flow-based algorithms for improving clusters: A unifying framework, software, and performance. *SIAM Review*, 2023.

[50] K. Fountoulakis, F. Roosta-Khorasani, J. Shun, X. Cheng, and M. W. Mahoney. Variational perspective on local graph clustering. *Mathematical Programming*, 174:553–573, 2017.

[51] K. Fountoulakis, D. Wang, and S. Yang. p-norm flow diffusion for local graph clustering. *International Conference on Machine Learning (ICML)*, 2020.

[52] S. Freitas, N. Cao, Y. Xia, D. H. P. Chau, and H. Tong. Local partition in rich graphs. In *IEEE International Conference on Big Data*. IEEE, 2018.

[53] J. Gasteiger, S. Weißenberger, and S. Günnemann. Diffusion improves graph learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[54] S. Oveis Gharan and L. Trevisan. Approximating the expansion profile and almost optimal local graph clustering. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2012.

[55] D. Ghoshdastidar and A. Dukkipati. Consistency of spectral partitioning of uniform hypergraphs under planted partition model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[56] C. L. Giles, K. D. Bollacker, and S. Lawrence. Citeseer: an automatic citation indexing system. In *Digital library*, 1998.

[57] D. F. Gleich. Pagerank beyond the web. *SIAM Review*, 57(3):321–363, 2015.

[58] D. F. Gleich and M. W. Mahoney. Anti-differentiating approximation algorithms:a case study with min-cuts, spectral, and flow. In *International Conference on Machine Learning (ICML)*, 2014.

[59] A. Green, S. Balakrishnan, and R. J. Tibshirani. Statistical guarantees for local spectral clustering on random neighborhood graphs. *Journal of Machine Learning Research*, 2021.

[60] W. Ha, K. Fountoulakis, and M. W. Mahoney. Statistical guarantees for local graph clustering. *The Journal of Machine Learning Research*, 22(1):6538–6591, 2021.

[61] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[62] K. Hayashi, S. G. Aksoy, C. H. Park, and H. Park. Hypergraph random walks, laplacians, and clustering. In *ACM International Conference on Information & Knowledge Management (CIKM)*, 2020.

[63] R. Ibrahim and D. Gleich. Nonlinear diffusion for community detection and semi-supervised learning. In *The Web Conference (WWW)*, 2019.

[64] R. Ibrahim and D. F. Gleich. Local hypergraph clustering using capacity releasing diffusion. *PLOS ONE*, 15(12):1–20, 12 2020.

[65] L. G. S. Jeub, P. Balachandran, M. A. Porter, P. J. Mucha, and M. W. Mahoney. Think locally, act locally: Detection of small, medium-sized, and large communities in large networks. *Physical Review E*, 91, 2015.

[66] C. Jia, Y. Li, M. B. Carson, X. Wang, and J. Yu. Node attribute-enhanced community detection in complex networks. *Scientific reports*, 7(1):1–15, 2017.

[67] M. Kapralov, S. Lattanzi, N. Nouri, and J. Tardos. Efficient and local parallel random walks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[68] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[69] K. Kloster and D. F. Gleich. Heat kernel based community detection. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[70] I. M. Kloumann and J. M. Kleinberg. Community membership identification from small seed sets. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[71] T. C. Kwok and L. C. Lau. Finding small sparse cuts by random walk. In *Proceedings of the 16th International Workshop on Randomization and Computation (RANDOM)*, 2012.

[72] T. C. Kwok, L. C. Lau, and Y. T. Lee. Improved cheeger's inequality and analysis of local graph partitioning using vertex expansion and expansion profile. *SIAM Journal on Computing*, 46(3):890–910, 2017.

[73] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78, 2008.

[74] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

[75] J. Leskovec, K.J. Lang, A. Dasgupta, and M.W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

[76] P. Li, N. He, and O. Milenkovic. Quadratic decomposable submodular function minimization: Theory and practice. *Journal of Machine Learning Research*, 21(106):1–49, 2020.

[77] P. Li and O. Milenkovic. Inhomogeneous hypergraph clustering with applications. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[78] P. Li and O. Milenkovic. Submodular hypergraphs: p-laplacians, cheeger inequalities and spectral clustering. In *International Conference on Machine Learning (ICML)*, 2018.

[79] M. Liu and D. F. Gleich. Strongly local p-norm-cut algorithms for semi-supervised learning and local graph clustering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[80] M. Liu, N. Veldt, H. Song, P. Li, and D. F. Gleich. Strongly local hypergraph diffusions for clustering and semi-supervised learning. In *The Web Conference (WWW)*, 2021.

[81] J. Łącki, S. Mitrović, K. Onak, and P. Sankowski. Walking randomly, massively, and efficiently. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, 2020.

[82] J. Łącki, S. Mitrović, K. Onak, and P. Sankowski. Walking randomly, massively, and efficiently. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, 2020.

[83] P. Lofgren, S. Banerjee, and A. Goel. Personalized pagerank estimation and search: A bidirectional approach. In *ACM International Conference on Web Search and Data Mining (WSDM)*, 2016.

[84] P. A. Lofgren, S. Banerjee, A. Goel, and C. Seshadhri. Fast-ppr: scaling personalized pagerank estimation for large graphs. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[85] L. Lovász. Random walks on graphs. *Combinatorics, Paul erdos is eighty*, 2(1-46):4, 1993.

[86] H. Ma, H. Yang, M. R. Lyu, and I. King. Mining social networks using heat diffusion processes for marketing candidates selection. In *ACM Conference on Information and Knowledge Management (CIKM)*, 2008.

[87] P. Macgregor and H. Sun. Local algorithms for finding densely connected clusters. In *International Conference on Machine Learning (ICML)*, 2021.

[88] 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. *The Journal of Machine Learning Research*, 13(1):2339–2365, 2012.

[89] D. Martínez-Rubio, E. Wirth, and S. Pokutta. Accelerated and sparse algorithms for approximate personalized pagerank and beyond. In *Conference on Learning Theory (COLT)*, 2023.

[90] R. Mastrandrea, J. Fournet, and A. Barrat. Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLOS ONE*, 10(9):e0136497, 2015.

[91] N. Masuda, M. A. Porter, and R. Lambiotte. Random walks and diffusion on networks. *Physics Reports*, 2017.

[92] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel. Image-based recommendations on styles and substitutes. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, 2015.

[93] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3:127–163, 2000.

[94] J. Ni, J. Li, and J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019.

[95] L. Orecchia and Z. A. Zhu. Flow-based algorithms for local graph clustering. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014.

[96] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[97] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[98] J. Qian and V. Saligrama. Efficient minimax signal detection on graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[99] P. Raghavendra and D. Steurer. Graph expansion and the unique games conjecture. In *ACM Symposium on Theory of Computing (STOC)*, 2010.

[100] A. Reid and P. Yuval. Finding sparse cuts locally using evolving sets. In *ACM Symposium on Theory of Computing (STOC)*, 2009.

[101] I. Reid, K. Choromanski, E. Berger, and A. Weller. General graph random features. In *International Conference on Learning Representations (ICLR)*, 2024.

[102] T. Saranurak and D. Wang. Expander decomposition and pruning: Faster, stronger, and simpler. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2019.

[103] A. Das Sarma, A. R. Molla, G. Pandurangan, and E. Upfal. Fast distributed pagerank computation. *Theoretical Computer Science*, 561:113–121, 2015.

[104] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93–93, 2008.

[105] J. L. Sharpnack, A. Krishnamurthy, and A. Singh. Near-optimal anomaly detection in graphs using lovasz extended scan statistic. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013.

[106] O. Shchur, M. Mumme, A. Bojchevski, and S. Günnemann. Pitfalls of graph neural network evaluation. *Relational Representation Learning Workshop, NeurIPS 2018*, 2018.

[107] P. Shi, K. He, D. Bindel, and J. Hopcroft. Local Lanczos spectral approximation for community detection. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2017.

[108] P. Shi, K. He, D. Bindel, and J. E. Hopcroft. Local lanczos spectral approximation for community detection. In *Machine Learning and Knowledge Discovery in Databases*, 2017.

[109] J. Shun, F. Roosta-Khorasani, K. Fountoulakis, and M. W. Mahoney. Parallel local graph clustering. In *International Conference on Very Large Data Bases (VLDB)*, 2016.

[110] A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. Hsu, and K. Wang. An overview of microsoft academic service (MAS) and applications. In *International World Wide Web Conference (WWW)*, 2015.

[111] L. M. Smith, L. Zhu, K. Lerman, and A. G. Percus. Partitioning networks with node attributes by compressing information flow. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(2), November 2016.

[112] D. A. Spielman and S.-H. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *ACM Symposium on Theory of Computing (STOC)*, 2004.

[113] D. A. Spielman and S.-H. Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Computing*, 2013.

[114] S. Stergiou. Scaling pagerank to 100 billion pages. In *The Web Conference (WWW)*, 2020.

[115] H. Sun, F. He, J. Huang, Y. Sun, Y. Li, C. Wang, L. He, Z. Sun, and X. Jia. Network embedding for community detection in attributed networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(3):1–25, 2020.

[116] Y. Takai, A. Miyauchi, M. Ikeda, and Y. Yoshida. Hypergraph clustering based on pagerank. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2020.

[117] A. L. Traud, P. J. Mucha, and M. A. Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, 391(16):4165–4180, 2012.

[118] A. Tsitsulin, M. Munkhoeva, D. Mottin, P. Karras, I. Oseledets, and E. Müller. Frede: anytime graph embeddings. In *International Conference on Very Large Data Bases (VLDB)*, 2021.

[119] N. Veldt, A. R. Benson, and J. Kleinberg. Minimizing localized ratio cut objectives in hypergraphs. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2020.

[120] N. Veldt, A. R. Benson, and J. Kleinberg. Hypergraph cuts with general splitting functions. *SIAM Review*, 64(3):650–685, 2022.

[121] N. Veldt, D. F. Gleich, and M. Mahoney. A simple and strongly-local flow-based method for cut improvement. In *International Conference on Machine Learning (ICML)*, 2016.

[122] N. Veldt, C. Klymko, and D. F. Gleich. Flow-based local graph clustering with better seed set inclusion. In *SIAM International Conference on Data Mining (SDM)*, 2019.

[123] R. Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

[124] D. Wang, K. Fountoulakis, M. Henzinger, M. W. Mahoney, and S. Rao. Capacity releasing diffusion for speed and locality. In *International Conference on Machine Learning (ICML)*, 2017.

[125] H. Wang, Z. Wei, J.-R. Wen, and M. Yang. Revisiting local computation of pagerank: Simple and optimal. In *ACM Symposium on Theory of Computing (STOC)*, 2024.

[126] P. Wang, S. Yang, Y. Liu, Z. Wang, and P. Li. Equivariant hypergraph diffusion neural operators. In *International Conference on Learning Representations (ICLR)*, 2023.

[127] R. Wang, A. Baranwal, and K. Fountoulakis. Analysis of corrected graph convolutions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[128] S. Wang, R. Yang, X. Xiao, Z. Wei, and Y. Yang. Fora: Simple and effective approximate single-source personalized pagerank. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2017.

[129] R. Wei, H. Yin, J. Jia, A. R. Benson, and P. Li. Understanding non-linearity in graph neural networks from the perspective of bayesian inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[130] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *ACM International Conference on Web Search and Data Mining (WSDM)*, 2010.

[131] X. Wu, Z. Chen, W. Wang, and A. Jadbabaie. An non-asymptotic analysis of oversmoothing in graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2023.

[132] W. Xie, D. Bindel, A. Demers, and J. Gehrke. Edge-weighted personalized pagerank: Breaking a decade-old performance barrier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2015.

[133] W. Xing and A. Ghorbani. Weighted pagerank algorithm. In *IEEE Annual Conference on Communication Networks and Services Research (CNSR)*, 2004.

[134] B. Yan and P. Sarkar. Covariate regularized community detection in sparse graphs. *Journal of the American Statistical Association*, 116(534):734–745, 2021.

[135] J. Yang, J. McAuley, and J. Leskovec. Community detection in networks with node attributes. In *IEEE International Conference on Data Mining (ICDM)*, 2013.

[136] S. Yang and K. Fountoulakis. Weighted flow diffusion for local graph clustering with node attributes: an algorithm and statistical guarantees. In *International Conference on Machine Learning (ICML)*, 2023.

[137] S. Yang, P. Senapati, D. Wang, C. T. Bauch, and K. Fountoulakis. Targeted pandemic containment through identifying local contact network bottlenecks. *PLOS Computational Biology*, 17(8):1–27, 8 2021.

[138] J. R. Yaros and T. Imielinski. Imbalanced hypergraph partitioning and improvements for consensus clustering. In *International Conference on Tools with Artificial Intelligence*, 2013.

[139] M. Yashtini. On the global convergence rate of the gradient descent method for functions with hölder continuous gradients. *Optimization Letters*, 10(6):1361–1370, 2016.

[140] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich. Local higher-order graph clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2017.

[141] C. Zhe, A. Sun, and X. Xiao. Community detection on large complex attribute network. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, 2019.

[142] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2003.

[143] D. Zhou, S. Zhang, M. Y. Yildirim, S. Alcorn, H. Tong, H. Davulcu, and J. He. A local algorithm for structure-preserving graph cut. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2017.

[144] J. Y. Zien, M. D. F. Schlag, and P. K. Chan. Multilevel spectral hypergraph partitioning with arbitrary vertex sizes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(9):1389–1399, 1999.