Analysis of Distance Functions in Graphs

Dissertation
zur Erlangung des akademischen Grades
Dr. rer. nat
an der Fakultät für Mathematik, Informatik und Naturwissenschaften
der Universität Hamburg

eingereicht beim Fachbereich Informatik von

Morteza Alamgir

aus Maragheh (Iran)

March 2014

Gutachterinnen/Gutachter:

Prof. Dr. Ulrike von Luxburg

Prof. Dr. Matthias Rarey

Prof. Marco Saerens

Tag der Disputation: 15. Juli 2014

Zusammenfassung

Viele Algorithmen des maschinellen Lernens verwenden Graphen, um die Beziehungen zwischen den Datenpunkten zu modellieren. Einer der interessantesten Aspekte für solche Algorithmen ist der des Abstands zwischen den Knoten. Es gibt verschiedene Abstandsfunktionen auf der Menge der Knoten eines Graphen, die jeweils unterschiedliche Eigenschaften des zugrunde liegenden Graphen reflektieren. Der erste Teil dieser Dissertation charakterisiert Eigenschaften zweier solcher globalen Abstandsfunktionen in Graphen:

- Kürzeste Pfad-Distanz: Das Verhalten der kürzesten Pfad-Distanz hängt davon ab, wie wir unseren Graphen aus den Datenpunkten konstruieren. Ich zeige, dass in ungewichteten k-Nächster-Nachbar-Graphen die kürzeste Pfad-Distanz gegen eine "unangenehme" Abstandsfunktion konvergiert, deren Eigenschaften sich nachteilig auf einige Ziele des maschinellen Lernens auswirken können.
- p-resistance-Distanz: Die p-resistance-Distanz ist eine Verallgemeinerung der resistance-Distanz und enthält auch noch weitere Abstandsfunktionen als Sonderfälle. Ich behandle die Konvergenz der p-resistance-Distanz in großen geometrischen Graphen und zeige, dass ein interessanter Phasenübergang stattfindet: Es gibt zwei kritische Schwellenwerte p^* und p^{**}, sodass für $p < p^*$ die p-resistance von bedeutenden globalen Eigenschaften des Graphen bestimmt wird, während sie für $p > p^{**}$ nur von trivialen lokalen Größen abhängt und keinerlei nützliche Information enthält.

Der zweite Teil dieser Dissertation befasst sich mit lokalen Abständen in Graphen. Lokales Clustering und Friend-Recommendation werden hier abgedeckt.

- Lokales Clustering ist die Aufgabe, einen "stark" verbundenen Cluster um einen Punkt hohen Interesses zu finden. Ich schlage ein neues Random-Walk-Modell für lokales Clustering vor, bestehend aus mehreren "Agenten", welche durch Seile miteindander verbunden sind. Alle Agenten können sich unabhängig voneinander bewegen, ihre Abstände zueinander werden aber durch die Seile zwischen ihnen begrenzt. Die entscheidende Einsicht dabei ist, dass es für mehrere Agenten schwieriger ist, gleichzeitig über den "Bottleneck" eines Graphen zu wandern, als dies für nur einen Agenten der Fall ist.
- Lokale Abstände werden für das Empfehlen von neuen Freunden in sozialen Netzwerken verwendet. Ich schlage dabei eine neue Distanzfunktion zwischen den Mitgliedern des Netzwerks vor, sodass zeitbezogene Daten des Netzwerks für die Empfehlung genützt werden.

Der dritte Teil meiner Dissertation ist dem Problem des "Downsamplings" großer Graphen gewidmet. Ziel dabei ist es, eine kleinere "Version" eines Graphen zu erzeugen, die leichter zu verarbeiten und zu visualisieren ist. Ich stelle in diesem Teil ein neues Verfahren und seine gründliche statistische Analyse vor. Resultat dieses Verfahrens ist ein verkleinerter Graph, der nachweislich viele Eigenschaften des ursprünglichen Graphen erbt.

Abstract

Many machine learning algorithms use graphs to model relations between data points. One of the main objects of interest for such algorithms is the distance between vertices. There are several distance functions defined between graph vertices, each reflecting different properties of the underlying graph. The first part of this thesis characterizes properties of global distance functions in graphs:

- Shortest path distance: The behavior of the shortest path distance depends on how we construct our graph from the data points. I show that in unweighted k-nearest neighbor graphs, the shortest path distance converges to an unpleasant distance function whose properties are detrimental to some machine learning problems.
- p-resistance distance: The p-resistance distance is a generalization of the resistance distance and contains several other distances as its special cases. I study the convergence of the p-resistance distance in large geometric graphs and show that an interesting phase transition takes place. There exist two critical thresholds p^* and p^{**} such that if $p < p^*$, then the p-resistance depends on meaningful global properties of the graph, whereas if $p > p^{**}$, it only depends on trivial local quantities and does not convey any useful information.

The second part of this thesis deals with local distances in graphs. Local clustering and friend recommendation are the topics covered in this part.

- Local clustering is the task of finding a highly connected cluster around a vertex of interest. I propose a new random walk model for local clustering, consisting of several "agents" connected by ropes. All agents move independently but their distances are constrained by the ropes between them. The main insight is that for several agents it is harder to simultaneously travel over the bottleneck of a graph than for just one agent.
- Local distances are used for recommending new friends in social networks. Here, I propose a new distance between members of the network that exploits the temporal data in the friendship network to recommend new friends.

The third part of my thesis is devoted to the problem of downsampling massive graphs. The goal of downsampling is to produce a smaller "version" of a given graph, which would be easier to process and visualize. Here, a new method is proposed and followed by its thorough statistical analysis. The output of this method is a downsampled graph that provably inherits many properties of the original graph.

Acknowledgements

First and foremost, I would like to express my gratitude to Prof. Ulrike von Luxburg for all her supports during my Phd time. This thesis would not have been possible without her help, knowledge and patience. She gave me the freedom to choose my own lines of research while always giving the best advice on how to proceed.

I am especially thankful to Bernhard Schölkopf for giving me the opportunity to do my thesis in an excellent research environment. I am appreciated to his advice and supports when I needed it. I would also like to thank the members of AGBS, for generously sharing their ideas and giving advice. In particular I would like to thank Suvrit Sra for pleasant discussions on diverse topics in linear algebra.

I thank Yasemin Altun, Moritz Grosse-Wentrup and Gábor Lugosi for fruitful collaborations, and Alexandros Karatzoglou for a joyful experience at Telefonica Research. A special thanks goes to Yoshikazu Terada for reading parts of the thesis and giving helpful comments.

Finally I would like to thank my family. Above all, my wife for her personal support, love and patience all the time, and my parents for their kind support and encouragement for pursuing my study.

Contents

Chapter 1

Introduction

A distance function describes the dissimilarity between two objects by a number. Devising a good distance function is a core task in many machine learning problems. The main goal of this thesis is to study distance functions in graphs, specifically the ones that often appear in machine learning problems. My focus is on properties of distance functions in large graphs:

- The convergence of distance functions in graphs: I am interested in the behavior of a distance function in very large graphs. Assume that our vertices are samples from a domain $\mathcal{X}$. Can we relate the graph distance between two vertices to a quantity in $\mathcal{X}$?
- Downsampling geometric graphs: Given a massive graph, a downsampling algorithm produces a smaller graph which "looks like" the original one. The hope is that visualizing or applying a machine learning algorithm on the smaller graph and on the original graph would produce similar results.
- Local distances in graphs: A local distance is a distance defined between "nearby" vertices. Local distance functions play an important role in applications such as local clustering and link prediction. In these applications, the underlying graphs usually have millions of nodes, and it is neither needed nor practical to compute all pairwise distances.

In the following, I briefly introduce the key concepts "graph" and "distance function in a graph" in the context of machine learning. Then an overview of the contents and results follows.

1.1 Graphs in machine learning

A graph represents a set of objects by vertices and the pairwise relations between these objects by links. In weighted graphs, these links carry some information about the strength of the relation. In machine learning problems, it is common to represent the sample points by vertices and the "similarity" between samples by weighted edges between vertices. This representation results in a graph built on the sample points. Techniques such as spectral clustering, semi-supervised learning and non-linear dimensionality reduction deal with such graphs.

From a mathematical point of view, these graphs are usually modeled as random geometric graphs: we assume that vertices are drawn i.i.d. from a density p on $\mathbb{R}^d$ and edges are between vertices that are "close". The closeness is usually measured by Euclidean distance (e.g., in *ϵ-graphs*), or by being among nearest neighbors of each other (e.g., in *kNN-graphs*). As an example, Figure 1.1b illustrates a kNN-graph ($k = 7$) built on 100 random sample points drawn i.i.d. from a density p. The underlying density p is depicted in Figure 1.1a.

Graphs are also used to represent complex structures such as social networks. In this type of graphs, both nodes and edges may carry some data. In social networks, nodes represent members of the network and we can assign properties such as age, sex and job to each node. Edges also carry some information such as the start time of the friendship, and the type of the connection: if they are colleagues, schoolmates or friends. Even when we have no access to such extra data, the graph structure itself carries valuable information about the network.

1.2 Distance functions in graphs

In machine learning, edges of a graph usually represent local information, e.g., distances between very "close" vertices. In problems such as classification and clustering, we are often interested in comparing objects that are "far" from each other. This comparison uses a distance function defined between vertices of the graph. A graph distance function represents the farness between two vertices by a real number. Imagine the graph vertices as cities and edges as roads between them. We expect a large distance between two cities (vertices) when it takes long to "travel" between them. Traveling along the shortest path between two cities leads to the shortest path distance. The resistance distance can be interpreted as traveling along random roads.

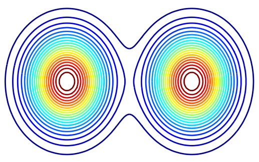
If we have a distance function that could assign the edge weights in building the graph, then why do we need the graph? Why not to use the same function for finding distances between far objects?

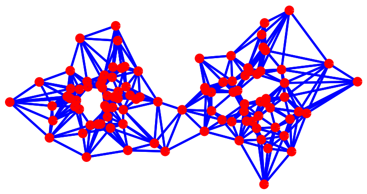
From local dissimilarities to global distances: Sometimes, it is only easy to assess distances between very similar objects. As an example, consider a set of hand-written digits, where digits are 16×16 grayscale images represented by vectors in $\mathbb{R}^{256}$. Samples such as and [1] are very similar and their distances can be measured by the Euclidean distance between them. However, samples such as have a very large Euclidean distance to and the Euclidean distance is not able to capture the similarity between these hand-written digits. Therefore, we build a graph on these samples. Figure 1.1c depicts an example graph built on samples of digits 1 and 7. After building the graph, we use a graph distance function for finding global distances between samples.

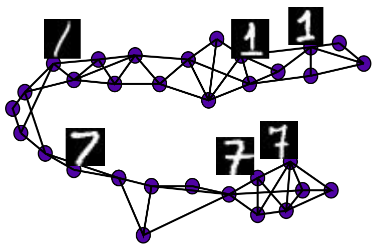
Chapters 2 and 3 explicitly deal with two global distance functions, namely the shortest path distance and the family of p-resistance distances.

Large sample analysis of graph distances: It is natural to ask about the quality of a graph distance function. Which distance function is "good" for our data? There is no single correct answer for this question and the performance of distance functions in graphs are usually very application dependent.

[1]Images are taken from MNIST dataset (LeCun et al., 1998)


(a) A contour plot of the underlying density p in $\mathbb{R}^2$.


(b) A kNN graph built on 100 random samples drawn i.i.d. from p.


(c) A neighborhood graph built on samples from hand-written digits 1 and 7.

But what we can do before having access to our data is to broaden our knowledge about properties of different distance functions. Examples of information that we can gather about a distance function are: Is the distance a metric? If yes, is it an Euclidean metric? When there are many paths between two vertices, do we get a shorter distance? Are the distances between points in the same "cluster" larger than the distances between points in different clusters?

Another important property of a graph distance is its applicability in large graphs: Can we compute it efficiently? How does the distance behave if the underlying graph gets very large? Does it converge to a "limit distance"?

For example, we know the answer for the shortest path distance in a specific family of graphs. In a kNN or ε-graph where the vertices are sampled from a uniform density, the shortest path distance converges to the underlying Euclidean distance (Tenenbaum et al., 2000).

To analyze the behavior of a distance function in large graphs, we need to fix a model for the underlying graph. In this thesis, I mainly work with the random geometric graph model which fits well to many machine learning scenarios. To make the chapters independent, the formal setting of each chapter is given inside that chapter.

1.3 Overview of the results

Here, I outline the topics discussed in each chapter and the results obtained:

Part I: Statistical Analysis of Global Distances in Random Geometric Graphs

In the first part, I study the convergence of the shortest path distance and the family of p-resistance distances. The underlying graph model is the random geometric graph model.

Chapter 2 The graph shortest path distance is the most fundamental distance function between vertices of a graph. The main goal of this chapter is to study the convergence of the shortest path distance in random geometric graphs. There are two parameters involved in determining the limit of the shortest path distance: the graph construction algorithm (e.g. kNN or ε-graph) and the edge weights. If the underlying density is uniform and the edges are unweighted or weighted by their Euclidean length, then the shortest path distance in both of kNN and ε-graphs will converge to the geodesic distance (Tenenbaum et al., 2000).

The story is different when the underlying density is not uniform. I study this case, and consider different weight functions. One of the results in this chapter is that the shortest path distance in unweighted kNN-graphs converges to a limit distance that does *not* conform to the natural intuition. More generally I study two questions:

1. **Limit distance:** We are given a function h that assigns weight $h(\|X_i - X_j\|)$ to the edge between X_i and X_j. How does the shortest path distance in this graph depend on h as $n \to \infty$?

2. **Weight assignment:** Given a distance measure D, how can we assign edge weights such that the shortest path distance in the graph converges to D?

For the first question, depending on properties of the function $h(x)$, the shortest path distance operates in two different regimes. The limit of the shortest path distance for one of these regimes is presented in this chapter. I also provide an answer to the second question without an explicit density estimation step. Some parts of this chapter have been published in Alamgir and von Luxburg (2012).

Chapter 3 The shortest path distance only looks at a single shortest path between two vertices and is not affected by the existence of several other paths between these vertices. The resistance distance or commute time is an important graph distance function which takes all the paths between two vertices into account. As a rule of thumb, more paths between u and v lead to a smaller distance between them. This distance can be interpreted as the electrical resistance between two vertices in an equivalent network, where each edge is replaced by a resistor. The resistance distance between u and v is also related to the expected time a random walk starting from u reaches v for the first time and returns.

The resistance distance is used in many machine learning applications, supposedly having the following property: vertices in the same cluster have small distances, as there are many short paths between them. Vertices in different clusters have a large resistance distance, as there are fewer paths connecting them. von Luxburg et al. (2010) proved that as the number of vertices increases, this property does not hold any more and the resistance distance converges to a meaningless limit function.

The main object under study in Chapter 3 is the family of p-resistances, which is a generalization of the standard resistance distance. First, I show that the family of p-resistances contains several special cases such as the shortest path distance, the standard resistance distance and the inverse min-cut. Second, I study the behavior of p-resistances in random geometric graphs and show that there are two completely different regimes of behavior. There exist critical thresholds p^* and p^{**} such that if $p < p^*$, the p-resistances convey useful information about the global topology of the data, whereas for $p > p^{**}$ the p-resistance distances approximate a limit that does not convey any useful information. The values of the critical thresholds are explicitly computed. The main results of this chapter have been published in Alamgir and von Luxburg (2011).

Chapter 3 ends with a short excursion to random walks, hitting times, the resistance distance and related concepts like Poisson kernels and Green functions. The resistance distance is discussed from a more general point of view: as a solution of a differential equation. This differential equation has two interpretations: an electrical network interpretation and a random walk interpretation. The p-resistance also fits into a similar framework: as a solution of a non-linear differential equation. Related concepts like p-harmonic functions are also discussed in this excursion.

Part II: Local Distances in Graphs

In massive graphs with millions of nodes, we cannot afford to compute global distances. However, some applications only need distance measures between nearby vertices. The next two chapters propose local similarity measures for two such applications: local clustering and friend recommendation.

Chapter 4 Graph clustering is the task of finding "clusters" or communities in the graph such that the vertices in the same cluster are highly connected to each other, and sparsely connected to vertices in other clusters. In real world applications with massive graphs, it is infeasible to apply usual clustering approaches on the whole graph.

A promising alternative is represented by the class of local clustering algorithms. Here, the goal is not to find a global clustering of the whole graph, but to find "the community" of a particular vertex of interest. For example, in social network analysis one might want to investigate the community a particular person belongs to.

In Chapter 4, a multi-agent random walk (MARW) is proposed for local clustering. Consider a "agents", each agent moving like an individual random walk on the graph. Any two agents are tied together by a "rope" of length l. The agents are constrained such that the distance between each agent to others is smaller than l. A vertex v belongs to a local cluster around u if it can be reached from u by MARW in few steps with high probability. I study the effect of the parameters a, l on the behavior of MARW and provide a bound on the mixing time of MARW. I present several theoretical approaches that show that in a MARW with a agents and a small rope length l, the probability of transitioning between different clusters decreases with the power of a. All results in this chapter have been published in Alamgir and von Luxburg (2010).

Chapter 5 Online Social Networks (OSN) aim at facilitating social relationships and friendships among users. Accurately recommending new friends to users is a vital task for the growth of the network. The friend recommendation task can be cast as a link prediction problem on the social graph. In Chapter 5, I propose a novel local similarity function for computing the social similarity between users. The similarity function takes the temporal information of the edge creation process into account. The proposed similarity measure can be interpreted as a local approximation of the diffusion distance kernel. The similarity function can be as easily computed as standard local measures such as the number of common neighbors, and is significantly more efficient in predicting future friendships. Experiments on industry scale data from a real world OSN with 12 million users show that this method significantly outperforms alternative link prediction methods. Some parts of this chapter have been written during my internship at Telefonica Research (Barcelona) under the supervision of Alexandros Karatzoglou.

Part III: Downsampling Random Geometric Graphs

Given a large neighborhood graph G, we would like to downsample it to some smaller graph G' with much fewer vertices. The ultimate goal in downsampling is to find a procedure that reduces the size of the graph, but at the same time keeps invariant those properties that are essential for our application. For example, in visualization the downsampled graph should "look like" the original one. In community detection, two vertices in the downsampled graph should belong to the same "community" whenever they are in the same community in G. Having a smaller version of a graph makes it possible to apply expensive visualization or clustering algorithms on it.

Chapter 6 We expect a downsampled graph to be "similar" to the original graph. In this chapter, a notion of similarity is defined between a neighborhood graph and its downsampled version, which is called *geometry-preserving* downsampling. A geometry-preserving downsampled graph keeps many basic properties of the original graph such as the shortest path distances between vertices intact.

The second step is to propose a geometry-preserving downsampling procedure. Graph downsampling consists of two steps: vertex selection and edge construction. I study a downsampling procedure that is based on a uniform subsample of vertices, and edges are connected based on shortest path distances in the original graph. I prove that this procedure is geometry-preserving when it is applied to random geometric graphs. I also show that some other popular downsampling algorithms are not geometry-preserving. The results in this chapter have not been published yet.

Chapter 7 This chapter provides an alternative method of selecting vertices for graph downsampling. Intuitively, it tries to select the vertices such that they approximately form a grid. The advantage of such a sample set is that it leads to a very sparse downsampled graph. This method is interesting on its own, and can be used for vector quantization.

The method is conceptually simple: construct an unweighted k-nearest neighbor graph on the sample points and use the k-mediod algorithm with respect to the shortest path distance on this graph for selecting the samples. Quite surprisingly, we do not even need to have access to the coordinates of the vertices or to their Euclidean distances. The algorithm works as soon as we know who are the k-nearest neighbors of each vertex for a proper k. The major part of this chapter is devoted to a thorough statistical analysis of the proposed algorithm.

1.4 List of published papers

Parts of this thesis have been published in the following papers:

Morteza Alamgir, Ulrike von Luxburg and Gábor Lugosi. Density-preserving quantization with application to graph downsampling, *Proceedings of the 27th Annual Conference on Learning Theory (COLT)*, pages 543–559, 2014.

Morteza Alamgir and Ulrike von Luxburg. Shortest path distance in random k-nearest neighbor graphs. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML)*, pages 1031–1038, 2012.

Morteza Alamgir and Ulrike von Luxburg. Phase transition in the family of p-resistances. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS) 24*, pages 379-–387, 2011.

Morteza Alamgir and Ulrike von Luxburg: Multi-agent random walks for local clustering on graphs. In Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu, editors, *The 10th IEEE International Conference on Data Mining (ICDM)*, pages 18—27, 2010.

During my time as a Phd student, I also published the following paper, whose contents have not been added to the final thesis:

Morteza Alamgir, Moritz Grosse-Wentrup and Yasemin Altun: Multitask Learning for Brain-Computer Interfaces. In Yee Whye Teh and D. Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 17-–24, 2010.

Part I

Statistical Analysis of Global Distances in Random Geometric Graphs

Chapter 2

Shortest path distance in random geometric graphs

2.1 Introduction

The shortest path distance is the most fundamental distance function between vertices in a graph, and it is widely used in computer science and machine learning. In this chapter we study the convergence of the shortest path distance and the geometry induced by it in randomly generated geometric graphs such as k-nearest neighbor graphs.

Consider a neighborhood graph G built from an i.i.d. sample $X_1, ..., X_n$ drawn according to some density p on $\mathcal{X} \subset \mathbb{R}^d$ (for exact definitions see Section 2.2). Assume that the sample size n goes to infinity. Two questions arise about the behavior of the shortest path distance between fixed points in this graph:

1. **Weight assignment:** Given a distance measure D on $\mathcal{X}$, how can we assign edge weights such that the shortest path distance in the graph converges to D?
2. **Limit distance:** Given a function h that assigns weights of the form $h(\|X_i - X_j\|)$ to edges in G, what is the limit of the shortest path distance in this weighted graph as $n \to \infty$?

The first question has already been studied in some special cases. Tenenbaum et al. (2000) discuss the case of ε- and kNN graphs when p is *uniform* and D is the Euclidean distance. It is enough to set the weight of the edge between X_i and X_j to $\|X_i - X_j\|$. Then the shortest path distances between vertices converges to the Euclidean distance between them (see Figure 2.1 for an illustration). Sajama and Orlitsky (2005) extend these results to ε-graphs from a general density p by introducing edge weights that depend on an explicit estimate of the underlying density. Hwang et al. (2012) consider completely connected graphs whose vertices come from a general density p and whose edge weights are powers of distances.

There is little work regarding the second question. Tenenbaum et al. (2000) answer the question for a very special case with $h(x) = x$ and uniform p. Hwang et al. (2012) study the case $h(x) = x^a$, $a > 1$ for arbitrary density p.

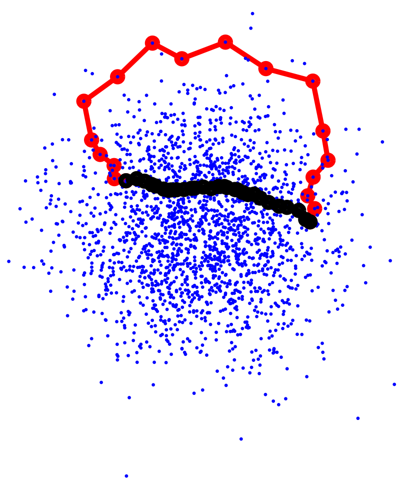


Figure 2.1: The shortest path based on an unweighted (red) and Euclidean weighted (black) kNN graph. For the sake of clarity, the graph edges are not plotted in the figure.

We have a more general point of view. In Section 2.4 we show that depending on properties of the function $h(x)$, the shortest path distance operates in different regimes, and we find the limit of the shortest path distance for particular function classes of $h(x)$. Our method also reveals a direct way to answer the first question without doing an explicit density estimation. An interesting special case is the unweighted kNN graph, which corresponds to the constant weight function $h(x) = 1$. We show that the shortest path distance on unweighted kNN-graphs converges to a limit distance on $\mathcal{X}$ that does *not* conform to the natural intuition and induces a geometry on $\mathcal{X}$ that can be detrimental for machine learning applications.

Our results have implications for many machine learning algorithms, see Section 2.5 for more discussion. (1) The shortest paths based on unweighted kNN graphs prefer to go through low density regions, and they even accept large detours if this avoids passing through high density regions (see Figure 2.1 for an illustration). This is exactly the opposite of what we would like to achieve in most applications. (2) For manifold learning algorithms like Isomap, unweighted kNN graphs introduce a fundamental bias that leads to huge distortions in the estimated manifold structure (see Figure 2.3 for an illustration).

(3) In the area of semi-supervised learning, a standard approach is to construct a graph on the sample points, then compute a distance between vertices of the graph, and finally use a standard distance-based classifier to label the unlabeled points (e.g., Sajama and Orlitsky, 2005 and Bijral et al., 2011). The crucial property exploited in this approach is that distances between points should be small if they are in the same high-density region. Shortest path distances in unweighted kNN graphs and their limit distances do exactly the opposite, so they can be misleading for this approach.

2.2 Basic definitions

Consider a closed, connected subset $\mathcal{X} \subseteq \mathbb{R}^d$ that is endowed with a density function p with respect to the Lebesgue measure. For the ease of presentation we assume for the rest of this chapter that the density p is Lipschitz continuous with Lipschitz constant L and bounded away from 0 and infinity

$$0 < p_{\min} \leq p(x) \leq p_{\max} < \infty.$$

We will consider different metrics on $\mathcal{X}$. A ball with respect to a particular metric D in $\mathcal{X}$ will be written as $B(x, r, D) := \{y \in \mathcal{X} \mid D(x, y) \leq r\}$. We denote the Euclidean volume of the unit ball in $\mathbb{R}^d$ by η_d.

Assume the finite dataset $X_1, ..., X_n$ has been drawn i.i.d according to p. We build a geometric graph $G = (V, E)$ that has the data points as vertices and connects vertices that are close. Specifically, for the kNN graph we connect X_i with X_j if X_i is among the k nearest neighbors of X_j or vice versa. For the ϵ-graph, we connect X_i and X_j whenever their Euclidean distance satisfies $\|X_i - X_j\| \leq \epsilon$. In this chapter, all graphs are undirected, but might carry edge weights $w_{ij} \geq 0$. In unweighted graphs, we define the length of a path by its number of edges, in weighted graphs we define the length of a path by the sum of the edge weights along the path. In both cases, the shortest path (shortest path) distance $D_{sp}(x, y)$ between two vertices $x, y \in V$ is the length of the shortest path connecting them.

Let f be a positive continuous scalar function defined on $\mathcal{X}$. For a given path γ in $\mathcal{X}$ that connects x with y and is parameterized by t, we define the f-length of the path as

$$D_{f,\gamma} = \int_\gamma f(\gamma(t))|\gamma'(t)|dt.$$

This expression is also known as the *line integral* along γ with respect to f. The f-geodesic path between x and y is the path with minimum f-length.

The f-length of the geodesic path is called the f-distance between x and y. We denote it by $D_f(x, y)$. If $f(x)$ is a function of the density p at x, then the f-distance is sometimes called a density based distance (Sajama and Orlitsky, 2005). The f-distance on $\mathcal{X}$ is a metric, and in particular it satisfies the triangle inequality. Another useful property is that for a point u on the f-geodesic path between x and y we have

$$D_f(x, y) = D_f(x, u) + D_f(u, y).$$

We introduce a shorthand notation for the f-distance with $f(x) = p(x)^{1/d}$ and call it PD-distance, denoted by D_{PD}. The function f determines the behavior of the f-distance. When $f(x)$ is a monotonically decreasing function of density $p(x)$, passing through a high density region will cost less than passing through a low density region. It works the other way round when f is a monotonically increasing function of density. A constant function does not impose any preference between low and high density regions, so the Riemannian geodesic path would be the preferred one.

The main purpose of this chapter is to study the relationship between the shortest path distance in various geometric graphs and particular f-distances on $\mathcal{X}$. For example, in Section 2.3 we show that the shortest path distance in unweighted kNN graphs converges to the PD-distance. In the rest of the chapter, all statements refer to points x and y in the interior of $\mathcal{X}$ such that their f-geodesic path is bounded away from the boundary of $\mathcal{X}$.

2.3 Shortest paths in unweighted graphs

In this section we study the behavior of the shortest path distance in the family of *unweighted* kNN graphs. We show that the rescaled graph shortest path distance converges to the PD-distance in the original space $\mathcal{X}$.

Theorem 2.1 (Shortest path distance limit in unweighted kNN graphs) *Consider the unweighted kNN graph G_n based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ from the density p. Choose $\lambda < 0.2$ such that*

$$\lambda \geq \frac{2^{1/d} L}{\eta_d^{1/d} p_{\min}^{1+1/d}} \Big(\frac{k}{n}\Big)^{1/d}.$$

Fix two points $x = X_i$ and $y = X_j$. Let $k' < k/8^{d+1}$ be a positive real. Then with probability at least

$$1 - 2ne^{-\lambda^2 k/6} - 2^{d+1} n e^{-k'/6}/k'$$

we have

$$c_1 D_{\text{PD}}(x,y) \leq \big(\frac{k}{\eta_d n}\big)^{1/d} D_{sp}(x,y) \leq c_2 D_{\text{PD}}(x,y) + \big(\frac{k}{\eta_d n}\big)^{1/d},$$

where

$$c_1 = \frac{(1-\lambda)^{2/d}}{(1+\lambda)^{1/d}} \ , \ c_2 = \frac{1}{\frac{(1-\lambda)^{1/d}}{(1+\lambda)^{2/d}} - 4(2k'/k)^{1/d}}.$$

Moreover, if $n \to \infty$, $k/n \to 0$, $k/\log(n) \to \infty$ and $k \leq \sqrt{n}$, we can set k' and λ such that the probability converges to 1 and $(k/(\eta_d n))^{1/d} D_{sp}(x,y)$ converges to $D_{\text{PD}}(x,y)$ almost surely.

The convergence conditions on n and k are the ones to be expected for random geometric graphs. The condition $k/\log(n) \to \infty$ is slightly stronger than the usual $k > c\log(n)$ condition. For k smaller than $\log(n)$, the graphs are not connected anyway (see e.g. Penrose, 1999) and are unsuitable for machine learning applications. The upper bound condition $k \leq \sqrt{n}$ holds in many machine learning applications, as we usually choose $k \sim \log(n)$. However, we can relax this condition to $k \leq n/\log(n)$ by adding a slightly stronger lower-bound assumption $k \geq \log(n)^{1+\alpha}$ for a fixed $\alpha > 0$.

Before proving Theorem 2.1, we need to state a couple of propositions and lemmas. We start by introducing some ad-hoc notation:

Definition 2.2 (Connectivity parameters) *Consider a geometric graph based on a fixed set of points $X_1, ..., X_n \in \mathbb{R}^d$. Let r_{low} be a real number such that $D_f(X_i, X_j) \leq r_{low}$ implies that X_i is connected to X_j in the graph. Analogously, consider r_{up} to be a real number such that $D_f(X_i, X_j) \geq r_{up}$ implies that X_i is not connected to X_j in the graph.*

In random geometric graphs, it is desired that every vertex is connected to all "nearby" vertices and not connected to any vertex which is "far". "Near" and "far" are measured with respect to a distance function D_f. In an ε-graph, distances are the Euclidean distance and the lower and the upper bounds on the connectivity parameter are equal $r_{low} = r_{up} = \varepsilon$. In unweighted kNN-graphs, Proposition 2.6 shows that r_{low} converges with high probability to r_{up} when the underlying distance is the PD-distance.

Definition 2.3 (Dense sampling assumption) *Consider a graph G with connectivity parameters r_{low} and r_{up}. We say that it satisfies the dense sampling assumption if there exists an $\varsigma < r_{low}/4$ such that for all $x \in \mathcal{X}$ there exists a vertex y in the graph with $D_f(x,y) \leq \varsigma$.*

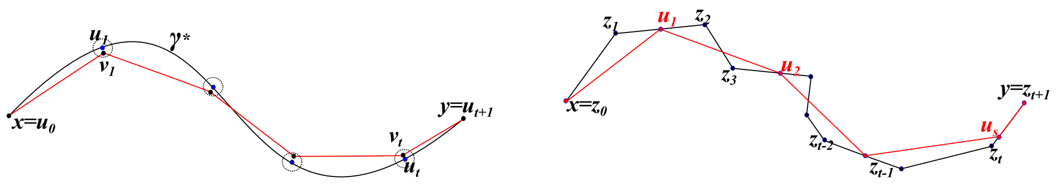


Figure 2.2: Path constructions in the proofs of Proposition 2.4 (left) and Theorem 2.13 (right).

The next proposition bounds the shortest path distance by the f-distance in random geometric graphs with connectivity parameters r_{low} and r_{up}.

Proposition 2.4 (Bounding D_{sp} by D_f) *Consider any unweighted geometric graph based on a fixed set $X_1, ..., X_n \in \mathcal{X} \subset \mathbb{R}^d$ that satisfies the dense sampling assumption (with respect to the f-distance with general f). Fix two vertices x and y of the graph and set*

$$e_1 = (r_{low} - 2\varsigma)/r_{up} \;\; , \;\; e_2 = r_{low} - 2\varsigma.$$

Then the following statement holds:

$$e_1 D_f(x, y) \leq e_2 D_{sp}(x, y) \leq D_f(x, y) + e_2.$$

Proof. *Right hand side.* Consider the f-geodesic path $\gamma^*_{x,y}$ connecting x to y. Divide $\gamma^*_{x,y}$ to segments by $u_0 = x, u_1, ..., u_t, u_{t+1} = y$ such that $D_f(u_i, u_{i+1}) = r_{low} - 2\varsigma$ for $i = 0, ..., t-1$ and $D_f(u_t, u_{t+1}) \leq r_{low} - 2\varsigma$ (see Figure 2.2). Because of the dense sampling assumption, for all $i = 1, ..., t$ there exists a vertex v_i in the ball $B(u_i, \varsigma, D_f)$ and we have

$$\begin{aligned} D_f(v_i, u_i) &\leq \varsigma \\ D_f(u_i, u_{i+1}) &\leq r_{low} - 2\varsigma \\ D_f(u_{i+1}, v_{i+1}) &\leq \varsigma. \end{aligned}$$

Applying the triangle inequality gives $D_f(v_i, v_{i+1}) \leq r_{low}$, which shows that v_i and v_{i+1} are connected. By summing up along the path we get

$$\begin{aligned} (r_{low} - 2\varsigma)(D_{sp}(x, y) - 1) &\leq (r_{low} - 2\varsigma)t \\ &= \sum_{i=0}^{t-1} D_f(u_i, u_{i+1}) \overset{(a)}{\leq} D_f(x, y). \end{aligned}$$

In step (a) we use the simple fact that if u is on the f-geodesic path from x to y, then

$$D_f(x, y) = D_f(x, u) + D_f(u, y).$$

Left hand side. Assume that the graph shortest path between x and y consists of vertices $z_0 = x, z_1, ..., z_s = y$. By $D_f(z_i, z_{i+1}) \leq r_{up}$ we can write

$$\begin{aligned} (r_{low} - 2\varsigma) D_{sp}(x, y) &\geq \frac{r_{low} - 2\varsigma}{r_{up}} \sum_{i=0}^{s-1} D_f(z_i, z_{i+1}) \\ &\geq \frac{r_{low} - 2\varsigma}{r_{up}} D_f(x, y). \end{aligned}$$

□

The next lemma uses the Lipschitz continuity and boundedness of p to show that $p(x)^{1/d} \|x - y\|$ is a good approximation of $D_{\text{PD}}(x, y)$ in small intervals.

Lemma 2.5 (Approximating $D_{\mathbf{pd}}$ by Euclidean distance) *For two arbitrary points $x, y \in \mathcal{X}$, the* PD*-distance between them can be bounded by*

1. $p_{\min}^{1/d}\|x-y\| \le D_{\text{PD}}(x,y) \le p_{\max}^{1/d}\|x-y\|$.

Consider a fix $\lambda < 0.2$ and assume that $\|x-y\| \le p_{\min}\lambda/L$. Then we can approximate the density at y by the density at x:

2. $p(y)(1-\lambda) \le p(x) \le p(y)(1+\lambda)$.

We can also approximate $D_{\text{PD}}(x,y)$ by $p(x)^{1/d}\|x-y\|$:

3. $(1-\lambda)^{1/d}p(x)^{1/d}\|x-y\| \le D_{\text{PD}}(x,y) \le (1+\lambda)^{1/d}p(x)^{1/d}\|x-y\|$.

Proof. *Part (1).* The statement is a direct result from the boundedness of the density $p_{\min} \le p(x) \le p_{\max}$.

Part (2). If $\|x-y\| \le \delta$, by the Lipschitz continuity of the density p we have

$$|p(x)-p(y)| \le L\|x-y\| \le L\delta.$$

Setting $\delta = \lambda p_{\min}/L$ leads to the result.

Part (3). The previous part can be written as

$$(1-\lambda)^{1/d}p(x)^{1/d} \le p(y)^{1/d} \le (1+\lambda)^{1/d}p(x)^{1/d}.$$

Denote the PD-geodesic path between x and y by γ^* and the line segment connecting x to y by l. Using the definition of a geodesic path, we can write

$$\begin{aligned} \int_{\gamma^*} p(\gamma^*(t))^{1/d}|\gamma^*(t)'|dt &\le \int_l p(l(t))^{1/d}|l(t)'|dt \le \\ (1+\lambda)^{1/d}\int_l p(x)^{1/d}|l(t)'|dt &= (1+\lambda)^{1/d}p(x)^{1/d}\|x-y\|. \end{aligned}$$

For the left hand side, let $\bar{\gamma^*}$ be the restriction of γ^* to an Euclidean ball with radius $\|x-y\|$ around x. Then

$$\begin{aligned} \int_{\gamma^*} p(\gamma^*(t))^{1/d}|\gamma^*(t)'|dt &\ge \int_{\bar{\gamma^*}} p(\gamma^*(t))^{1/d}|\gamma^*(t)'|dt \\ &\ge (1-\lambda)^{1/d}\int_{\bar{\gamma^*}} p(x)^{1/d}|\gamma^*(t)'|dt \\ &\ge (1-\lambda)^{1/d}p(x)^{1/d}\|x-y\|. \end{aligned}$$

□

Now we are going to show how the quantities r_{low} and r_{up} introduced in Definition 2.2 can be bounded in random unweighted kNN graphs and how they are related to the metric D_{PD}. Define the kNN radii at vertex x as $R_k(x) = D_{\text{PD}}(x,y)$ and the approximated kNN radii at vertex x as $\hat{R}_k(x) = p(x)^{1/d}\|x-y\|$, where y is the k-nearest neighbor of x. The minimum and maximum values of kNN radii are defined as

$$R_k^{min} = \min_u R_k(u) \ \ , \ \ R_k^{max} = \max_u R_k(u).$$

Accordingly we define $\hat{R}_k^{min}$ and $\hat{R}_k^{max}$ for the approximate radii.

Proposition 2.6 (Bounding R_k^{min} and R_k^{max}) *Given $\lambda < 0.2$, define r_{low} and r_{up} as*

$$r_{low} := \Big(\frac{1-\lambda}{(1+\lambda)^2}\Big)^{1/d}\Big(\frac{k}{n\eta_d}\Big)^{1/d} \quad \text{and} \quad r_{up} := \Big(\frac{1+\lambda}{(1-\lambda)^2}\Big)^{1/d}\Big(\frac{k}{n\eta_d}\Big)^{1/d}.$$

Assume that for large enough n, $r_{up} \le \lambda p_{\min}^{1+1/d}/L$. Then

$$P\Big(R_k^{min} \le r_{low}\Big) \le n\exp(-\lambda^2 k/6)\ ,$$
$$P\Big(R_k^{max} \ge r_{up}\Big) \le n\exp(-\lambda^2 k/6).$$

Proof. Define radius $\hat{r}_{low}$ and $\hat{r}_{up}$ as

$$\hat{r}_{low} = \frac{r_{low}}{(1-\lambda)^{1/d}}\ \ ,\ \ \hat{r}_{up} = \frac{r_{up}}{(1+\lambda)^{1/d}}.$$

Bound on R_k^{min}: Consider an Euclidean ball $B_x := B(x, \hat{r}_{low}/p(x)^{1/d})$ with radius $\hat{r}_{low}/p(x)^{1/d}$ around a vertex x. Note that

$$\frac{\hat{r}_{low}}{p(x)^{1/d}} \le \frac{r_{low}}{p_{min}^{1/d}(1-\lambda)^{1/d}} \le \frac{r_{up}}{p_{min}^{1/d}} \le p_{\min}\lambda/L,$$

so we can bound the density of points in B_x by $(1+\lambda)p(x)$ using Lemma 2.5. Denote the probability mass of the ball by $\mu(B_x)$, which is bounded by

$$\begin{aligned}\mu(B_x) = \int_{B_x} p(s)ds \quad &\le \quad (1+\lambda)p(x)\int_{B_x} ds \\ &= \quad (1+\lambda)\hat{r}_{low}^d\eta_d =: \mu_{\max}.\end{aligned}$$

Observe that $\hat{R}_k(x) \le \hat{r}_{low}$ if and only if there are at least k data points in B_x. Let $Q \sim \mathit{Binomial}(n, \mu(B_x))$ and $S \sim \mathit{Binomial}(n, \mu_{\max})$. By the choice of $\hat{r}_{low}$, we have $E(S) = k/(1+\lambda)$ and

$$\begin{aligned}P\Big(\hat{R}_k(x) \le \hat{r}_{low}\Big) \quad &= \quad P\Big(Q \ge k\Big) \le P\Big(S \ge k\Big) \\ &= \quad P\Big(S \ge (1+\lambda)E(S)\Big).\end{aligned}$$

Apply a concentration inequality for binomial random variables (see Prop. 28 in von Luxburg et al., 2010) to get

$$\begin{aligned}P\Big(\hat{R}_k(x) \le \hat{r}_{low}\Big) &\le \exp\Big(\frac{-\lambda^2 k}{3(1+\lambda)}\Big) \\ &\le \exp(-\lambda^2 k/6).\end{aligned}$$

Now, we prove that $P(R_k(x) \leq r_{low}) \leq P(\hat{R}_k(x) \leq \hat{r}_{low})$. This can be done by showing that $R_k(x) \leq r_{low}$ always implies $\hat{R}_k(x) \leq \hat{r}_{low}$. Let y denote the k-nearest neighbor of x. From Part 1 of Lemma 2.5,

$$p_{\min}^{1/d} \|x - y\| \leq R_k(x) \leq r_{low},$$

which shows that

$$\|x - y\| \leq \frac{r_{low}}{p_{\min}^{1/d}} < \frac{r_{up}}{p_{\min}^{1/d}} \leq \frac{\lambda p_{\min}}{L}.$$

By Part 3 of Lemma 2.5,

$$(1-\lambda)^{1/d} p(x)^{1/d} \|x - y\| \leq R_k(x) \leq r_{low}.$$

Use the definition of $\hat{R}_k(x)$ to get

$$\hat{R}_k(x) = p(x)^{1/d}\|x - y\| \leq \frac{r_{low}}{(1-\lambda)^{1/d}} = \hat{r}_{low}.$$

At the end, applying a union bound leads us to the result

$$P\Big(R_k^{min} \leq r_{low}\Big) \leq P\Big(\exists i \ : \ R_k(X_i) \leq r_{low}\Big) \leq n \exp(-\lambda^2 k/6).$$

Bound on R_k^{max}: The proof is similar to the argument for bounding R_k^{min}. Again, consider a ball $\mathcal{B}_x$ with radius $\hat{r}_{up}/p(x)^{1/d}$ around a vertex x and note that $\hat{r}_{up}/p(x)^{1/d} \leq \lambda p_{\min}/L$, so

$$\begin{aligned} \mu(\mathcal{B}_x) = \int_{\mathcal{B}_x} p(s)ds \quad &\geq \quad (1-\lambda)p(x)\int_{B_x} ds \\ &= \quad (1-\lambda)\hat{r}_{up}^d \eta_d =: \mu_{\min}. \end{aligned}$$

Observe that $\hat{R}_k(x) \geq \hat{r}_{up}$ if and only if there are at most k data points in $\mathcal{B}_x$. Let $Q \sim Binomial(n, \mu(\mathcal{B}_x))$ and $S \sim Binomial(n, \mu_{\min})$. By the choice of $\hat{r}_{up}$ we have $E(S) = k/(1-\lambda)$. It follows that

$$\begin{aligned} P\Big(\hat{R}_k(x) \geq \hat{r}_{up}\Big) \quad &= \quad P\Big(Q \leq k\Big) \leq P\Big(S \leq k\Big) \\ &= \quad P\Big(S \leq (1-\lambda)E(S)\Big). \end{aligned}$$

Now we apply a concentration inequality for binomial random variables to get

$$\begin{aligned} P\Big(\hat{R}_k(x) \geq \hat{r}_{up}\Big) &\leq \exp\Big(\frac{-\lambda^2 k}{3(1-\lambda)}\Big) \\ &\leq \exp(-\lambda^2 k/6). \end{aligned}$$

We show that $P(R_k(x) \geq r_{up}) \leq P(\hat{R}_k(x) \geq \hat{r}_{up})$, which implies

$$P\Big(R_k(x) \geq r_{up}\Big) \leq \exp(-\lambda^2 k/6).$$

We prove this by showing that $R_k(x) \geq r_{up}$ results in $\hat{R}_k(x) \geq \hat{r}_{up}$. The proof is by contradiction: assume that $\hat{R}_k(x) < \hat{r}_{up}$. Replacing the definitions of $\hat{R}_k(x)$ and $\hat{r}_{up}$ gives

$$\hat{R}_k(x) = p(x)^{1/d}\|x-y\| < \hat{r}_{up} = \frac{r_{up}}{(1+\lambda)^{1/d}} \leq \frac{\lambda p_{\min}^{1+1/d}}{L},$$

where y is the k-nearest neighbor of x. Therefor, $\|x-y\| \leq \lambda p_{\min}/L$ and by Part 3 of Lemma 2.5,

$$R_k(x) = D_{\text{PD}}(x,y) \leq (1+\lambda)^{1/d} p(x)^{1/d}\|x-y\| < r_{up},$$

which contradicts with the assumption $R_k(x) \geq r_{up}$. At the end, we use a union bound to get

$$P\Big(R_k^{max} \geq r_{up}\Big) \leq P\Big(\exists i \;:\; R_k(X_i) \geq r_{up}\Big) \leq n \exp(-\lambda^2 k/6).$$

□

The following proposition shows how the sampling parameter ς can be chosen to satisfy the dense sampling assumption.

Lemma 2.7 (Sampling lemma) *Assume $X_1, ..., X_n \in \mathcal{X}$ are sampled i.i.d. from a probability distribution p. Let $k' > 0$ be a real number such that*

$$k' < \frac{0.2^d \eta_d p_{\min}^{d+1}}{L^d} n.$$

Then with probability at least $1 - 2^{d+1} n \exp(-k'/6)/k'$, for every $x \in \mathcal{X}$ exists a $v \in X_1, ..., X_n$ such that

$$D_{\text{PD}}(x,v) \leq 2\Big(\frac{2k'}{\eta_d n}\Big)^{1/d}.$$

Proof. To prove the lemma, we show that the following statement holds with probability at least $1 - 2^{d+1} n \exp(-k'/6)/k'$: There exist a $\big(\frac{2k'}{\eta_d n}\big)^{1/d}$-covering of $\mathcal{X}$ with respect to the PD-distance, such that each ball contains at least one sample point.

Choose $\lambda < 0.2$ such that

$$\lambda \geq \Big(\frac{1}{\eta_d p_{\min}^{d+1}}\Big)^{1/d} L\Big(\frac{k'}{n}\Big)^{1/d}.$$

Set $\varsigma_0 = \big(\frac{k'}{\eta_d n}\big)^{1/d}$. We prove that for every $x \in \mathcal{X}$, there exist a vertex v such that $p(x)^{1/d}\|x - v\| \leq \varsigma_0$. Then, $\varsigma_0/p(x)^{1/d} \leq \lambda p_{\min}/L$ and by Part 3 of Lemma 2.5

$$D_{\text{PD}}(x,v) \leq (1+\lambda)^{1/d} p(x)^{1/d}\|x-v\| \leq \Big(\frac{2k'}{\eta_d n}\Big)^{1/d},$$

which is our desired result.

The proof is based on the standard argument for bounding the covering number. We first construct a covering of $\mathcal{X}$ that consists of balls with approximately the same probability mass. The centers of the balls are chosen by an iterative procedure that ensures that no center is contained in any of the balls we have so far. We choose the radius $\varsigma_0/p(x)^{1/d}$ for the ball at point x and call it $B_p(x, \varsigma_0)$. The probability mass of this ball can be bounded by

$$\mathcal{V}(B_p(x, \varsigma_0)) \geq (1-\lambda)\varsigma_0^d \eta_d.$$

The smaller balls $B_p(x, (1-\lambda)^{1/d}\varsigma_0/2)$ are all disjoint. If not, consider two intersecting balls $B_p(x, (1-\lambda)^{1/d}\varsigma_0/2)$ and $B_p(y, (1-\lambda)^{1/d}\varsigma_0/2)$. Observe that

$$\frac{(1-\lambda)^{1/d}\varsigma_0}{2p(x)^{1/d}} + \frac{(1-\lambda)^{1/d}\varsigma_0}{2p(y)^{1/d}} \leq \frac{\varsigma_0}{p(x)^{1/d}},$$

which contradicts with how we chose the centers of balls. We can bound the total number of balls by

$$S \leq \frac{1}{\mathcal{V}(B_p(x, (1-\lambda)^{1/d}\varsigma_0/2))} \leq \frac{2^d}{\eta_d(1-\lambda)^2\varsigma_0^d} \leq \frac{2^{d+1}}{\eta_d \varsigma_0^d}.$$

Now we sample points from the underlying space and apply the same concentration inequality as above. We bound the probability that a ball $B_p(u, \varsigma_0)$ does not contain any sample point ("is empty") by

$$Pr(\text{Ball } i \text{ is empty}) \leq \exp(-n\varsigma_0^d \eta_d/6).$$

Rewriting and Substituting the value of ς_0 gives

$$\begin{aligned} Pr(\text{no ball is empty}) &\geq 1 - \sum_i Pr(\text{Ball i is empty}) \geq 1 - S \cdot e^{-n\varsigma_0^d \eta_d/6} \\ &\geq 1 - \frac{2^{d+1} n e^{-k'/6}}{k'}. \end{aligned}$$

□

Proof of Theorem 2.1. Set r_{low} and r_{up} as in Proposition 2.6. The assumption on λ ensures that $\hat{r}_{up} \leq \lambda p_{\min}^{1+1/d}/L$. It follows from Proposition 2.6 that the statements about r_{low} and r_{up} in Definition 2.2 both hold for G_n with probability at least $1 - 2n\exp(-\lambda^2 k/6)$. Set $\varsigma = 2\big(\frac{2k'}{\eta_d n}\big)^{1/d}$ as in Lemma 6.5. By the choice of k', we have $r_{low} > 4\varsigma$. Lemma 2.7 shows that the sampling assumption holds in G_n for the selected ς with probability at least $1 - 2^{d+1} n \exp(-k'/6)/k'$. Using Proposition 2.4 completes the first part of the theorem.

For the convergence, set $k' = 18\log(n)$ and $\lambda = (\log(n)/k)^{1/d}$. If n is large enough and $k < \sqrt{n}$, λ will satisfy the lower bound condition

$$\lambda = \big(\frac{\log(n)}{k}\big)^{1/d} \geq \frac{2^{1/d} L}{\eta_d^{1/d} p_{\min}^{1+1/d}} \Big(\frac{k}{n}\Big)^{1/d}.$$

As $\frac{k}{\log(n)}$ goes to infinity, λ converges to 0 and $k'/k \to 0$. Therefor c_1 and c_2 also converge to 1. For the selection of k', λ and large enough n, $\lambda^2 k$ is larger than $18\log(n)$ and

$$1 - 2n\exp(-\lambda^2 k/6) - \frac{2^{d+1}}{k'} n \exp(-k'/6) > 1 - \frac{2^{d+2}}{n^2}.$$

Applying the Borel-Cantelli lemma results in the almost sure convergence of $(\frac{k}{\eta_d n})^{1/d} D_{sp}(x,y)$ to $D_{\text{PD}}(x,y)$. $\square$

2.4 Shortest paths in weighted graphs

In this section we discuss both questions raised in Section 2.1.

1. **Weight assignment:** Given a distance measure D on $\mathcal{X}$, how can we assign edge weights such that the shortest path distance in the graph converges to D?
2. **Limit distance:** Given a function h that assigns weights of the form $h(\|X_i - X_j\|)$ to edges in G, what is the limit of the shortest path distance in this weighted graph as $n \to \infty$?

We extend our results from the previous section to weighted kNN graphs and ε-graphs.

2.4.1 Weight assignment problem

Consider a graph based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ from the density p. We are given a positive scalar function f which is only a function of the density: $f(x) = \tilde{f}(p(x))$. We want to assign edge weights such that the graph shortest path distance converges to the f-distance in $\mathcal{X}$.

It is well known that the f-length of a curve $\gamma : [a,b] \to \mathcal{X}$ can be approximated by a Riemann sum over a partition of $[a,b]$ to subintervals $[x_i, x_{i+1}]$:

$$\hat{D}_{f,\gamma} = \sum_i f\Big(\frac{\gamma(x_i) + \gamma(x_{i+1})}{2}\Big) \|\gamma(x_i) - \gamma(x_{i+1})\|.$$

As the partition gets finer, the approximation $\hat{D}_{f,\gamma}$ converges to $D_{f,\gamma}$ (cf. Chapter 3 of Gamelin, 2007). This suggests using weight

$$w_{ij} = \tilde{f}\Big(p(\frac{X_i + X_j}{2})\Big) \|X_i - X_j\|$$

for the edge between X_i and X_j. However the underlying density $p(x)$ is not known in many machine learning applications. Sajama and Orlitsky (2005) already proved that the plug-in approach using a kernel density estimator $\hat{p}(x)$ for $p(x)$ will lead to the convergence of the shortest path distance to f-distance in ε-graphs. Our next result shows how to choose edge weights in kNN graphs without estimating the density. It is a corollary from a theorem that will be presented in Section 2.4.2.

We use a notational convention to simplify our arguments and hide approximation factors that will eventually go to 1 as the sample size goes to infinity. We say that f is approximately larger than g ($f \succcurlyeq_\lambda g$) if there exists a function $e(\lambda)$ such that $f \geq e(\lambda) g$ and $e(\lambda) \to 1$ as $n \to \infty$ and $\lambda \to 0$. The symbol $\preccurlyeq_\lambda$ is defined similarly. We use the notation $f \approx_\lambda g$ if $f \preccurlyeq_\lambda g$ and $f \succcurlyeq_\lambda g$.

Corollary 2.8 (Weight assignment) *Consider the kNN graph based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ from the density p. Let f be of the form $f(x) = \tilde{f}(p(x))$ with $\tilde{f}$ increasing. We assume that $\tilde{f}$ is Lipschitz continuous and f is bounded away from 0. Define $r = (k/(n\eta_d))^{1/d}$ and set the edge weights*

$$w_{ij} = \|X_i - X_j\| \tilde{f}\Big(\frac{r^d}{\|X_i - X_j\|^d}\Big). \tag{2.1}$$

Fix two points $x = X_i$ and $y = X_j$. Assume that $k \leq \sqrt{n}$. Then as $n \to \infty$, $k/n \to 0$ and $k/\log(n) \to \infty$, we have $D_{sp}(x, y) \approx_\lambda D_f(x, y)$ almost surely.

2.4.2 Limit distance problem

Consider a weighted graph based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ from the density p. We are given a increasing edge weight function $h : \mathbb{R}^+ \to \mathbb{R}^+$ which assigns weight $h(\|x - y\|)$ to the edge (x, y). We are interested in finding the limit of the graph shortest path distance with respect to edge weight function h as the sample size goes to infinity. In particular we are looking for a distance function f such that the shortest path distance converges to the f-distance.

Assume we knew the solution $f^* = \tilde{f}^*(p(x))$ of this problem. To guarantee the convergence of the distances, f^* should assign weights of the form of

$$w_{ij} \approx \tilde{f}^*(p(X_i))\|X_i - X_j\|.$$

This would mean

$$\tilde{f}^*(p(X_i)) \approx \frac{h(\|X_i - X_j\|)}{\|X_i - X_j\|},$$

which shows that determining $\tilde{f}^*$ is closely related to finding a density based estimation for $\|X_i - X_j\|$.

Depending on h, we distinguish two regimes for this problem. In the first regime, the function h is such that shortest path distance would prefer using long edges rather than short edges. It is the other way around in the second regime.

Fix a subset of real numbers $A \subset \mathbb{R}$. A function $h : \mathbb{R}^+ \to \mathbb{R}^+$ is called A-subhomogeneous (Burai and Száz, 2005) if

$$\forall a \in A \ : \ h(ax) \leq ah(x).$$

A A-superhomogeneous function is defined analogously.

Lemma 2.9 *The following statements are equivalent for a function $h : \mathbb{R} \to \mathbb{R}$.*

- *h is $(0, 1]$-superhomogeneous*
- *h is $[1, \infty)$-subhomogeneous*
- *$-h$ is $(0, 1]$-subhomogeneous*

The shortest path in kNN-graphs with $[1, \infty)$-subhomogeneous weights would prefer to jump through longer edges (see Lemma 2.10). It is the other way around with $(0, 1]$-subhomogeneous weights. Therefor, we use the names *long-jump* and *short-jump* for $[1, \infty)$-subhomogeneous and $(0, 1]$-subhomogeneous functions respectively.

Long-jump weights

From Lemma 2.9, a function $h(x)$ is long-jump if

$$\forall a \in (0,1] : h(ax) \geq ah(x).$$

Common examples of long-jump functions are $f(x) = x^t$, $t < 1$ and $f(x) = xe^{-x}$. The next lemma shows that for a long-jump function h, the triangle inequality holds.

Lemma 2.10 (Long-jump and triangle inequality) *Every long-jump function h satisfies the triangle inequality.*

Proof. From the definition of long-jump function,

$$\begin{aligned} h(x) + h(y) &= h\big(\frac{x}{x+y}(x+y)\big) + h\big(\frac{y}{x+y}(x+y)\big) \\ &\geq \frac{x}{x+y}h(x+y) + \frac{y}{x+y}h(x+y) = h(x+y). \end{aligned}$$

□

In fact, long-jump weight functions satisfy even an stronger property that we call it the *long-jump property*: the shortest path distance with long-jump weights will prefer jumping along distant vertices. This property is formalized in the next lemma.

Lemma 2.11 (Long-jump property for long-jump functions)
Consider a long-jump function h and two sets of positive numbers $\{x_1, \cdots, x_s\}$ *and* $\{y_1, \cdots, y_t\}$ *such that* $\forall i, j : x_i \leq y_j$ *and*

$$\sum_{i=1}^{s} x_i = \sum_{j=1}^{t} y_j.$$

Then

$$\sum_{i=1}^{s} h(x_i) \geq \sum_{j=1}^{t} h(y_j).$$

Proof. Set $S = \sum_{i=1}^{s} x_i$ and without loss of generality, assume that $y_1 \leq y_2 \leq \cdots \leq y_t$. Observe that

$$\begin{aligned} \sum_{i=1}^{s} h(x_i) &= \sum_{i=1}^{s} h\big(\frac{x_i}{y_1}y_1\big) \geq \sum_{i=1}^{s} \frac{x_i}{y_1}h(y_1) \\ &= \frac{S}{y_1}h(y_1) = \sum_{j=1}^{t} \frac{y_j}{y_1}h\big(\frac{y_1}{y_j}y_j\big) \geq \sum_{j=1}^{t} h(y_j), \end{aligned}$$

which finishes the proof. □

The long-jump functions are the only continuous functions that satisfy the long-jump property. This is proved in the next lemma.

Lemma 2.12 *If a continuous function h satisfies the property:*

$$\sum_{i=1}^{s} h(x_i) \geq \sum_{j=1}^{t} h(y_j),$$

for all sets of positive numbers $\{x_1, \cdots, x_s\}$ and $\{y_1, \cdots, y_t\}$ such that $\forall i, j : x_i \leq y_j$ and $\sum_{i=1}^s x_i = \sum_{j=1}^t y_j$, then h is a long-jump function.

Proof. Choose $x_i = x$ and $y_j = \frac{s}{t}x$ for arbitrary integers $s \geq t$. Then we have

$$\frac{s}{t}h(x) \geq h\big(\frac{s}{t}x\big),$$

which shows that h is $[1, \infty) \cap \mathbb{Q}$-subhomogeneous. Now using the continuity of h shows that it is $[1, \infty)$-subhomogeneous or long-jump . □

Based on this intuition, we come up with the following guess for vertices along the shortest path: For ϵ-graphs we have the approximation $\|X_i - X_j\| \approx \epsilon$ and $f(x) = h(\epsilon)/\epsilon$. For kNN-graphs we have $\|X_i - X_j\| \approx r/p(X_i)^{1/d}$ with $r = (k/(n\eta_d))^{1/d}$ and

$$f(x) = h\big(\frac{r}{p(x)^{1/d}}\big)\frac{p(x)^{1/d}}{r}\ ,\ \tilde{f}(x) = h\big(\frac{r}{x^{1/d}}\big)\frac{x^{1/d}}{r}.$$

We formally prove this statement for kNN graphs in the next theorem. In contrast to Theorem 2.1, the scaling factor is moved into f. The proof for ϵ-graphs is much simpler and can be adapted by setting $r = \epsilon$, $p(x)^{1/d} = 1$, and $r_{low} = r_{up} = \epsilon$.

Theorem 2.13 (Limit of shortest path in weighted graphs) *Consider the kNN graph based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ from the density p. Let h be an increasing, Lipschitz continuous and long-jump function, and define the edge weights $w_{ij} = h(\|X_i - X_j\|)$. Fix two points $x = X_i$ and $y = X_j$. Define $r = (k/(n\eta_d))^{1/d}$ and set*

$$f(x) = h\big(\frac{r}{p(x)^{1/d}}\big)\frac{p(x)^{1/d}}{r}.$$

Assume that $k \leq \sqrt{n}$. Then as $n \to \infty$, $k/n \to 0$ and $k/\log(n) \to \infty$ we have $D_{sp}(x, y) \approx_\lambda D_f(x, y)$ almost surely.

Proof. The essence of the proof is similar to the one in Theorem 2.1, we present a sketch only. The main step is to adapt Proposition 2.4 to weighted graphs with weight function h. Adapting Lemma 2.5 for general f is straightforward. The lemma states that $D_f(x, y) \approx_\lambda f(x)\|x - y\|$ for nearby points. We set r_{low} and ς as in the sampling lemma and Proposition 2.6 (these are properties of kNN graphs and hold for any f). Proposition 2.6 says that in kNN graphs, x is connected to y with high probability iff $\|x - y\| \preccurlyeq_\lambda r/p(x)^{1/d}$. The probabilistic argument and the criteria on choosing λ are similar to Theorem 2.1.

First we show that $D_{sp}(x, y) \preccurlyeq_\lambda D_f(x, y)$. Consider the f-geodesic path $\gamma^*_{x,y}$ connecting x to y. Divide $\gamma^*_{x,y}$ into segments

$$u_0 = x, u_1, ..., u_t, u_{t+1} = y$$

such that $D_{\text{PD}}(u_i, u_{i+1}) = r_{low} - 2\varsigma$ for $i = 0, ..., t-1$ and $D_{\text{PD}}(u_t, u_{t+1}) \leq r_{low} - 2\varsigma$ (see Figure 2.2). There exists a vertex v_i near to u_i, and v_{i+1} near to $u_{i+1}i$ such that v_i and v_{i+1} are connected. We show that the length of the path $x, v_1, ..., v_t, y$ is approximately smaller than $D_f(x, y)$. From the path construction we have

$$\|v_i - v_{i+1}\| \approx_\lambda \|u_i - u_{i+1}\| \approx_\lambda \frac{r}{p(u_i)^{1/d}}.$$

By summing up along the path we get

$$\begin{aligned} D_{sp}(x,y) &\leq \sum_i h(\|v_i - v_{i+1}\|) \\ &\approx_\lambda \sum_i h(\|u_i - u_{i+1}\|) \approx_\lambda \sum_i h(\frac{r}{p(u_i)^{1/d}}) \\ &= \sum_i f(u_i) \frac{r}{p(u_i)^{1/d}} \approx_\lambda \sum_i f(u_i)\|u_i - u_{i+1}\|. \end{aligned}$$

From the adaptation of Lemma 2.5 we have $D_f(u_i, u_{i+1}) \approx_\lambda f(u_i)\|u_i - u_{i+1}\|$, which gives

$$\sum_i f(u_i)\|u_i - u_{i+1}\| \approx_\lambda \sum_i D_f(u_i, u_{i+1}) = D_f(x,y).$$

This shows that $D_{sp}(x,y) \preccurlyeq_\lambda D_f(x,y)$.

For the other way round, we use a technique different from Proposition 2.4. Denote the graph shortest path between x and y by $\pi : z_0 = x, z_1, ..., z_s, z_{s+1} = y$. Consider π' as a continuous path in $\mathcal{X}$ corresponding to π. As in the previous part, divide π' into segments $u_0 = x, u_1, ..., u_t, u_{t+1} = y$ (see Figure 2.2). From $D_{\text{PD}}(z_i, z_{i+1}) \preccurlyeq_\lambda r$ and $D_{\text{PD}}(u_i, u_{i+1}) \approx_\lambda r$ we have $s \succcurlyeq_\lambda t$. Using this and Lemma 2.11 we get

$$D_{sp}(x,y) = \sum_i h(\|z_i - z_{i+1}\|) \succcurlyeq_\lambda \sum_i h(\|u_i - u_{i+1}\|).$$

To prove $D_{sp}(x,y) \succcurlyeq_\lambda D_f(x,y)$, we can write

$$\begin{aligned} \sum_i h(\|u_i - u_{i+1}\|) &\approx_\lambda \sum_i h(\frac{r}{q(u_i)}) = \sum_i \frac{f(u_i)r}{q(u_i)} \\ &\approx_\lambda \sum_i f(u_i)\|u_i - u_{i+1}\| \\ &\approx_\lambda \sum_i D_f(u_i, u_{i+1}) \succcurlyeq_\lambda D_f(x,y). \end{aligned}$$

□

The **proof of Corollary 2.8** is a direct consequence of this theorem. It follows by choosing $h(t) = t\tilde{f}(r^d/t^d)$ (which is long-jump if $\tilde{f}$ is increasing) and setting $w_{ij} = h(\|X_i - X_j\|)$.

Short-jump weights

From Lemma 2.9, a function $h(x)$ is short-jump if

$$\forall a \in (0,1] : h(ax) \leq ah(x).$$

Examples are $f(x) = x^a; a > 1$ and $f(x) = xe^x$. To get an intuition on the behavior of the shortest path for a short-jump h, take an example of three vertices x, y, z which are all connected in the graph and sit on a straight line such that $\|x - y\| + \|y - z\| = \|x - z\|$. By the short-jump property, the shortest path between x and z will prefer going through y rather than directly jumping to z. More generally, the graph shortest path will prefer taking many "small" edges rather than fewer "long" edges. For this reason, we do not expect a big difference between short-jump weighted kNN graphs and ϵ-graphs: the long edges in the kNN graph will not be used anyway. However, due to technical problems we did not manage to prove a formal theorem to this effect.

The special case of the short-jump family $h(x) = x^a$, $a > 1$ is treated in Hwang et al. (2012) by completely different methods. Although their results are presented for complete graphs, we believe that it can be extended to ϵ and kNN graphs. We are not aware of any other result for the limit of shortest path distance for short-jump weight functions.

2.5 Consequences in applications

In this section we study the consequences of our results in two applications: manifold embedding using Isomap and on a particular semi-supervised learning method.

There are two cases where we do not expect a drastic difference between the shortest path in weighted and unweighted kNN graph: (1) If the underlying density p is close to uniform. (2) If the intrinsic dimensionality of our data d is high. The latter is because in the PD-distance, the underlying density arises in the form of $p(x)^{1/d}$, where the exponent flattens the distribution for large d.

2.5.1 Isomap

Isomap is a widely used method for low dimensional manifold embedding (Tenenbaum et al., 2000). The main idea is to use metric multidimensional scaling on the matrix of pairwise geodesic distances. Using the Euclidean length of edges as their weights will lead to the convergence of the shortest path distance to the geodesic distance. But what would be the effect of applying Isomap to unweighted graphs?

Our results of the last section already hint that there is no big difference between unweighted and weighted ϵ-graphs for Isomap. However, the case of kNN graphs is different because weighted and unweighted shortest paths measure different quantities. The effect of applying Isomap to unweighted kNN graphs can easily be demonstrated by the following simulation. We sample 2000 points in $\mathbb{R}^2$ from a distribution that has two uniform high-density squares, surrounded by a uniform low density region. An unweighted kNN graph is constructed with $k = 10$, and we apply Isomap with target dimension 2. The result is depicted in Figure 2.3. We can see that the Isomap embedding heavily distorts the original data: it stretches high density regions and compacts low density regions to make the vertex distribution close to uniform.

2.5.2 Semi-supervised learning

Our work has close relationship to some of the literature on semi-supervised learning (SSL). In regularization based approaches, the underlying density is either exploited implicitly as attempted in Laplacian regularization (Zhu et al., 2003 but see Nadler et al., 2009, Alamgir and von Luxburg, 2011 and Zhou and Belkin, 2011), or more explicitly as in measure based regularization (Bousquet et al., 2004). Alternatively, one defines new distance functions on

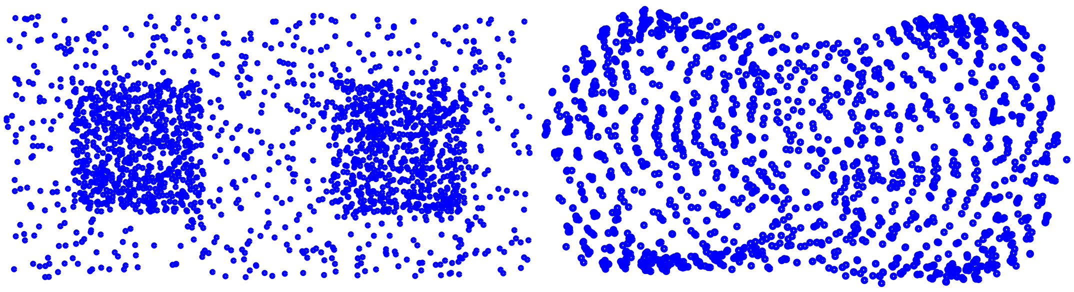


Figure 2.3: Original data (left) and its Isomap reconstruction based on an unweighted kNN graph (right).

the data that take the density of the unlabeled points into account. Here, the papers by Sajama and Orlitsky (2005) and Bijral et al. (2011) are most related to our work. Both papers suggest different ways to approximate the density based distance from the data. In Sajama and Orlitsky (2005) it is achieved by estimating the underlying density while in Bijral et al. (2011), the authors omit the density estimation and use an approximation.

Our work shows a simpler way to converge to a similar distance function for a specific family of f-distances, namely constructing a kNN graph and assigning edge weights as in Equation 2.1.

2.6 Conclusions and outlook

We have seen in this chapter that the shortest path distance on unweighted kNN graphs has a very funny limit behavior: it prefers to go through regions of low density and even takes large detours in order to avoid the high density regions.

In some sense, unweighted ε-graphs and unweighted kNN graphs behave as "duals" of each other: while degrees in ε-graphs reflect the underlying density, they are independent of the density in kNN graphs. While the shortest path in ε-graphs is independent of the underlying density and converges to the Euclidean distance, the shortest paths in kNN graphs take the density into account. Current practice is to use ε and kNN graphs more or less interchangeably in many applications, and the decision for one or the other graph is largely driven by robustness or convenience considerations. However, as our results show it is important to be aware of the implicit consequences of this choice. Each graph carries different information about the underlying density, and depending on how a particular machine learning algorithm makes use of the graph structure, it might either miss out or benefit from this information.

Last but not least, the behavior of the shortest path distance in unweighted kNN graphs is not always undesired. The fact that it depends on the density may exploited in different applications. An example is density-preserving quantization which is discussed in Chapter 7.

Chapter 3

Phase transition in the family of p-resistances

3.1 Introduction

The graph Laplacian is a popular tool for unsupervised and semi-supervised learning problems in graphs. It is used in the context of spectral clustering, as a regularizer for semi-supervised learning, or to compute the resistance distance in graphs. However, it has been observed that under certain circumstances, standard Laplacian-based methods show undesired artifacts. In the semi-supervised learning setting Nadler et al. (2009) showed that as the number of unlabeled points increases, the solution obtained by Laplacian regularization degenerates to a non-informative function. von Luxburg et al. (2010) proved that as the number of points increases, the resistance distance converges to a meaningless limit function. Independently of these observations, a number of authors suggested to generalize Laplacian methods. The observation was that the "standard" Laplacian methods correspond to a vector space setting with L_2-norms, and that it might be beneficial to work in a more general L_p setting for $p \neq 2$ instead. See Bühler and Hein (2009) for an application to clustering and Herbster and Lever (2009) for an application to label propagation. In this chapter we take up several of these loose ends and connect them.

The main object under study in this chapter is the family of p-resistances, which is a generalization of the standard resistance distance. Our first major result proves that the family of p-resistances is very rich and contains several special cases. The general picture is that the smaller p is, the more the resistance is concentrated on "short paths". In particular, the case $p = 1$ corresponds to the shortest path distance in the graph, the case $p = 2$ to the standard resistance distance, and the case $p \to \infty$ to the inverse s-t-mincut.

Second, we study the behavior of p-resistances in the setting of random geometric graphs like lattice graphs, ε-graphs or k-nearest neighbor graphs in a d-dimensional space. We prove that as the sample size n increases, there are two completely different regimes of behavior. Namely, there exist two critical thresholds p^* and p^{**} such that if $p < p^*$, the p-resistances convey useful information about the global topology of the data (such as its cluster properties), whereas for $p > p^{**}$ the resistance distances approximate a limit that does not convey any useful information. We can explicitly compute the value of the critical thresholds $p^* := 1+1/(d-1)$ and $p^{**} := 1 + 1/(d-2)$. This result even holds independently of the exact construction of the geometric graph.

Third, as we will see in Section 3.5, our results also shed light on the Laplacian regularization and semi-supervised learning setting. As there is a tight relationship between p-resistances and graph Laplacians, we can reformulate the artifacts described in Nadler et al. (2009) in terms of p-resistances. Taken together, our results suggest that standard Laplacian regularization should be replaced by q-Laplacian regularization (where q is such that $1/p^* + 1/q = 1$).

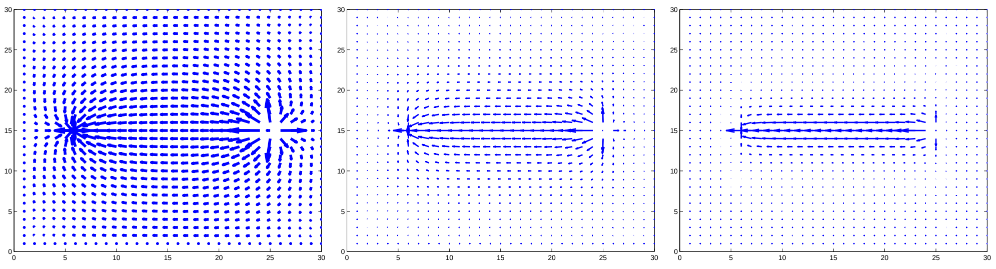
3.2 Intuition and main results

Consider an undirected, weighted graph $G = (V, E)$ with n vertices. As is standard in machine learning, the edge weights are supposed to indicate similarity of the adjacent points (not distances). Denote the weight of edge e by $w_e \geq 0$ and the degree of vertex u by d_u. The length of a path γ in the weighted graph is defined as $\sum_{e \in \gamma} 1/w_e$. In the electrical network interpretation, a graph is considered as a network where each edge $e \in E$ has resistance $r_e = 1/w_e$. The *effective resistance* (or *resistance distance*) $R(s,t)$ between two vertices s and t in the network is defined as the overall resistance one obtains when connecting a unit volt battery to s and t (see also Section 3.9). It can be computed in many ways, but the one most useful for our purpose is the following representation in terms of flows (cf. Section IX.1 of Bollobas, 1998):

$$R(s,t) = \min \Big\{ \sum_{e \in E} r_e i_e^2 \;\Big|\; i = (i_e)_{e \in E} \text{ unit flow from } s \text{ to } t \Big\}. \tag{3.1}$$

In von Luxburg et al. (2010) it has been proved that in many random graph models, the resistance distance $R(s,t)$ between two vertices s and t converges to the trivial limit expression $1/d_s + 1/d_t$ as the size of the graph increases. We now want to present some intuition as to how this problem can be resolved in a natural way. For a subset $M \subset E$ of edges we define the contribution of M to the resistance $R(s,t)$ as the part of the sum in (3.1) that runs over the edges in M. Let i^* be a flow minimizing (3.1). To explain our intuition we separate this flow into two parts: $R(s,t) = R(s,t)^{local} + R(s,t)^{global}$. The part $R(s,t)^{local}$ stands for the contribution of i^* that stems from the edges in small neighborhoods around s and t, whereas $R(s,t)^{global}$ is the contribution of the remaining edges (exact definition given below). A useful distance function is supposed to encode the global geometry of the graph, for example its cluster properties. Hence, $R(s,t)^{global}$ should be the most important part in this decomposition. However, in case of the standard resistance distance the contribution of the global part becomes negligible as $n \to \infty$ (for many different models of graph construction). This effect happens because as the graph increases, there are so many different paths between s and t that once the flow has left the neighborhood of s, electricity can flow "without considerable resistance". The "bottleneck" for the flow is the part that comes from the edges in the local neighborhoods of s and t, because here the flow has to concentrate on relatively few edges. So the dominating part is $R(s,t)^{local}$.

In order to define a useful distance function, we have to ensure that the global part has a significant contribution to the overall resistance. To this end, we have to avoid that the flow is distributed over "too many paths". In machine learning terms, we would like to achieve a flow that is "sparser" in the number of paths it uses. From this point of view, a natural attempt is to replace the 2-norm-optimization problem (3.1) by a p-norm optimization problem for some



(a) $p = 2$ (b) $p = 1.33$ (c) $p = 1.1$

Figure 3.1: The s-t-flows minimizing $(*)$ in a two-dimensional grid for different values of p. The smaller p, the more the flow concentrates along the shortest path.

$p < 2$. Based on this intuition, our idea is to replace the squares in the flow problem (3.1) by a general exponent $p \geq 1$ and define the following distance function on the graph.

Definition 3.1 (p-resistance) *On any weighted graph G, for any $p \geq 1$ we define*

$$R_p(s,t) := \min \left\{ \sum_{e \in E} r_e |i_e|^p \;\middle|\; i = (i_e)_{e \in E} \textit{ unit flow from } s \textit{ to } t \right\}. \qquad (*)$$

As it turns out, our defined distance function R_p is closely related but not completely identical to the p-resistance R_p^H defined by Herbster and Lever (2009). A discussion of this issue can be found in Section 3.6.1.

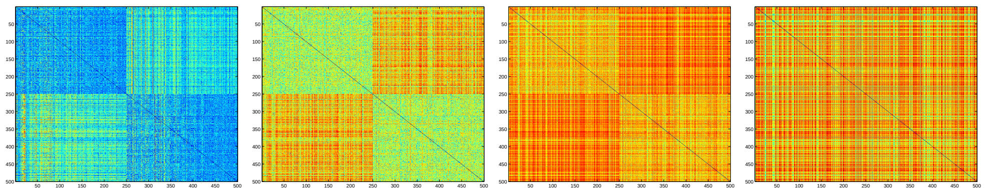
In toy simulations we can observe that the desired effect of concentrating the flow on fewer paths takes place indeed. In Figure 3.1 we show how the optimal flow between two points s and t gets propagated through the network. We can see that the smaller p is, the more the flow is concentrated along the shortest path between s and t.

We are now going to formally investigate the influence of the parameter p. Our first question is how the family $R_p(s,t)$ behaves as a function of p (that is, on a fixed graph and for fixed s,t). The answer is given in the following theorem.

Theorem 3.2 (Family of p-resistances) *For any weighted graph G the following statements are true:*

1. *For $p = 1$, the p-resistance coincides with the shortest path distance on the graph.*
2. *For $p = 2$, the p-resistance reduces to the standard resistance distance.*
3. *For $p \to \infty$, $R_p(s,t)^{q-1}$ converges to $1/m$ where m is the unweighted s-t-mincut.*

This theorem shows that our intuition as outlined above was exactly the right one. The smaller p is, the more flow is concentrated along straight paths. The extreme case is $p = 1$, which yields the shortest path distance. In the other direction, the larger p is, the more widely distributed the flow is. Moreover, the theorem above suggests that for p close to 1, R_p encodes global information about the part of the graph that is concentrated around the shortest path. As p increases, global information is still present, but now describes a larger portion of the graph, say, its cluster structure. This is the regime that is most interesting for machine learning. The larger p becomes, the less global information is present in R_p (because flows even use extremely long paths that take long detours), and in the extreme case $p \to \infty$ we are left with nothing but the information about the minimal s-t-cut. In many large graphs,



(a) $p = 1$ (b) $p = 1.11$ (c) $p = 1.5$ (d) $p = 2$

Figure 3.2: Heat plots of the R_p distance matrices for a mixture of two Gaussians in $\mathbb{R}^{10}$. We can see that the larger p is, the less pronounced the "global information" about the cluster structure is.

the latter just contains local information about one of the points s or t (see the discussion at the end of this section). An illustration of the different behaviors can be found in Figure 3.2.

The next question, inspired by the results of von Luxburg et al. (2010), is what happens to $R_p(s,t)$ if we fix p but consider a family $(G_n)_{n\in\mathbb{N}}$ of graphs such that the number n of vertices in G_n tends to ∞. Let us consider geometric graphs such as k-nearest neighbor graphs or ε-graphs. We now give exact definitions of the local and global contributions to the p-resistance. Let r and R be real numbers that depend on n (they will be specified in Section 3.4) and $C \geq R/r$ a constant. We define the local neighborhood $\mathcal{N}(s)$ of vertex s as the ball with radius $C \cdot r$ around s. We will see later that the condition $C \geq R/r$ ensures that $\mathcal{N}(s)$ contains at least all vertices adjacent to s. By abuse of notation we also write $e \in \mathcal{N}(s)$ if both endpoints of edge e are contained in $\mathcal{N}(s)$. Let i^* be the optimal flow in Problem $(*)$. We define

$$R_p^{local}(s) := \textstyle\sum_{e\in\mathcal{N}(s)} r_e |i_e^*|^p,$$

$R_p^{local}(s,t) := R_p^{local}(s) + R_p^{local}(t)$, and $R_p^{global}(s,t) := R_p(s,t) - R_p^{local}(s,t)$. Our next result conveys that the behavior of the family of p-resistances shows an interesting phase transition. The statements involve a term τ_n that should be interpreted as the average degree in the graph G_n (exact definition see later).

Theorem 3.3 (Phase transition for p-resistances in geometric graphs) *Consider a family $(G_n)_{n\in\mathbb{N}}$ of unweighted geometric graphs on $\mathbb{R}^d$, $d > 2$ that satisfies some general assumptions (see Section 3.4 for definitions and details). Fix two vertices s and t. Define the two critical values $p^* := 1 + 1/(d-1)$ and $p^{**} := 1 + 1/(d-2)$. Then, as $n \to \infty$, the following statements hold:*

1. If $p < p^*$ and τ_n is sub-polynomial in n, then $R_p^{global}(s,t)/R_p^{local}(s,t) \to \infty$, that is the global contribution dominates the local one.

2. If $p > p^{**}$ and $\tau_n \to \infty$, then $R_p^{local}(s,t)/R_p^{global}(s,t) \to \infty$ and

$$R_p(s,t) \to \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}},$$

that is all global information vanishes.

This result is interesting. It shows that there exists a non-trivial point of phase transition in the behavior of p-resistances: if $p < p^*$, then p-resistances are informative about the global topology of the graph, whereas if $p > p^{**}$ the p-resistances converge to trivial distance functions that do not depend on any global properties of the graph. In fact, we believe that p^{**} should be $1 - 1/(d-1)$ as well, but our current proof leaves the tiny gap between $p^* = 1 - 1/(d-1)$ and $p^{**} = 1 - 1/(d-2)$.

Theorem 3.3 is a substantial extension of the work of von Luxburg et al. (2010), in several respects. First, and most importantly, it shows the complete picture of the full range of $p \geq 1$, and not just the single snapshot at $p = 2$. We can see that there is a range of values for p for which p-resistance distances convey very important information about the global topology of the graph, even in extremely large graphs. Also note how nicely Theorems 3.2 and 3.3 fit together. It is well-known that as $n \to \infty$, the shortest path distance corresponding to $p = 1$ converges to the (geodesic) distance of s and t in the underlying space (Tenenbaum et al., 2000), which of course conveys global information. von Luxburg et al. (2010) proved that the standard resistance distance ($p = 2$) converges to the trivial local limit. Theorem 3.3 now identifies the point of phase transition p^* between the boundary cases $p = 1$ and $p = 2$. Finally, for $p \to \infty$, we know by Theorem 3.2 that the p-resistance converges to the inverse of the s-t-min-cut. It is widely believed that the minimal s-t cut in geometric graphs converges to the minimum of the degrees of s and t as $n \to \infty$ (even though a formal proof has yet to be presented and we cannot point to any reference). This is in alignment with the result of Theorem 3.3 that the p-resistance converges to $1/d_s^{p-1} + 1/d_t^{p-1}$. As $p \to \infty$, only the smaller of the two degrees contributes to the local part, which agrees with the limit for the s-t-mincut.

3.3 Equivalent optimization problems

In this section we will consider different optimization problems that are inherently related to p-resistances. In Section 3.9, we discuss the electrical network interpretation of p-resistances. There, the relation between these optimization problems becomes evident, as they both minimize the energy consumption in the network. We postpone this discussions to that section and here use only the definition of the p-resistance to prove this equivalence. All graphs in this section are considered to be weighted.

Consider the following two optimization problems for $p > 1$:

Flow-problem: $(*)$

$$R_p(s,t) := \min \Big\{ \textstyle\sum_{e \in E} r_e |i_e|^p \;\Big|\; i = (i_e)_{e \in E} \text{ unit flow from } s \text{ to } t \Big\} \tag{3.2}$$

Potential problem: $(**)$

$$C_p(s,t) = \min \Big\{ \sum_{e=(u,v)} \frac{|\varphi(u) - \varphi(v)|^{1+\frac{1}{p-1}}}{r_e^{\frac{1}{p-1}}} \;\Big|\; \varphi(s) - \varphi(t) = 1 \Big\} \tag{3.3}$$

It is well known that these two problems are equivalent for $p = 2$ (see Section 1.3 of Doyle and Snell, 1984). We will now extend this result to general $p > 1$.

Proposition 3.4 (Equivalent optimization problems) *For $p > 1$, the following statements are true:*

1. *The flow problem $(*)$ has a unique solution.*

2. *The solutions of* $(*)$ *and* $(**)$ *satisfy* $R_p(s,t) = C_p(s,t)^{-(p-1)}$.

To prove this proposition, we derive the Lagrange dual of problem $(*)$ and use the homogeneity of the variables to convert it to the form of problem $(**)$. Details can be found in Section 3.8.1.

3.4 Local and global contributions of flows

In this section we consider the class of random geometric graphs. The vertices of such graphs consist of points $X_1, .., X_n \in \mathbb{R}^d$, and vertices are connected by edges if the corresponding points are "close" (for example, they are k-nearest neighbors of each other). In most cases, we consider the set of points as drawn i.i.d from some density on $\mathbb{R}^d$. Consider the following general assumptions.

General Assumptions: *A family* $(G_n)_{n\in\mathbb{N}}$ *of unweighted geometric graphs is given, where* G_n *is based on* $X_1, ..., X_n \in M \subset \mathbb{R}^d$, $d > 2$. *We assume that there exist* $0 < r \leq R$ *(depending on n and d) such that the following statements about* G_n *holds simultaneously for all* $x \in \{X_1, ..., X_n\}$*:*

1. *Distribution of points: For* $\rho \in \{r, R\}$ *the number of sample points in* $B(x, \rho)$ *is of the order* $\Theta(n \cdot \rho^d)$.
2. *Graph connectivity:* x *is connected to all sample points inside* $B(x, r)$ *and* x *is not connected to any sample point outside* $B(x, R)$.
3. *Geometry of* M*:* M *is a compact, connected set such that* $M \setminus \partial M$ *is still connected. The boundary* ∂M *is regular in the sense that there exist positive constants* $\alpha > 0$ *and* $\varepsilon_0 > 0$ *such that if* $\varepsilon < \varepsilon_0$, *then for all points* $x \in \partial M$ *we have* $\mathrm{vol}(B_\varepsilon(x) \cap M) \geq \alpha\, \mathrm{vol}(B_\varepsilon(x))$ *(where* vol *denotes the Lebesgue volume). Essentially this condition just excludes the situation where the boundary has arbitrarily thin spikes.*

It is a straightforward consequence of these assumptions that there exists some function $\tau(n) =: \tau_n$ such that r and R are both of the order $\Theta((\tau_n/n)^{1/d})$ and all degrees in the graph are of order $\Theta(\tau_n)$.

To prove Theorem 3.3 we need to study the balance between R_p^{local} and R_p^{global}. We introduce the shorthand notation

$$T_1 = \Theta\Big(\frac{1}{n^{p(1-1/d)-1}\tau_n^{p(1+1/d)-1}}\Big) \quad , \quad T_2 = \Theta\Big(\frac{1}{\tau_n^{2(p-1)}} \sum_{k=1}^{1/r} \frac{1}{k^{(d-2)(p-1)}}\Big).$$

Theorem 3.5 (General bounds on R_p^{local} and R_p^{global}) *Consider a family* $(G_n)_{n\in\mathbb{N}}$ *of unweighted geometric graphs that satisfies the general assumptions. The constant* C *is chosen such that* $C \geq R/r$. *Then the following statements are true for any fixed pair* s, t *of vertices in* G_n*:*

$$4C \;>\; R_p^{local}(s,t) \;\geq\; \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} \qquad \textit{and} \qquad T_1 + T_2 \;\geq\; R_p^{global}(s,t) \;\geq\; T_1.$$

Note that by taking the sum of the two inequalities this theorem also leads to upper and lower bounds for $R_p(s,t)$ itself. The proof of Theorem 3.5 consists of several parts. To derive lower bounds on $R_p(s,t)$ we construct a second graph G'_n which is a contracted version of G_n. Lower bounds can then be obtained by Rayleigh's monotonicity principle. To get upper bounds on $R_p(s,t)$ we exploit the fact that the p-resistance in an unweighted graph can be upper bounded by $\sum_{e \in E} i_e^p$, where i is any unit flow from s to t. We construct a particular flow that leads to a good upper bound. Finally, investigating the properties of lower and upper bounds we can derive the individual bounds on R_p^{local} and R_p^{global}. Details can be found in Section 3.8.4.

3.4.1 Applications

Our general results can directly be applied to many standard geometric graph models.

The ε-graph. We assume that $X_1, ..., X_n$ have been drawn i.i.d from some underlying density f on $\mathbb{R}^d$, where $M := \text{supp}(f)$ satisfies Part (3) of the general assumptions. Points are connected by unweighted edges in the graph if their Euclidean distances are smaller than ε. Exploiting standard results on ε-graphs (cf. the appendix in von Luxburg et al., 2010), it is easy to see that the general assumptions (1) and (2) are satisfied with probability at least $1 - c_1 n \exp(-c_2 n \varepsilon^d)$ (where c_1, c_2 are constants independent of n and d) with $r = R = \varepsilon$ and $\tau_n = \Theta(n\varepsilon^d)$. The probability converges to 1 if $n \to \infty$, $\varepsilon \to 0$ and $n\varepsilon^d / \log(n) \to \infty$.

k-nearest neighbor graphs. We assume that $X_1, ..., X_n$ have been drawn i.i.d from some underlying density f on $\mathbb{R}^d$, where $M := \text{supp}(f)$ satisfies Part (3) of the general assumptions. We connect each point to its k nearest neighbors by an undirected, unweighted edge. Exploiting standard results on kNN-graphs (cf. Proposition 2.6 and Lemma 2.7), it is easy to see that the general assumptions (1) and (2) are satisfied with probability at least $1 - c_1 k \exp(-c_2 k)$ with $r = \Theta\big((k/n)^{1/d}\big)$, $R = \Theta\big((k/n)^{1/d}\big)$, and $\tau_n = k$. The probability converges to 1 if $n \to \infty$, $k \to \infty$, and $k/\log(n) \to \infty$.

Lattice graphs. Consider uniform lattices such as the square lattice or triangular lattice in $\mathbb{R}^d$. These lattices have constant degrees, which means that $\tau_n = \Theta(1)$. If we denote the edge length of grid by ε, the total number of nodes in the support will be in the order of $n = \Theta(1/\varepsilon^d)$. This means that the general assumptions hold for $r = R = \varepsilon = \Theta(\frac{1}{n^{1/d}})$ and $\tau_n = \Theta(1)$. Note that while the lower bounds of Theorem 3.3 can be applied to the lattice case, our current upper bounds do not hold because they require that $\tau_n \to \infty$.

3.5 Regularization by p-Laplacians

One of the most popular methods for semi-supervised learning in graphs is based on Laplacian regularization. In Zhu et al. (2003) the label assignment problem is formulated as

$$\varphi^* = \text{argmin}_\varphi \, C(\varphi) \qquad \text{subject to} \qquad \varphi(x_i) = y_i \ , \ i = 1, \ldots, l \qquad (3.4)$$

where $y_i \in \{\pm 1\}$, $C(\varphi) := \varphi^T L \varphi$ is the energy function involving the standard ($p = 2$) graph Laplacian L. This formulation is appealing and works well for small sample problems. However, Nadler et al. (2009) showed that the method is not well posed when the number of unlabeled data points is very large. In this setting, the solution of the optimization problem converges to a constant function with "spikes" at the labeled points. We now present a simple theorem that connects these findings to those concerning the resistance distance.

Theorem 3.6 (Laplacian regularization in terms of resistance distance) *Consider a semi-supervised classification problem with one labeled point per class: $\varphi(s) = 1$, $\varphi(t) = -1$. Denote the solution of (3.4) by φ^*, and let v be an unlabeled data point. Then*

$$\varphi^*(v) - \varphi^*(t) > \varphi^*(s) - \varphi^*(v) \iff R_2(v,t) > R_2(v,s).$$

Proof. First we show that the optimal solution is in the form

$$\varphi^* = c_1 L^\dagger(e_s - e_t) + c_2,$$

where $L^\dagger$ is the pseudo-inverse of the Laplacian matrix L. Decompose L using its eigenvectors and eigenvalues $L = \sum \lambda_i v_i v_i^T$ and express φ in the base of these eigenvectors $\varphi = \sum a_i v_i$.

Now our optimization problem turns into

$$\begin{aligned} \underset{a}{\text{minimize}} \ & \sum a_i^2 \lambda_i \\ \text{subject to} \ & \sum a_i v_i(s) = 1 \ , \ \sum a_i v_i(t) = -1. \end{aligned}$$

Using the Lagrange multiplier technique, the end solution is $\varphi^* = c_1 L^\dagger(e_s - e_t) + c_2$ where

$$c_1 = \frac{2}{R_2(s,t)} \ ; \ c_2 = -\frac{e_t^T L^\dagger(e_s - e_t)}{R_2(s,t)} = \frac{L^\dagger(t,t) - L^\dagger(s,t)}{R_2(s,t)}.$$

It is easy to verify that

$$R_2(s,t) = \frac{1}{2}(e_s - e_t)^T L^\dagger (e_s - e_t).$$

Note that the extra factor 1/2 appear because of setting voltages to +1 and −1. In the standard case, they are set to +1 and 0, so the factor 1/2 does not appear there.

Finally we have

$$\begin{aligned} & \varphi^*(v) - \varphi^*(t) > \varphi^*(s) - \varphi^*(v) \\ \iff & (e_v - e_t)^T L^\dagger (e_s - e_t) > (e_s - e_v)^T L^\dagger (e_s - e_t) \\ \overset{(a)}{\iff} & (e_v - e_t)^T L^\dagger (e_v - e_t) > (e_v - e_s)^T L^\dagger (e_v - e_s) \\ \iff & R_2(v,t) > R_2(v,s). \end{aligned}$$

Here in step (a) we use the symmetry of $L^\dagger$ to state that $e_v^T L^\dagger e_s = e_s^T L^\dagger e_v$. □

What does this theorem mean? We have seen above that in case $p = 2$, if $n \to \infty$,

$$R_2(v,t) \approx \frac{1}{d_v} + \frac{1}{d_t} \qquad \text{and} \qquad R_2(v,s) \approx \frac{1}{d_v} + \frac{1}{d_s}.$$

Hence, the theorem states that if we threshold the function φ^* at 0 to separate the two classes, then all the points will be assigned to the labeled vertex with the larger degree.

Our conjecture is that an analogue to Theorem 3.6 also holds for the right p in random geometric graphs. For a precise formulation, define the matrix ∇ as

$$\nabla_{i,j} = \begin{cases} \varphi(i) - \varphi(j) & i \sim j \\ 0 & \text{otherwise} \end{cases}$$

and introduce the matrix norm $\|A\|_{m,n} = \big(\sum_i((\sum_j a_{ij}^m)^{1/m})^n\big)^{1/n}$. Consider q such that $1/p + 1/q = 1$. We conjecture that if we used $\|\nabla\|_{q,q}$ as a regularizer for semi-supervised learning, then the corresponding solution φ^* would satisfy

$$\varphi^*(v) - \varphi^*(t) > \varphi^*(s) - \varphi^*(v) \iff R_p(v,t) > R_p(v,s).$$

That is, the solution of the q-regularized problem would assign labels according to the R_p-distances. In particular, using q-regularization for the value q with $1/q + 1/p^* = 1$ would resolve the artifacts of Laplacian regularization described in Nadler et al. (2009).

It is worth mentioning that this regularization is different from others in the literature. The usual Laplacian regularization term as in Zhu et al. (2003) coincides with $\|\nabla\|_{2,2}$, Zhou and Schölkopf (2005) use the $\|\nabla\|_{2,p}$ norm, and our conjecture is that the $\|\nabla\|_{q,q}$ norm would be a good candidate. Proving whether this conjecture is right or wrong is a subject of future work.

3.6 Related families of distance functions

In this section we sketch some relations between p-resistances and other families of distances.

3.6.1 Herbster's definition of p-resistances

For $p \leq 2$, Herbster and Lever (2009) introduced the following definition of p-resistances:

$$R_{p'}^H(s,t) := \frac{1}{C_{p'}^H(s,t)}$$

with

$$C_{p'}^H(s,t) := \min\Big\{ \sum_{e=(u,v)} \frac{|\varphi(u)-\varphi(v)|^{p'}}{r_e} \mid \varphi(s) - \varphi(t) = 1 \Big\}.$$

In Section 3.3 we have seen that the potential and flow optimization problems are duals of each other. Based on this derivation we believe that the natural way of relating R^H and C^H would be to replace the p' in Herbster's potential formulation by q' such that $1/p' + 1/q' = 1$. That is, one would have to consider $C_{q'}^H$ and then define $\widehat{R}_{p'}^H := 1/C_{q'}^H$. In particular, reducing Herbster's p' towards 1 has the same influence as increasing our p to infinity and makes $R_{p'}^H$ converge to the minimal s-t-cut.

To ease further comparison, let us assume for now that we use "our" p in the definition of Herbster's resistances. Then one can see by similar arguments as in Section 3.3 that R_p^H can be rewritten as

$$R_p^H(s,t) := \min\Big\{ \sum_{e\in E} r_e^{p-1}\, |i_e|^p \;\Big|\; i = (i_e)_{e\in E} \text{ unit flow from } s \text{ to } t\Big\}. \tag{H}$$

Now it is easy to see that the main difference between Herbster's definition (H) and our definition $(*)$ is that (H) takes the power $p-1$ of the resistances r_e, while we keep the resistances with power 1. In many respects, R_p and R_p^H have properties that are similar to each other: they satisfy slightly different versions (with different powers or weights) of the triangle inequality, Rayleigh's monotonicity principle, laws for resistances in series and in parallel, and so on.

3.6.2 Other families of distances

There also exist other families of distances in graphs that share some of the properties of p-resistances. We will only discuss the ones that are most related to our work, for more references see von Luxburg et al., 2010, Bavaud and Guex, 2012 and Kivimäki et al. (2014). The first such family was introduced by Yen et al. (2008), where the authors use a statistical physics approach to reduce the influence of long paths to the distance. This family is parameterized by a parameter θ, contains the shortest path distance at one end ($\theta \to \infty$) and the standard resistance distance at the other end ($\theta \to 0$). However, the construction is somewhat ad hoc, the resulting distances cannot be computed in closed form and do not even satisfy the triangle inequality. A second family is the one of "logarithmic forest distances" by Chebotarev (2011). Even though its derivation is complicated, it has a closed form solution and can be interpreted intuitively: The contribution of a path to the overall distance is "discounted" by a factor $(1/\alpha)^l$ where l is the length of the path. For $\alpha \to 0$, the logarithmic forest distance distance converges to the shortest path distance, for $\alpha \to \infty$, it converges to the resistance distance.

At the time of writing this thesis, the major disadvantage of both the families introduced by Yen et al. (2008) and Chebotarev (2011) is that it is unknown how their distances behave as the size of the graph increases. It is clear that on the one end (shortest path), they convey global information, whereas on the other end (resistance distance) they depend on local quantities only when $n \to \infty$. But what happens to all intermediate parameter values? Do all of them lead to meaningless distances as $n \to \infty$, or is there some interesting phase transition as well? As long as this question has not been answered, one should be careful when using these distances. In particular, it is unclear how the parameters (θ and α, respectively) should be chosen, and it is hard to get an intuition about this.

3.7 Conclusions

We proved that the family of p-resistances has a wide range of behaviors. In particular, for $p = 1$ it coincides with the shortest path distance, for $p = 2$ with the standard resistance distance and for $p \to \infty$ it is related to the minimal s-t-cut. Moreover, an interesting phase transition takes place: in large geometric graphs such as k-nearest neighbor graphs, the p-resistance is governed by meaningful global properties as long as $p < p^* := 1 + 1/(d-1)$, whereas it converges to the trivial local quantity $1/d_s^{p-1} + 1/d_t^{p-1}$ if $p > p^{**} := 1 + 1/(d-2)$. Our suggestion for practice is to use p-resistances with $p \approx p^*$. For this value of p, the p-resistances encode those global properties of the graph that are most important for machine learning, namely the cluster structure of the graph.

Our findings are interesting on their own, but also help in explaining several artifacts discussed in the literature. They go much beyond the work of von Luxburg et al. (2010) (which only studied the case $p = 2$) and lead to an intuitive explanation of the artifacts of Laplacian regularization discovered in Nadler et al. (2009). An interesting line of future research will be to connect our results to the ones about p-eigenvectors of p-Laplacians (Bühler and Hein, 2009). For $p = 2$, the resistance distance can be expressed in terms of the eigenvalues and eigenvectors of the Laplacian. We are curious to see whether a refined theory on p-eigenvalues can lead to similarly tight relationships for general values of p.

3.8 Proofs

3.8.1 Proof of Proposition 3.4

Proof of Part (1). It is easy to see that the set of all unit s-t-flows is a compact convex set and the p-norm function on this set is strictly convex for $p > 1$, thus the minimum exists. From strict convexity, this minimum is unique for $p > 1$. □

Proof of Part (2). We are going to rewrite the problems $(*)$ and $(**)$ as convex optimization problems and show that they are equivalent to the Lagrange duals of each other. We denote the $n \times m$ vertex-edge incidence matrix of a graph by Q. For any edge e_k between vertices i and j, there exist two entries in this matrix: $q_{ik} = +1$ and $q_{jk} = -1$ (flipping these signs does not make any difference in the following). Denoting the vector of edge currents by I, we can rewrite problem $(*)$ as

$$R_p(s,t)^{1/p} = \min \|I\|_{p,r} \;\text{ subject to } QI = e_s - e_t \tag{3.5}$$

where $\|\cdot\|_{p,r}$ is the r-weighted p-norm with $r_i = 1/w_i$ and e_j is the j-th unit vector. On the other hand, problem $(**)$ can be rewritten as

$$C_p(s,t)^{1/q} = \min \|Q^T u\|_{q,\overline{r}} \;\text{ subject to } u_s - u_t = 1 \tag{3.6}$$

where $\frac{1}{p} + \frac{1}{q} = 1$ and $\|\cdot\|_{q,\overline{r}}$ is the weighted q-norm with weights $\overline{r_i} = w_i^{q-1}$. The conjugate of the norm function $f = \|\cdot\|$ is

$$f^*(y) = \begin{cases} 0 & \|y\|_* \le 1 \\ \infty & \text{otherwise} \end{cases}$$

where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. Using the fact that the dual norm of $\|\cdot\|_{p,r}$ is $\|\cdot\|_{q,\overline{r}}$ with $\overline{r_i} = 1/r^{q-1}$, we can write the Lagrange dual problem of (3.5) in the minimization form as

$$R_*(v) = \min(v_s - v_t) \text{ subject to } \|Q^T v\|_{q,\overline{r}} \le 1. \tag{3.7}$$

The objective function is convex and Slater's condition holds, so we have strong duality and zero duality gap. This problem is homogeneous and the objective is linear in v, hence achieves its minimum in $\|Q^T v\|_{q,\overline{r}} = 1$. We can rewrite it as

$$R_{dual}(v) = \min(v_s - v_t) \text{ subject to } \|Q^T v\|_{q,\overline{r}} = 1. \tag{3.8}$$

If we exchange the objective function and the equality constraint in a homogeneous problem, the optimum value of the resulting problem will be the inverse of the optimum in the original problem. This means that if we form the problem

$$R'_{dual}(v) = \min \|Q^T v\|_{q,\overline{r}} \text{ subject to } (v_s - v_t) = 1 \tag{3.9}$$

and denote the solution of (3.8) by v^* and of (3.9) by v'^*, we will have

$$R_{dual}(v^*) = \frac{1}{R'_{dual}(v'^*)}. \tag{3.10}$$

As Problems (3.6) and (3.9) are identical, we can conclude

$$R_p(s,t)^{1/p} = \frac{1}{C_p(s,t)^{1/q}} \qquad \text{or} \qquad R_p(s,t) = \frac{1}{C_p(s,t)^{p/q}} = \frac{1}{C_p(s,t)^{p-1}}.$$

□

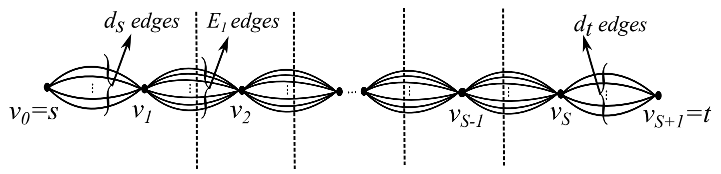


Figure 3.3: Construction of the contracted graph G' in the proof of the lower bound.

3.8.2 Proof of Theorem 3.2

Part (1). If we set $p = 1$, Problem $(*)$ coincides with the well-known linear programming formulation of the shortest path problem, see Chapter 12 of Bazaraa et al. (2010).
Part (2). For $p = 2$, we get the well-known formula for the effective resistance.
Part (3). For $p \to \infty$, the objective function in the dual problem $(**)$ converges to

$$C_\infty(s,t) := \min\Big\{ \sum_{e=(u,v)} |\varphi(u) - \varphi(v)| \;\Big|\; \varphi(s) - \varphi(t) = 1 \Big\}.$$

This coincides with the linear programming formulation of the min-cut problem in unweighted graphs. Using Proposition 3.4 we finally obtain

$$\lim_{p\to\infty} R_p(s,t)^{q-1} = \lim_{p\to\infty} \frac{1}{C_p(s,t)} = \frac{1}{C_\infty(s,t)} = \frac{1}{\text{s-t-mincut}}.$$

3.8.3 Upper and lower bounds on p-resistances

Before proving Theorem 3.5, we need to state two key propositions. In the following, recall the notation T_1 and T_2 from Section 3.4.

Proposition 3.7 (Lower bound on R_p) *Consider a family $(G_n)_{n\in\mathbb{N}}$ of unweighted geometric graphs that satisfies the general assumptions. Then for any fixed pair s,t of vertices in G_n we have*

$$R_p(s,t) \geq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + T_1.$$

Proof. Fix n. Given $G_n =: G$, we are going to construct a second graph in which the resistance between s and t is at most as large as in the original graph. We are going to exploit that Rayleigh's monotonicity principle also holds for p-resistances (proof omitted, it is an easy adaptation of the corresponding proof in Herbster and Lever, 2009). To this end, we divide the space between s and t into "slices" of width R that are perpendicular to the line connecting s and t. To build the new graph, we first add edges of resistance 0 between all neighbors of s and merge these vertices to one new "super-vertex" v_1. We also take all remaining sample points in the slice of s, except s itself, and merge them to v_1 (first adding edges of resistance 0 between all these points and then contracting them to one vertex). For all intermediate slices (the ones not containing s or t) we add edges of resistance 0 to all points in the slice and contract them to one vertex, leading to vertex v_k in slice k. In the slice containing t, we apply the analogue construction. See Figure 3.3 for an illustration. We denote the total number of slices by S, the resulting graph by $G' = (V', E')$ and the number of edges between v_j and v_{j+1} by E_j. We also set $v_0 := s$ and $v_{S+1} := t$. Note that by Rayleigh's monotonicity principle,

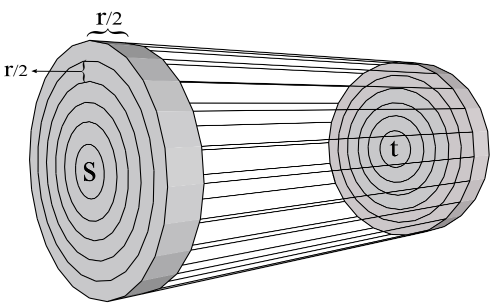


Figure 3.4: Slices and rings between s and t

the p-resistances in G and G' satisfy $R_p^G(s,t) \geq R_p^{G'}(s,t)$. Consider a unit flow $i = (i_e)_{e \in E'}$ from s to t. Denote the k-th edge between v_j and v_{j+1} by $i_{k,j}$, $(j = 0, ..., S)$. By the flow definition of the p-resistance distance we know that

$$R_p^{G'}(s,t) = \sum_{e \in E'} |i_e|^p = \sum_{j=0}^{S} \sum_{k=1}^{E_j} |i_{k,j}|^p \overset{(a)}{\geq} \sum_{j=0}^{S} \frac{1}{E_j^{p-1}} \Big(\sum_{k=1}^{E_j} |i_{k,j}| \Big)^p \overset{(b)}{=} \sum_{j=0}^{S} \frac{1}{E_j^{p-1}}.$$

Here, inequality (a) is a consequence of the generalized mean inequality (applied to the two powers p and 1). Equality (b) is a consequence of the definition of a unit flow because the sum of the flow over all edges of an $s-t$ cut is 1, and all flows from v_j to v_{j+1} are non-negative. By definition we know that $E_0 = d_s$ and $E_S = d_t$. Denoting by $E_{\max} := \max\{E_k \mid k = 1, ..., S\}$ the maximal number of edges between the intermediate vertices, we obtain

$$R_p^{G'}(s,t) \;=\; \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + \sum_{j=1}^{S-1} \frac{1}{E_j^{p-1}} \;\geq\; \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + \frac{S-1}{E_{\max}^{p-1}}$$

To bound $E_{\max}$, observe that the number of edges between v_j and v_{j+1} in G' corresponds to the number of edges of G between points in slice j and $j+1$. This number can be upper bounded by the number of edges that are attached to vertices in the smaller of the two slices. This is upper bounded by the number N of points in the slice times the maximal degree in the graph.

Under the general assumptions we have $N = \Theta(nR) = \Theta(n^{1-1/d}\tau_n^{1/d})$ and $d_{\max} = \Theta(\tau_n)$. This gives $E_{max} = \Theta(n^{1-1/d}\tau_n^{1+1/d})$. The number S of slices is of the order $1/R = \Theta((n/\tau_n)^{1/d})$. Together we get

$$\begin{aligned} R_p^G(s,t) \;&\geq\; \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + \Theta\Big(\frac{1}{\big(\tau_n/n\big)^{1/d} \big(n^{1-1/d}\tau_n^{1+1/d}\big)^{p-1}} \Big) \\ &= \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + \Theta\Big(\frac{1}{n^{p(1-1/d)-1}\tau_n^{p(1+1/d)-1}} \Big). \end{aligned}$$

□

Proposition 3.8 (Upper bound on R_p) *Consider a family $(G_n)_{n \in \mathbb{N}}$ of unweighted geometric graphs that satisfies the general assumptions. Then for any fixed pair s, t of vertices in G_n we have*

$$R_p(s,t) \leq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + T_1 + T_2.$$

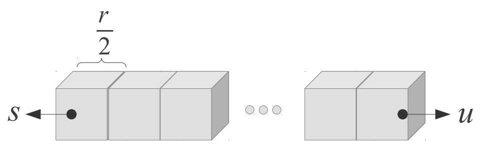


Figure 3.5: Cubes between s and u

Proof. We know from $(*)$ that the p-resistance in an unweighted graph can be upper bounded by $\sum_{e \in E} i_e^p$, where i is any unit flow from s to t. We are now going to construct a particular flow that leads to a good upper bound. We construct our flow as follows. Starting in s, we first distribute the flow to all the neighbors of s, and from there to all points in a big ring around s (see Figure 3.4). After that, every vertex in the ring sends the same amount of flow directed to the destination t. We collect all the flow at the ring around t, and in an inverse procedure to the one in the beginning we direct all flow to t itself.

In detail, consider a set of concentric rings around s, each with distance $r/2$ to the next one and width $r/2$ (see Figure 3.4). We are going to consider $\theta(1/r)$ such rings. We now distribute the unit flow starting from s uniformly through the rings: starting at s we distribute the flow to all its neighbors (leading to a contribution of $1/d_s^{p-1}$). Then we propagate the flow from the neighbors of s to all points in the inner-most circle, and then from there layer by layer to the outside.

Denote by N_k the number of sample points in ring k. As a consequence of the general assumptions, N_k is proportional to n times the volume of ring k. This volume is of the order

$$V_k = r \cdot \Big(\Theta\big((k+1)^{d-1} r^{d-1}\big) - \Theta\big(k^{d-1} r^{d-1}\big) \Big) = \Theta\big(k^{d-2} r^d\big)$$

which implies

$$N_k = nV_k = n\,\Theta(k^{d-2} r^d) = \Theta(\tau_n k^{d-2}).$$

By a similar argument, the total number of points in all rings is $N_s^{total} = \Theta(nr) = \Theta(n^{1-1/d} \tau_n^{1/d})$. By the definition of r (see general assumptions) we know that each point in ring k has on the order of $\Theta(\tau_n)$ neighbors in ring $k+1$. Every vertex in ring k sends $\Theta(1/N_k)$ flow to all its neighbors in ring $k+1$, using $\Theta(\tau_n)$ edges. The contribution of the flow between ring k and $k+1$ to the resistance is thus of the order $1/\tau_n^{2(p-1)} k^{(d-2)(p-1)}$, hence the overall contribution of the "ring part" R_s^{ring} is bounded by

$$R_s^{ring} \;\leq\; \Theta\Big(\frac{1}{\tau_n^{2(p-1)}} \sum_{k=1}^{1/r} \frac{1}{k^{(d-2)(p-1)}} \Big).$$

The analogous bound holds for the ring around t.

Now consider the contribution of the traverse flow that goes from the ring around s to the ring around t. Every vertex in the ring around s sends the same amount of flow $(1/N_s^{total})$ to its neighbors in the next slice directed to t. It follows from the definition of r that each point in slice k has on the order of $\Theta(\tau_n)$ neighbors in the slice $k+1$, so it will send $\Theta(\frac{1}{\tau_n N_s^{total}})$ flow to each of its neighbors in the next slice. Every vertex in the next slice has $\Theta(\tau_n)$ neighbors in the current slice, so it will receive $\Theta(\frac{\tau_n}{\tau_n N_s^{total}}) = \Theta(\frac{1}{N_s^{total}})$ flow. Denoting the number of edges between slice j and $j+1$ by E_j and using a similar argument to determining E_{max} in the proof of Part 1, we get $E_j = \Theta(n^{1-1/d}\tau_n^{1+1/d})$. So we have

$$\begin{aligned} R^{traverse}(s,t) &\leq \sum_{j=1}^{S}\sum_{k=1}^{E_j} |i_{k,j}|^p = \sum_{j=1}^{S} \Theta\Big(\frac{E_j}{\big(\tau_n N_s^{total}\big)^p}\Big) \\ &= \sum_{j=1}^{S} \Theta\Big(\frac{n^{1-1/d}\tau_n^{1+1/d}}{\big(n^{1-1/d}\tau_n^{1+1/d}\big)^p}\Big) = \sum_{j=1}^{S} \Theta\Big(\frac{1}{\big(n^{1-1/d}\tau_n^{1+1/d}\big)^{p-1}}\Big) \\ &= \Theta\Big(\frac{S}{\big(n^{1-1/d}\tau_n^{1+1/d}\big)^{p-1}}\Big). \end{aligned}$$

Setting $S = \Theta(1/r) = \Theta\big(n/\tau_n\big)^{1/d}$ results in

$$R^{traverse}(s,t) \leq \Theta\Big(\frac{1}{n^{p(1-1/d)-1}\tau_n^{p(1+1/d)-1}}\Big).$$

All in all, the flow sums to the following contributions

$$\begin{aligned} R_p(s,t) &\leq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + R_s^{ring} + R_t^{ring} + R^{traverse} \\ &\leq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} \\ &+ \Theta\Big(\frac{1}{\tau_n^{2(p-1)}}\sum_{k=1}^{1/r}\frac{1}{k^{(d-2)(p-1)}}\Big) + \Theta\Big(\frac{1}{n^{p(1-1/d)-1}\tau_n^{p(1+1/d)-1}}\Big). \end{aligned}$$

□

Before finally moving on to the proof of Theorem 3.5, we state a helpful little proposition that will be needed later on.

Proposition 3.9 *Consider the local neighborhood $\mathcal{N}(s)$ of vertex s and a vertex $u \in \mathcal{N}(s)$. The length of the shortest path between s and u is smaller than $2C$.*

Proof: Consider hypercubes with side length $r/2$ between s and u (see Figure 3.5). The distance between s and u is at most $C \cdot r$, so there are at most $2C$ of these hypercubes. Following the general assumptions, each vertex inside a hypercube is connected to at least one vertex in neighboring hypercubes. This means that the length of the shortest path between s and u is smaller than $2C$. □

3.8.4 Proof of Theorem 3.5

We are now going to use the bounds of Propositions 3.7 and 3.8 to prove Theorem 3.5.

Proof of the lower bound on R_p^{local}. Denote the optimal flow in the p-resistance problem by i^*. We obtain

$$R_p^{local}(s) \geq \sum_{u\sim s} |i^*_{(s,u)}|^p \geq \frac{1}{d_s^{p-1}}\big(\sum |i^*_{(s,u)}|\big)^p = \frac{1}{d_s^{p-1}}.$$

The first inequality is true because by choice of the constant C, all points adjacent to s are contained in $\mathcal{N}(s)$. The second inequality uses the the generalized mean inequality (applied to the two powers p and 1). The desired bound now follows directly from $R_p^{local}(s,t) = R_p^{local}(s) + R_p^{local}(t)$. □

Proof of the upper bound on R_p^{local}. It is enough to show that $R_p^{local}(s) < 2C$.

Let i^* denote the optimal flow in Problem $(*)$. Due to its optimality, any other unit flow i satisfies

$$R_p^{local}(s) = \sum_{\text{edge } e \in \mathcal{N}(s)} |i_e^*|^p \leq \sum_{\text{edge } e \in \mathcal{N}(s)} |i_e|^p.$$

We now want to construct a particular flow i that coincides with i^* "on the boundary of $\mathcal{N}(s)$", but which we can control inside $\mathcal{N}(s)$. To make this precise, we call a vertex $u \in \mathcal{N}(s)$ a boundary vertex if it is adjacent to some $v \notin \mathcal{N}(s)$. Denote the set of boundary vertices in $\mathcal{N}(s)$ by $\partial\mathcal{N}(s)$. For every $u \in \partial\mathcal{N}(s)$ and every unit flow i, we denote by $out_i(u)$ the sum of the flow on all edges (u,v) with $v \notin \mathcal{N}(s)$. Because i is a unit flow, it is easy to check that

$$\sum_{u \in \partial\mathcal{N}(s)} out_i(u) = 1.$$

We now want to construct a nice flow i that satisfies $\forall u \in \partial\mathcal{N}(s) : out_i(u) = out_{i^*}(u)$. To this end, for any vertex $u \in \partial\mathcal{N}(s)$ we consider a shortest path $\pi_{s,u}$ from s to u that completely stays inside $\mathcal{N}(s)$. Now we construct a flow $\tilde{i}$ as follows. We first send flow with amount $out(u)$ from s to u along $\pi_{s,u}$. For all boundary edges and edges outside of $\mathcal{N}(s)$ we define $\tilde{i}_e = i_e^*$. It is easy to check that $\tilde{i}$ is a valid unit flow from s to t.

Now we want to bound the resistance for this particular flow. We denote the j-th edge on the path $\pi_{s,u}$ by $\pi_{s,u}^j$ and define the set S_j as

$$S_j = \left\{ \pi_{s,u}^j \;\middle|\; u \in \partial\mathcal{N}(s) \right\}.$$

The set S_j contains all j-th edges on shortest paths from s to the vertices in $\partial\mathcal{N}(s)$, so we get

$$\sum_{e \in S_j} \tilde{i}_e \leq 1.$$

Now we can see that

$$R_p^{local}(s) \leq \sum_{\text{edge } e \in \mathcal{N}(s)} |\tilde{i}_e|^p \overset{(a)}{=} \sum_{j=1}^{2C-1} \sum_{e \in S_j} |\tilde{i}_e|^p \overset{(b)}{\leq} \sum_{j=1}^{2C-1} \sum_{e \in S_j} |\tilde{i}_e| < 2C.$$

Here, equality (a) is a consequence of the definition of $\tilde{i}$ and Proposition 3.9 and inequality (b) follows from $|\tilde{i}_e|^p \leq |\tilde{i}_e|$. □

Proof of the lower bound on R_p^{global}. Fix the contributions of the optimal flow i^* on the edges in $\mathcal{N}(s)$ and $\mathcal{N}(t)$ and consider the following problem:

$$\tilde{R}_p^G(s,t) := \min \left\{ \sum_{e \in E} |i_e|^p \;\middle|\; i = (i_e)_{e \in E} \text{ unit } s\text{-}t \text{ flow } ; \forall e \in \mathcal{N}(s) \cup \mathcal{N}(t) : i_e = i_e^* \right\}.$$

Obviously, the solution of this problem equals $R_p^G(s,t)$. Now we shrink the graph G except the vertices in $\mathcal{N}(s)$ and $\mathcal{N}(t)$ to a new graph G' as we did in the proof of Proposition 3.7 and consider the optimization problem

$$\tilde{R}_p^{G'}(s,t) := \min\Big\{ \sum_{e\in E'} |i_e|^p \;\Big|\; i = (i_e)_{e\in E'} \text{ unit } s\text{-}t \text{ flow };\forall e \in \mathcal{N}(s)\cup\mathcal{N}(t) : i_e = i_e^*\Big\}.$$

That is, we consider the shrunken graph, but keep the contributions of the edges in the local neighborhood of s and t fixed. By Rayleigh's monotonicity principle we get that $\tilde{R}_p^{G'}(s,t) \leq \tilde{R}_p^{G}(s,t) = R_p^G(s,t)$. By the same argument as in the proof of Proposition 3.7 we get

$$R_p^{global}(s,t) \geq \frac{S - 4CK}{E_{max}^{p-1}}$$

where $K = R/r$ is a constant. Following similar computations we obtain

$$R_p^{global}(s,t) \geq \Theta\Big(\frac{1}{n^{p(1-1/d)-1}\tau_n^{p(1+1/d)-1}}\Big).$$

□

Proof of the upper bound on R_p^{global}.
From Proposition 3.8 and the lower bound on R_p^{local} we have

$$R_p^{local}(s,t) + R_p^{global}(s,t) = R_p(s,t) \leq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}} + T_1 + T_2$$

and

$$R_p^{local}(s,t) \geq \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}.$$

Combining these two bounds gives

$$R_p^{global}(s,t) \leq T_1 + T_2.$$

□

3.8.5 Proof of Theorem 3.3

Now we can finally proceed to the proof of our main theorem.

Part (1). As a result of Theorem 3.5 we get

$$\frac{R_p^{local}(s,t)}{R_p^{global}(s,t)} < \frac{4C}{T_1}.$$

Having $p < 1 + 1/(d-1)$ allows us to set $c_2 := 1 - p(1-1/d) > 0$. Also it is easy to see that

$$p(1+\frac{1}{d}) - 1 < \frac{2}{d-1}.$$

So we can conclude that

$$\frac{R_p^{local}(s,t)}{R_p^{global}(s,t)} < \frac{4C}{T_1} < \frac{4C \cdot \tau_n^{2/(d-1)}}{n^{c_2}}.$$

This expression converges to 0 if τ_n is sub-polynomial in n (if $\tau_n/n^c \to 0$ for every fixed $c > 0$).

Part (2). From Theorem 3.5 we get

$$\frac{R_p^{local}(s,t)}{R_p^{global}(s,t)} \geq \frac{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}}{T_1 + T_2}.$$

To examine the limit behavior, recall the expression for T_2. It is easy to see that for $p > 1 + 1/(d-2)$, the harmonic sum in T_2 will converge to a constant Z and

$$T_2 \;\leq\; \Theta\Big(\frac{Z}{\tau_n^{2(p-1)}}\Big).$$

Again we can bound T_1 by

$$T_1 \;\leq\; \Theta\Big(\frac{1}{n^{1/d(d-2)}\tau_n^{(p-1)+p/d}}\Big).$$

Together we get

$$\frac{R_p^{global}(s,t)}{R_p^{local}(s,t)} \;\leq\; \Theta\Big(\frac{1}{n^{1/d(d-2)}\tau_n^{(d-1)/d(d-2)}}\Big) + \Theta\Big(\frac{1}{\tau_n^{1/(d-2)}}\Big).$$

For $n \to \infty$ and $\tau_n \to \infty$, the right hand side goes to zero and

$$\frac{R_p^{local}(s,t)}{R_p^{global}(s,t)} \to \infty.$$

Finally, to see the limit expression we rewrite Propositions 3.7 and 3.8 as

$$1 + \frac{T_1 + T_2}{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}} \geq \frac{R_p(s,t)}{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}} \geq 1 + \frac{T_1}{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}}.$$

Observing that

$$\frac{T_1 + T_2}{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}} \to 0 \quad \text{and} \quad \frac{T_1}{\frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}} \to 0$$

as $n \to \infty$ and $\tau_n \to \infty$ leads to the limit statement

$$R_p(s,t) \to \frac{1}{d_s^{p-1}} + \frac{1}{d_t^{p-1}}.$$

□

Let us make some final remarks:

It is worth to note that our results not only hold for a fixed constant C, but even if we consider a larger neighborhood around s by setting C to a sub-polynomial function of n.

In the regime $p < p^*$ we require that the degrees τ_n grow sub-polynomially in n. This condition is needed to make sure that the set $\mathcal{N}(s)$ is indeed "local" and does not cover too large a part of the graph. To get some intuition on this, just consider the case where the degrees are in the order of n, that is each vertex is connected with most other vertices in the graph. Then even single edges in the graph behave "global" as they might connect very far away parts in the graph. In this case the distinction between local and global contributions breaks down.

3.9 Excursion

This section contains a review on random walks and related concept like harmonic functions on graphs and can be skipped for a first reading. Random walks, harmonic functions on graph and potential theory are in a close relationship with the resistance distance. Here, we review these relations and then extend these concepts to p-harmonic functions and the p-resistance distance.

3.9.1 Notation

Consider an undirected weighted graph $G = \{V, E\}$ with vertex set $V = \{v_i\}_{i=1}^n$. For the sake of simplicity, we assume that G is connected. The weighted adjacency matrix is denoted by W where $w(x, y)$ indicates the similarity between vertices x and y. Let d_x denotes the degree of the vertex x, defined as

$$d_x = \sum_{u:u\sim x} w(x, u),$$

and $\mathrm{vol}(G)$ the volume of the graph, defined as $\mathrm{vol}(G) = \sum_{x\in V} d_x$.

Let M be a $n \times n$ matrix indexed by vertices in V, and $A, B \subset V$ two subsets of vertices. The matrix $M_{A,B}$ is defined as the submatrix of M with rows corresponding to vertices in A and columns corresponding to vertices in B. The element (i, j) of the matrix M is referred to as $M(i, j)$ and the j-th unit vector is denoted by e_j.

Define the degree matrix D as the diagonal matrix with entries d_x on the diagonal, and show the identity matrix by I. There are different Laplacian matrices defined in the literature, namely unnormalized Laplacian L, random walk Laplacian $\mathcal{L}$ and sysmmetric normalized Laplacian L_{sym} (von Luxburg, 2007):

$$\begin{aligned} L &= D - W \\ \mathcal{L} &= D^{-1}L = I - D^{-1}W \\ L_{\mathrm{sym}} &= D^{-1/2}LD^{-1/2} = D^{-1/2}\mathcal{L}D^{1/2} = I - D^{1/2}WD^{-1/2}. \end{aligned}$$

The pseudo-inverses (also called Moore-Penrose inverse) of a real matrix L is defined as $L^\dagger = (L^T L)^{-1}L^T$. The next proposition shows the element-wise relation between the pseudo-inverses of different Laplacian matrices.

Proposition 3.10 *The pseudoinverse of the Laplacian matrices are related element-wise as*

$$L^\dagger(i, j) = \frac{1}{d_j}\mathcal{L}^\dagger(i, j) = \frac{1}{\sqrt{d_i d_j}} L^\dagger_{sym}(i, j). \tag{3.11}$$

Proof. Using definitions of Laplacian matrices, we have

$$L^\dagger = \mathcal{L}^\dagger D^{-1} = D^{-1/2}L^\dagger_{\mathrm{sym}}D^{-1/2}.$$

Writing down the elements leads us to the result.

□

3.9.2 Random walk

Given a graph G and a starting vertex $S_0 = s$, a random walk starts at s and at each step jumps to one of the neighbor vertices at random. The jumping probability is usually related to the weight of the connected edges:

$$P_{x,y} := \mathbb{P}(S_{i+1} = y | S_i = x) = \frac{w(x,y)}{d_x}.$$

Let us denote the transition matrix by P, where $P_{i,j}$ is the probability of jumping from vertex i to j. It is easy to verify that

$$P = D^{-1}W.$$

The matrix P can be seen as an operator on functions defined on the graph's vertices:

$$Pf(x) = \sum_{v \sim x} P_{x,v} f(v).$$

We are interested in finding $\mathbb{P}(S_n = t | S_0 = s)$, which is the probability of reaching t from s in n steps. It is easy to show that the probability of reaching t in l steps is the corresponding element of P^l

$$\mathbb{P}(S_n = t | S_0 = s) = [P^l](s,t) = e_s^T P^l e_t.$$

For a random walk starting from s, denote the total number of visits to t by $N(s,t)$. The expected number of visits can be written as

$$\mathbb{E}[N(s,t)] = \sum_{i=0}^{\infty} \mathbb{P}(S_i = t | S_0 = s) = e_s^T \sum_{i=0}^{\infty} P^i e_s.$$

In finite undirected graphs, the total number of visits is unbounded and diverges as time goes to infinity. This can also verified by observing the spectrum of P, that has an eigenvalue 1. However, in infinite graphs like the d-dimensional lattice ($d > 2$), the expected total number of visits is bounded (see also the discussion in Section 3.9.3).

3.9.3 Recurrency and hitting time

The **probability of eventual return** is defined as the probability that a random walk starting from s eventually returns to s. A random walk is called recurrent if the probability of its eventual return to the origin is 1. A random walk that is not recurrent is called transient.

The recurrence of a random walk is in a close relation to its expected number of returns to the origin. By means of the Borel-Cantelli lemma, a random walk is recurrent if and only if its expected number of returns is infinite. However there exists another simple proof.

Theorem 3.11 *A random walk starting from vertex s is recurrent if and only if its expected number of returns to s is infinity.*

Proof. We denote the probability of eventual return to the origin by p and the number of returns by N_s. Then the probability that the random walk returns back only k times is $\mathbb{P}(N_s = k) = p^{k-1}(1-p)$ and

$$\mathbb{E}[N_s] = \sum_{k=1}^{\infty} k\mathbb{P}(N_s = k) = \sum_{k=1}^{\infty} kp^{k-1}(1-p) = \frac{1}{1-p}.$$

It is clear that $p = 1$ if and only if $\mathbb{E}[N_s] = \infty$.

$\square$

In finite undirected connected graphs, the probability of eventual return to each vertex is 1 and the random walk is recurrent. The recurrence in infinite graphs is a challenging problem. Polya studied this problem for d-dimensional lattices and showed that a simple random walk is recurrent for $d = 1, 2$ and transient for $d \geq 3$.

Hitting time: The hitting time $H(s,t)$ is defined as the expected number of steps needed for a random walk starting from s to reach t for the first time. One can show that

$$\frac{1}{vol(G)}H(s,t) = e_t^T L^\dagger (e_t - e_s) = \frac{e_t^T}{\sqrt{d_t}} L_{\text{sym}}^\dagger \left(\frac{e_t}{\sqrt{d_t}} - \frac{e_s}{\sqrt{d_s}}\right) = \frac{(e_t - e_s)^T}{d_t} \mathcal{L}^\dagger e_t.$$

The **commute time** between s and t is defined as $C(s,t) = H(s,t) + H(t,s)$ and can be computed as

$$\begin{aligned} \frac{1}{vol(G)}C(s,t) &= (e_s - e_t)^T L^\dagger (e_s - e_t) \\ &= \left(\frac{e_s}{\sqrt{d_s}} - \frac{e_t}{\sqrt{d_t}}\right)^T L_{\text{sym}}^\dagger \left(\frac{e_s}{\sqrt{d_s}} - \frac{e_t}{\sqrt{d_t}}\right) \\ &= (e_s - e_t)^T \mathcal{L}^\dagger \left(\frac{e_s}{d_s} - \frac{e_t}{d_t}\right). \end{aligned}$$

3.9.4 Harmonic functions on graphs

A function $f : V \to \mathbb{R}$ defined on vertices of a graph $G = \{V, E\}$ is called harmonic on $A \subset V$, if for every $x \in A$, the value of $f(x)$ equals the average of its neighbors, weighted by edge weights:

$$f(x) = \sum_{v:v\sim x} P_{x,v} f(v),$$

or equivalently

$$\sum_{v:v\sim x} P_{x,v}\big(f(x) - f(v)\big) = 0.$$

The *Laplace operator* (matrix) on G is defined as

$$\Delta f(x) = \sum_{v:v\sim x} P_{v,x}\big(f(x) - f(v)\big).$$

It is easy to verify that $\Delta = \mathcal{L} = I - P$. The notation $\mathcal{L}$ is often used for denoting the Laplace operator on graphs, while Δ is common in continuous domains. We use Δ for graphs in equations that have a well-known continuous counterpart.

A function is harmonic on $A \subset V$ if $\forall x \in A; \Delta f(x) = 0$. The boundary nodes are the set of nodes that f is not harmonic on them $\partial A = V \setminus A$. We always assume that each boundary vertex $v \in \partial A$ has a neighbor in A, and the induced sub-graph A is connected.

Harmonic extension: Consider a function $\bar{f}$ defined on an nonempty boundary set $\partial A \subset V$. A harmonic function on A is called the harmonic extension of $\bar{f}$ on G. We can reformulate the harmonic extension problem as finding f such that

$$\begin{cases} \Delta f(x) = 0 & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A. \end{cases} \tag{3.12}$$

The Equation 3.12 is called *Laplace equation.* This equation has a unique solution, which is proved in Proposition 3.14. First we show that the minimum and the maximum of non-constant harmonic functions are at their boundary nodes.

Proposition 3.12 (Maximum principle) *In finite graphs, every non-constant harmonic function achieves its minimum and maximum at the boundary nodes.*

Proof. The maximum of a function can not be attained at a node on which it is harmonic, unless it gets the same value at its neighbors. The same holds for the minimum. $\square$

Energy minimization: The harmonic extension problem 3.12 can also be written in an equivalent energy minimization form

$$\begin{aligned} \text{minimize} \qquad & E(f) = \sum_{x,y:x\sim y} P_{x,y}\big(f(x) - f(y)\big)^2 \\ \text{subject to} \qquad & f(x) = \bar{f}(x), \quad x \in \partial A. \end{aligned}$$

The objective function is sometimes referred as the discrete Dirichlet energy of function f. The equivalence can be proved by taking the derivatives of the objective function with respect to the value of f at each vertex and setting them to zero.

Proposition 3.13 *The discrete Dirichlet energy functional $E(f)$ can be written as $\langle f, \Delta f\rangle$, where Δ is the Laplace operator (matrix). In another word,*

$$\langle f, \Delta f\rangle = \frac{1}{2} \sum_{x,y:x\sim y} P_{x,y}(f(x) - f(y))^2.$$

Discrete Dirichlet problem: The harmonic extension problem is a special case of the discrete Dirichlet problem. Given a function g defined on A and the boundary value function $\bar{f}$ defined on ∂A, the discrete Dirichlet problem is to find a function f on the graph nodes such that

$$\begin{cases} \Delta f(x) = g(x) & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A. \end{cases} \tag{3.13}$$

We would discuss two different views on the discrete Dirichlet problem. Below is the relation to random walks. In Section 3.9.7 it would also appear in the context of electrical networks.

Random walk interpretation: Consider an agent that starts a random walk from a vertex $x \in V$ and stops as soon as it reaches a boundary vertex. The agent pays $g(u)$ whenever it passes through an internal node $u \in A$ and pays $\bar{f}(v)$ when it stops at a boundary node $v \in \partial A$. The expected amount of the money payed by the agent before it stops is a solution of the Equation 3.13 at x. The Dirichlet problem has always a unique solution if the boundary is not empty.

Proposition 3.14 (Dirichlet problem has a unique solution) *Consider a finite graph $G = \{V, E\}$, a nonempty boundary set $\partial A \subset V$ and a function g defined on A. Then there exist a unique solution to the Dirichlet problem with boundary value $\bar{f}$.*

Proof. The existence of a solution is a result of the random walk interpretation, as the expected value always exists. For the proof of uniqueness, consider two different solutions f_1 and f_2. Then $f_0 = f_1 - f_2$ is a solution to the problem

$$\begin{cases} \Delta f_0(x) = 0 & x \in A \\ f_0(x) = 0 & x \in \partial A. \end{cases}$$

Following Proposition 3.12, the solution f_0 achieves its maximum and minimum at ∂A. However, $\bar{f}_0(x) = 0$ for all $x \in \partial A$. This shows that f_0 is the constant 0 function and $f_1 = f_2$. □

3.9.5 Laplace equation and the Poisson kernel

Fix a non-empty boundary set ∂A and assume that it has c nodes. For each function $\bar{f}$ on ∂A (a vector in $\mathbb{R}^c$), there exists a unique function f (a vector in $\mathbb{R}^{n-c}$) which is the harmonic extension $\bar{f}$ on G . The next proposition states that this function (the solution of the Lapalce equation 3.12) is a linear function of the boundary value function $\bar{f}$.

Proposition 3.15 (The linearity of the Laplace problem) *Fix a boundary set ∂A. Consider two functions $\bar{f}_1$ and $\bar{f}_2$ defined on ∂A and let f_1 and f_2 be the solutions to the corresponding Laplace equations. Then for every $\alpha, \beta \in \mathbb{R}$, the Laplace problem on ∂A with boundary value $\bar{f} = \alpha \bar{f}_1 + \beta \bar{f}_2$ has the solution $f = \alpha f_1 + \beta f_2$.*

Proposition 3.15 is called the superposition principle in the electrical network theory. The proof is a straightforward result of the linearity of the equations.

Any linear transformation from $\mathbb{R}^c$ to $\mathbb{R}^{n-c}$ can be expressed as an $(n-c) \times c$ matrix. This means that we can write the solution of the Laplace problem as

$$f = Z\bar{f}.$$

The matrix Z depends on the underlying graph and the boundary set ∂A, but not on the boundary value function $\bar{f}$. This matrix (operator) is often called the *Poisson kernel.* If we set the boundary values to one, then the harmonic extension would be the constant one vector. So we infer that

$$Z\mathbf{1}_c = \mathbf{1}_{n-c}.$$

This means that $\sum_{j=1}^{c} Z(i,j) = 1$. This property can also be justified from a probabilistic interpretation of the Poisson kernel.

Proposition 3.16 (Probabilistic interpretation of the Poisson kernel) *The element $Z(i,j)$ of the Poisson kernel Z can be interpreted as the probability that a random walk starting from i hits the boundary for the first time at j.*

Proof. Consider the boundary value function $\bar{f} = e_j$ which takes the value 1 at node j and 0 at other boundary nodes. The harmonic extension of u to G would correspond to the j-th column of the Poisson kernel Z, so $f(i) = Z(i,j)$. Using the proof of Proposition 3.14, $f(i)$ is the expected value of the function $\bar{f}$ observed by a random walk agent starting from node i, at the first time it hits the boundary.

By the construction of $\bar{f}$, it has value 0 at all boundary nodes except j. This means that $Z(i,j)$ is the probability that a random walk starting from i hits the boundary for the first time at j. □

Computing the Poisson kernel: The Poisson kernel can be computed by solving a system of linear equations. Without loss of generality, assume that vertices $v_{n-c+1}, .., v_n$ are boundary vertices. Then the Equation 3.12 can be written as

$$\mathcal{L}_{A,V} \begin{bmatrix} f(v_1) \\ \vdots \\ f(v_{n-c}) \\ \bar{f}(v_{n-c+1}) \\ \vdots \\ \bar{f}(v_n) \end{bmatrix} = 0 \tag{3.14}$$

or equivalently

$$\mathcal{L}_{A,A} f = -\mathcal{L}_{A,\partial A} \bar{f}. \tag{3.15}$$

One can show that the submatrix of the random walk Laplacian $\mathcal{L}_{A,A}$ is always invertible (when ∂A is not empty), so the Poisson kernel is

$$Z = \mathcal{L}_{A,A}^{-1} \mathcal{L}_{A,\partial A}.$$

3.9.6 Dirichlet equation and the Green's function

Before discussing the solution of discrete Dirichlet problem, we consider a special case where the boundary value function is 0 everywhere:

$$\begin{cases} \Delta \bar{f}(x) = g(x) & x \in A \\ \bar{f}(x) = 0 & x \in \partial A. \end{cases} \tag{3.16}$$

This problem has an interesting probabilistic interpretation which is discussed below. Similar to Propositions 3.14 and 3.15, we can show the uniqueness and the linearity of the solution. Following the same line as in Section 3.9.5, the solution, which is a vector in $\mathbb{R}^{n-c}$, can be obtained from a linear transformation of the input, which is again a vector in $\mathbb{R}^{n-c}$. Every such linear transformation can be written as a matrix, so

$$f = Gg. \tag{3.17}$$

The transformation matrix (operator) G is the *Green's function* of graph G with boundary nodes ∂A. The Green's function only depends on the graph structure and the set of boundary nodes. The element $G(i,j)$ of the Green's function has an interesting random walk interpretation:

Proposition 3.17 (Random walk interpretation of the Green's function) *Consider a random walk starting from vertex i. The element $G(i,j)$ of the Green's function can be interpreted as the expected number of visits to j before hitting the boundary ∂A.*

Proof. Set $g = e_i$, where it takes 1 on the vertex i and 0 everywhere else. Then the solution of the Dirichlet problem 3.16 is $f(j) = G(i,j)$. Following the random walk interpretation of the Dirichlet problem (see Section 3.9.4), the random walk starting from i will only pay whenever it passes from j, and stop as soon as it hits the boundary. This means that $f(j) = G(i,j)$ is the expected number of visits to j before hitting the boundary. □

Computing the Green's function: The Green's function can be computed by solving a system of linear equations similar to Equation 3.14

$$\mathcal{L}_{A,V} \begin{bmatrix} f(v_1) \\ \vdots \\ f(v_{n-c}) \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} g(v_1) \\ \vdots \\ g(v_{n-c}) \end{bmatrix} \tag{3.18}$$

or

$$\mathcal{L}_{A,A} f = g. \tag{3.19}$$

As the submatrix of Laplacian $\mathcal{L}_{A,A}$ is always invertible, the Green's function can be computed by $G = \mathcal{L}_{A,A}^{-1}$.

Solving the general Dirichlet problem: We have the solution to the Laplace Equation and the Green's function. By the linearity of the Dirichlet problem, combining the solutions of Equations 3.14 and 3.18 solves the general Dirichlet problem. The final solution is

$$f = -\mathcal{L}_{A,A}^{-1}\mathcal{L}_{A,\partial A}\bar{f} + \mathcal{L}_{A,A}^{-1}g = \mathcal{L}_{A,A}^{-1}(g - \mathcal{L}_{A,\partial A}\bar{f}) \tag{3.20}$$

Dirichlet problems with empty boundary set

In general, a Dirichlet problem with empty boundary set does not have any solution. This problem has only a solution for a specific family of functions g on A. The solution is not unique anymore, but all solutions only differ by a constant vector.

Proposition 3.18 (Dirichlet problem with empty boundary) *Denote the degrees of vertices in A by the vector $\mathbf{d}$. A Dirichlet problem with empty boundary set has a solution if and only if $\langle g, \mathbf{d} \rangle = 0$. Then, the Dirichlet problem has a set of solutions*

$$\{f | f = \mathcal{L}^{\dagger} g + a\mathbf{1}_n; a \in \mathbb{R}\}.$$

Proof. Assume that f is a solution to the Dirichlet problem, $\mathcal{L}f = g$. For a connected subgraph A, the degree vector $\mathbf{d}$ is the only left eigenvector of $\mathcal{L}$ with eigenvalue 0. This means that $\mathbf{d}^T g = \mathbf{d}^T \mathcal{L} f = 0$. Also for a vector g orthogonal to the nullspace of $\mathcal{L}$, the equation $\mathcal{L}f = g$ has at least one solution. The constant vector $\mathbf{1}_n$ is the right eigenvector of $\mathcal{L}$ with eigenvalue 0 and multiplicity one. This shows that $f^* = \mathcal{L}^\dagger g$ is one solution of the equation, and the set $\{f | f = f^* + a\mathbf{1}_n; a \in \mathbb{R}\}$ indicates the set of all solutions. □

3.9.7 Electrical networks

In this section we review the basic concepts in electrical networks and the connection of these concepts to random walks on graphs. It turns out that many different problems related to random walks are in analogy to problems in electrical networks. These analogy problems are sometimes more intuitive and easier to understand. The Dirichlet problem has an interesting analogy in electrical networks, which is discussed in Section 3.9.7.

Flow on a graph

A flow on a graph is a function $\mathbf{i} = (i_{uv})$ defined on the graph edges such that $i_{uv} = -i_{vu}$. The net in-flow of a vertex v is defined as $\overrightarrow{i_v} = \sum_{u\sim v} i_{vu}$. The net out-flow of a vertex is defined accordingly $\overleftarrow{i_v} = \sum_{u\sim v} i_{uv} = -\overrightarrow{i_v}$. It is easy to verify that the total net in(out)-flow of a graph G is 0, $\sum_{v\in V} \overrightarrow{i_v} = 0$. A vertex s is called a source if $\overrightarrow{i_s} > 0$, and a vertex t is called a sink if $\overleftarrow{i_t} > 0$. We call the remaining nodes with in(out)-flow zero as internal nodes.

An interesting set of flow functions are the ones that run from a single source vertex s to a single sink vertex t. Every other node v in the graph has the flow conservation property: $\sum_{u\sim v} i_{vu} = 0$. We call such a flow as a s-t flow. The total in-flow of a network is zero, which shows that $\overrightarrow{i_t} = -\overrightarrow{i_s}$. When $\overrightarrow{i_s} = 1$, the flow is called a unit s-t flow. If we denote the vertex-edge incidence matrix of the graph by Q, every unit s-t flow satisfies $Q\mathbf{i} = (e_s - e_t)$.

Electrical flow in a network

From a graph G, build an electrical network where each edge e has resistance $r_e = 1/w_e$. Connect the node s to a 1-Amper current source and the sink node t to the ground (set its voltage to 0). Then we will have a unit s-t flow in the network, and the out-flow of vertex t is 1. The unit s-t *electrical* flow $\mathbf{i}^*$ has a unique property that distinguishes it from the set of all unit s-t flows: We can assign a voltage(potential) function $\mathbf{v} : V \to \mathbb{R}$ such that Ohm's law holds for every edge:

$$v(a) - v(b) = \mathbf{i}_{ab} r_{ab}.$$

Proposition 3.19 *Assume* $\mathbf{i}$ *is a s-t flow and a voltage(potential) function* $\mathbf{v}$ *on nodes satisfies the Ohm's law on every edge. Then* $\mathbf{v}$ *is harmonic on* $V \setminus \{s,t\}$.

Remark 3.20 *If* $\mathbf{v}$ *is harmonic on vertex x, it satisfies both* $\Delta\mathbf{v}(x) = 0$ *and* $L\mathbf{v}(x) = 0$. *This is because* $\Delta = D^{-1}L$. *Note that the condition* $\Delta\mathbf{v}(x) = g(x)$ *is equivalent to* $L\mathbf{v}(x) = d_x g(x)$.

Another characteristic of an electrical flow is that it minimizes the power dissipated in the network

$$\mathrm{Pow}(\mathbf{i}) := \sum_{e=(a,b)\in E} \big(v(a) - v(b)\big) i_{ab}, \tag{3.21}$$

or equivalently

$$\mathrm{Pow}(\mathbf{i}) = \sum_{e=(a,b)\in E} r_{ab} i_{ab}^2 = \sum_{e=(a,b)\in E} \frac{\big(v(a)-v(b)\big)^2}{r_{ab}}. \tag{3.22}$$

It is easy to show that these two characteristics are equivalent: if a unit s-t flow $\mathbf{i}$ minimizes the power, then we can assign a voltage function $\mathbf{v}$ such that the Ohm's law holds. If for a unit s-t flow $\mathbf{i}$, we could find a corresponding voltage function $\mathbf{v}$, then the flow $\mathbf{i}$ minimizes the power dissipated in the network.

Effective resistance: The effective resistance (or resistance distance) $R(s,t)$ between two vertices s and t in the network is the resistance measured between s and t. We can replace the network between s and t by a single resistor with resistance $R(s,t)$. The resistance distance can be computed by minimizing the energy dissipated in the network (Equation 3.21) from a unit flow:

$$R(s,t) = \min\Big\{ \sum_{e\in E} r_e i_e^2 \;\Big|\; i = (i_e)_{e\in E} \text{ unit flow from } s \text{ to } t \Big\}. \tag{3.23}$$

We can intuitively check the correctness of this formula. A unit electrical flow from s to t is the unit s-t flow that minimizes the power dissipation $\mathrm{Pow}(\mathbf{i})$. Now replace the network with an equivalent resistor between s and t with resistance $R(s,t)$. Then we have

$$\mathrm{Pow}(\mathbf{i}) = R(s,t)\overrightarrow{i_s}^2 = R(s,t).$$

The resistance distance can also derived from the energy dissipation function in Equation 3.22:

$$C(s,t) = \min\Big\{ \sum_{e=(u,v)} \frac{(v(a)-v(b))^2}{r_e} \mid v(s)-v(t) = 1 \Big\} \tag{3.24}$$

Then $R(s,t) = 1/C(s,t)$. Again we can intuitively check the correctness of this formula. Replace the network with an equivalent resistor between s and t with resistance $R(s,t)$. As we have a unit voltage drop,

$$C(s,t) = \mathrm{Pow}(\mathbf{i}) = (v(s)-v(t))^2/R(s,t) = 1/R(s,t).$$

An alternative method for finding the effective resistance $R(s,t)$ is the following: Assume that the current 1 is injected into the vertex s and consequently goes out from vertex t. The voltage vector $\mathbf{v}$ satisfies

$$L\mathbf{v} = e_s - e_t. \tag{3.25}$$

This equation has many solutions, but they all differ by a constant vector. This can also verified using the fact that L has eigenvalue 0 with multiplicity 1 and constant eigenvector (see also Section 3.9.6). If $\mathbf{v}^*$ satisfies Equation 3.25, then $R(s,t) = v^*(s) - v^*(t)$. One such solution is $\mathbf{v}^* = L^\dagger(e_s - e_t)$, so

$$R(s,t) = (e_s - e_t)^T L^\dagger (e_s - e_t). \tag{3.26}$$

Another way to find the solution of Equation 3.25 is by setting the voltage at node t to 0. If we set $A = V \setminus \{t\}$, then the solution to Equation 3.25 also satisfies $L_{A,A}\mathbf{v}_A = (e_s)_A$. As the matrix $L_{A,A}$ is non-singular, the last equation has a unique solution $\mathbf{v}_A^* = L_{A,A}^{-1}(e_s)_A$ and

$$R(s,t) = v_s^* - v_t^* = v_s^* = L_{A,A}^{-1}(s,s). \tag{3.27}$$

Dirichlet problem and electrical networks

The Dirichlet problem has an interesting interpretation in the context of electrical networks. Using the notation from Equation 3.13, the boundary function $\bar{f}$ can be interpreted as connecting each boundary vertex $v \in \partial A$ to a voltage source $\bar{f}(v)$. Also the function g can be seen as connecting each internal node $v \in A$ to a current source with in-flow $g(v)$. The problem is then to find the induced voltages on nodes in A.

Proposition 3.21 *We are given a resistive network $G = \{V, E\}$. The graph vertices are divided into two sets $V = A \cup \partial A$, where every vertex in ∂A is connected to at least one vertex in A. Each vertex $v \in \partial A$ is connected to a voltage source $\bar{f}(v)$ and each vertex $v \in A$ is connected to a current source with in-flow $g(v)$. Then the voltages of nodes in A are determined by solving*

$$\begin{cases} Lf(x) = g(x) & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A. \end{cases} \tag{3.28}$$

This problem is equivalent to the Dirichlet problem

$$\begin{cases} \Delta f(x) = \bar{g}(x) & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A \end{cases} \tag{3.29}$$

with $\bar{g} = D_{A,A}^{-1} g$.

Proof. It simply follows from the Ohm's law. □

Resistance and commute time

The following theorem states the relation between the resistance distance and the commute time, first proved by Chandra et al. (1989).

Theorem 3.22 *For any two vertex s and t, the commute time equals to*

$$C(s,t) = 2\,\mathrm{vol}(G) R(s,t).$$

The next proposition relates the resistance distance and the frequency of return to the source before hitting the sink vertex:

Proposition 3.23 *Denote the number of times a random walk starting from s returns to s before visiting t by $n_{s,t}$. Then*

- $n_{s,t} = R(s,t) \times d_s$.
- $n_{s,t} \times d_t = n_{t,s} \times d_s$

Proof. *Part a)* By the mean of Proposition 3.21, the resistance distance can be written as $R(s,t) = f(s) - f(t)$ where f is an arbitrary solution of the Dirichlet problem with an empty boundary:

$$\begin{cases} Lf(s) = 1 \\ Lf(t) = -1 \\ Lf(x) = 0 & x \in V \setminus \{s,t\} \end{cases}$$

This equation has many solutions, all differ by a constant vector. We can fix to one of them by setting $f(t) = 0$. If we set t as a single boundary node, then the condition $Lf(t) = -1$ will hold automatically. The reason can be seen from electrical network interpretation: the total in-flow of the network is zero and $Lf(s) = 1$. The vertex t is the only vertex that has a non-zero out-flow, so $Lf(t) = -1$. This means that the resistance distance can be written as $R(s,t) = f(s)$ where f is the solution of following Dirichlet problem

$$\begin{cases} Lf(x) = e_s(x) & x \in V \setminus \{t\} \\ f(t) = 0 \end{cases}$$

or equivalently

$$\begin{cases} \Delta f(x) = d_s^{-1} e_s(x) & \in V \setminus \{t\} \\ f(t) = 0 \end{cases}$$

Now using Proposition 3.17 shows that $R(s,t) = n_{s,t}/d_s$.

Part b) If we repeat the procedure in Part a by setting $f(s) = 0$, we get $R(s,t) = n_{t,s}/d_t$. Putting together, we have $n_{s,t} \times d_t = n_{t,s} \times d_s$ which finishes the proof. □

This proposition helps us in analyzing the recurrence of a random walk. Consider a random walk on an infinite graph that starts from vertex s. Connect all vertices outside a "ring" with (shortest path) distance r from s together and call it t. As r goes to infinity, $n_{s,t}$ will converge to the expected number of returns to the origin. This means that the effective resistance between s and infinity $R_{\text{eff}}(s)$ equals to the expected number of returns to the origin divided by d_s. On the other hand, Theorem 3.11 shows that a random walk is recurrent if and only if the expected number of returns to the origin is infinite. So we proved the following theorem:

Theorem 3.24 *A random walk is recurrent if and only if the resistance between a vertex and infinity is infinite.*

3.9.8 p-harmonic functions

A function $f : V \to \mathbb{R}$ defined on vertices of a graph $G = \{V, E\}$ is called p-harmonic ($p > 1$) on $A \subset V$ if for every $x \in A$,

$$\sum_{v:v\sim x} P_{x,v} |f(x) - f(v)|^{p-2} \big(f(x) - f(v)\big) = 0,$$

or equivalently

$$\sum_{v:v\sim x} P_{x,v} |f(x) - f(v)|^{p-1} \operatorname{sgn}\big(f(x) - f(v)\big) = 0.$$

In the first definition, we make a convention that $|f(x) - f(v)|^{p-2}\big(f(x) - f(v)\big) = 0$ for $f(x) = f(v)$.

Accordingly, the *p-Laplace operator* on G is defined as

$$\Delta_p f(x) = \sum_{v:v\sim x} P_{x,v} |f(x) - f(v)|^{p-2} \big(f(x) - f(v)\big).$$

Discrete p-Dirichlet problem: Given a function g defined on A and the boundary value function $\bar{f}$ defined on ∂A, the discrete p-Dirichlet problem is to find a function f on graph nodes such that

$$\begin{cases} \Delta_p f(x) = g(x) & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A. \end{cases} \tag{3.30}$$

Similar to the case $p = 2$, the p-harmonic extension problem is defined accordingly:

$$\begin{cases} \Delta_p f(x) = 0 & x \in A \\ f(x) = \bar{f}(x) & x \in \partial A. \end{cases} \tag{3.31}$$

The p-harmonic extension problem for $p \neq 2$ is a non-linear system of equations and we can not use the standard algorithms such as Gaussian elimination to solve this set of nonlinear equations. Alternatively, we can solve an equivalent optimization problem which minimizes the p-Dirichlet energy function

$$\begin{aligned} \text{minimize} \qquad & E_p(f) = \sum_{x \sim y} P_{x,y} |f(x) - f(y)|^p \\ \text{subject to} \qquad & f(x) = \bar{f}(x), \quad x \in \partial A. \end{aligned}$$

This optimization can solved "efficiently" for $p \in \{1, 2, \infty\}$. For other values of p, we need to use conventional convex optimization methods such as gradient descent or quasi-Newton methods.

The Maximum principle (see Proposition 3.12) also holds for the p-harmonic extension problem:

Proposition 3.25 *In finite graphs, every non-constant p-harmonic function ($p > 1$) achieves its minimum and maximum at the boundary nodes.*

Proof. The maximum of a function can not be attained at a node on which it is p-harmonic, unless it gets the same value at its neighbors. The same holds for the minimum. □

3.9.9 p-resistance: an electrical network interpretation

The p-resistance can be interpreted as a nonlinear resistance in an electrical network. Consider the p-resistor as a type of resistor that for voltage difference $v(a)-v(b)$ lets through the current i_{ab} with

$$v(a) - v(b) = r_{ab} |i_{ab}|^{p-1} \operatorname{sgn}(i_{ab}). \tag{3.32}$$

Consider a network of p-resistors with unit current from source s to sink t. Similar to the linear resistor networks, one can determine the voltages in this network by writing the flow conservation properties for each node, and reformulating it based on voltages

$$i_{ab} = \frac{1}{r_{ab}} |v(a) - v(b)|^{1/p-1} \operatorname{sgn}\big(v(a) - v(b)\big).$$

Then we end up with the following equations

$$\begin{cases} \Delta_q v(x) = 0 & v \in V \setminus \{s, t\} \\ \Delta_q v(s) = 1 \\ \Delta_q v(t) = -1, \end{cases} \tag{3.33}$$

with $q = 1 + 1/(p-1)$. This is the q-Dirichlet problem (q is the dual of p, $1/p + 1/q = 1$) with empty boundary set. If v^* is a solution of this problem, then the p-resistance between s and t is equal to $R_p(s,t) = v^*(s) - v^*(t)$. The solution v^* is unique up to addition with a constant vector (see Proposition 3.4).

An alternative method for computing the p-resistance is to minimize the power dissipation function, written as a function of currents

$$\mathrm{Pow}_p(\mathbf{i}) := \sum_{e=(a,b)\in E} \big(v(a) - v(b)\big)\mathbf{i}_{ab} = \sum_{e\in E} r_e |\mathbf{i}_e|^p$$

over all unit flows from s to t. Then replace the whole network with an equivalent resistor $R_p(s,t)$ to get $\mathrm{Pow}_p(\mathbf{i}) = R_p(s,t)|\mathbf{i}_{s,t}|^p = R_p(s,t)$. In another word, we can find the p-resistance between s and t by solving

$$R_p(s,t) = \min\Big\{ \textstyle\sum_{e\in E} r_e |i_e|^p \;\big|\; i = (i_e)_{e\in E} \text{ unit flow from } s \text{ to } t \Big\}, \tag{3.34}$$

which was used as the definition of the p-resistance distance in Section 3.2.

Another alternative for finding the p-resistance is to minimize the power dissipation function, written as a function of voltages

$$R_p(s,t) = \mathrm{Pow}_p(\mathbf{i}) := \sum_{(a,b)\in E} \big(v(a) - v(b)\big)\mathbf{i}_{ab} = \sum_{(a,b)\in E} \frac{|v(a) - v(b)|^{1+\frac{1}{p-1}}}{r_{ab}^{\frac{1}{p-1}}},$$

over all unit flows from s to t. This formulation is related to the Lagrange dual of Equation 3.34 and can be written in an optimization form

$$\min\Big\{ \sum_{e=(a,b)} \frac{|v(a) - v(b)|^{1+\frac{1}{p-1}}}{r_{ab}^{\frac{1}{p-1}}} \;\big|\; v(s) - v(t) = 1 \Big\}.$$

For more details on deriving the Lagrange dual of Equation 3.34, see also Proposition 3.4.

Part II

Local Distances in Graphs

Chapter 4

Local clustering in graphs

4.1 Introduction

Graph clustering is an omnipresent problem in machine learning. Given a particular graph, the goal is to find "clusters" or communities in the graph such that the vertices in the same community are highly connected to each other and only sparsely connected to vertices outside of their community. A big problem in practice is that graphs often have an enormous size, which makes the application of standard graph clustering algorithms such as spectral clustering infeasible.

A promising alternative is represented by the class of local clustering algorithms. Here the goal is not to find a global clustering of the whole graph, but to find "the cluster" of a particular vertex of interest. For example, in social network analysis one might want to investigate the community a particular person belongs to. The advantage of a local clustering algorithm is that we never consider the graph as a whole. All computations are local in the sense that we only need to have access to and operate on a small "local" part of the adjacency matrix, which leads to efficiency in space and time complexity of the algorithm. The price we pay is that we cannot guarantee that the cluster found by the algorithm is a true, global cluster of the graph; we can only say that it looks like a good cluster based on the local information we have seen.

Recently there has been a lot of interest in local algorithms in the data mining and machine learning community. One of the most popular local clustering algorithms is the Nibble algorithm described in Spielman and Teng (2008). Given the vertex v of interest, one considers a random walk (RW) on the graph that starts at vertex v. This random walk is biased in the sense that it only jumps along "important" edges, unimportant edges (with low weight) are ignored. The random walk runs until a certain stopping criterion is met, for example when the conductance of the set of visited vertices gets large enough. The set of visited vertices is then considered as a local cluster around v. For further developments based on the nibble algorithm see Andersen et al. (2006) and Andersen and Peres (2009). Random walk based clustering algorithms are used in applications like image segmentation (Grady, 2006, Pan et al., 2004).

The reason why the random walks is a well-suited tool for local clustering is obvious: they locally explore the neighborhood of a vertex and with reasonably high probability stay inside a cluster rather than transitioning to a different cluster. Moreover, they can be implemented with low space complexity.

We suggest to use a novel kind of random walk for local exploration, called multi-agent random walk (MARW). Consider a “agents”, each agent moving like an individual random walk on the graph. The agents are tied together by a “rope” of length l. As long as the distance between all agents is smaller than l, all of them can move independently from each other. However, they are constrained such that the distance between any two agents is never larger than l.

The reason why the MARW is advantageous for local clustering is that the probability of transitioning between two different clusters is significantly reduced in the MARW compared to the standard RW. Consequently, the MARW discovers tighter clusters than the standard RW.

To the best of our knowledge, the idea of studying a MARW is completely new. Only after we had started working on this question, a group of mathematicians submitted two papers on arxiv that study a similar object called “spider walk”, but from a completely different perspective (Gallesco et al., 2009, 2010). Another related paper is the one by Alon et al. (2008), which has an opposite intention: they want to construct an alternative to the standard random walk that travels faster between different clusters. To this end, the authors run several random walks independent from each other. In this way one can speed up the global exploration of large graphs. This setting can also be covered in our model, namely as the extreme case of having rope of length infinity. See below for further discussion.

The goal of this chapter is to introduce the multi-agent random walk and study some of its basic properties. We present several theoretical approaches that show that if we consider a MARW with a agents coupled by a small rope length l, then the probability of transitioning between different clusters decreases with the power of a: if this probability is p for the standard RW, it is about p^a for the MARW. We show in experiments on toy data and on real world data that MARWs are well-suited for local clustering indeed. In particular, they outperform the state-of-the-art algorithms by Spielman and Teng (2008) and Andersen et al. (2006). In general, we believe that the technique of multi-agent random walk can be very fruitful in several domains of data mining and machine learning.

4.2 The multi-agent random walk

Let $G = (V, E)$ be a weighted graph with weighted adjacency matrix $W = (w_{ij})_{ij}$. The graph can be directed or undirected. Denote the degree of vertex v_i by $d_i = \sum_j w_{ij}$. Let the degree matrix D be the diagonal matrix with entries d_i. The simple RW on G is defined as the random walk with transition probabilities $P = D^{-1}W$.

To introduce the MARW we need to define a distance function $\mathbf{dist} : V \times V \to \mathbb{R}$ on the original set of vertices. When working on geometric graphs, that is, the vertices have an interpretation as points in some metric space, we use the distance in the underlying space as distance between vertices. Otherwise we use the shortest path distance.

Formally, a MARW with a agents and rope length l on graph G can be defined as a simple random walk on the state space S consisting of all unordered *valid* a-tuples of the n original vertices (possibly with repeated entries). We call an a-tuple valid if the distance between any two vertices is not greater than l. The probability of transitioning between valid a-tuples

$s = (s_1, ..., s_a) \in S$ and $t = (t_1, ..., t_a) \in S$ is defined as

$$M(s,t) := \sum_{\sigma \in S_a} \prod_{i=1}^{a} P(s_{\sigma(i)}, t_j) \mathbf{1}(\mathbf{dist}(s_{\sigma(i)}, t_j) > l).$$

Here, σ runs over the set S_a of all permutations of a elements. Note that the definition is fine for the purpose of defining the MARW, but the matrix M is impractical to work with: in the worst case it has $O(n^{2a})$ elements. Moreover, the permutations are hard to track.

To implement the MARW in practice there are several natural variants: the agents can walk one after the other or all at the same time, and they can be allowed to sit on the same vertex or not. Intuitively, in natural graphs these variants will behave similarly, even though they might behave slightly different in certain pathological examples. However, they have different complexities.

In this chapter we focus on the most natural construction: agents move at the same time and are allowed to sit on the same vertex. Assume we start with a agents in locations $v_1, ..., v_a$. A step of MARW consist of a independent simple random walks, one for each agent. Then we check if the new vertices still satisfy the condition on the rope length. If yes, we accept the step, otherwise we discard it. If the out-degree of each vertex is rather small, we would not discard too many steps in practice.

In applications with large degree vertices, we suggest to use the MARW variant where agents do not move together, but one after the other. Here we randomly select one agent at a time, suggest a new position for this agent by a standard random step and accept it if the rope length condition is satisfied. Note that much less positions are rejected in this version than in the version where all agents move at the same time. It can be seen that both variants of the MARW have similar properties, but the one where we move all agents at the same time is easier to analyze theoretically. This is the case we consider in the rest of this chapter.

If the vertices are not embedded in an ambient space, the shortest path distance is the natural distance that can be used for **dist**. Checking the condition of the rope length is tractable as we usually work with small rope lengths l and only need to compute the shortest paths between vertices in small neighborhoods.

The MARW has two parameters: the number a of agents and the rope length l. Much of the following analysis is devoted to understanding the influence and the interplay of these two parameters. We are going to characterize the behavior of the MARW by studying how easily it travels between different clusters. We provide several arguments that, for reasonably small a and l, the MARW behaves like a "power" of the original RW.

For example, we will see that given a neighborhood graph on a sample from some underlying density g, the MARW with a agents and reasonably small rope length l behaves like a random walk on a sample drawn according to g^a. As the peaks of the density g are amplified in g^a and the valleys get even deeper, the likelihood of transitioning between different clusters of the distribution becomes smaller for the MARW than for the original RW. This means that clusters are "more expressed" in the MARW than in RW.

Let us introduce one concept that will be used many times in the rest of the chapter: the MARW-graph. We have seen above that a MARW on a graph G can be described as a Markov chain on the product space of the vertices of G. To compare the properties of the standard random walk RW and the MARW, it is often convenient to consider the MARW as a simple

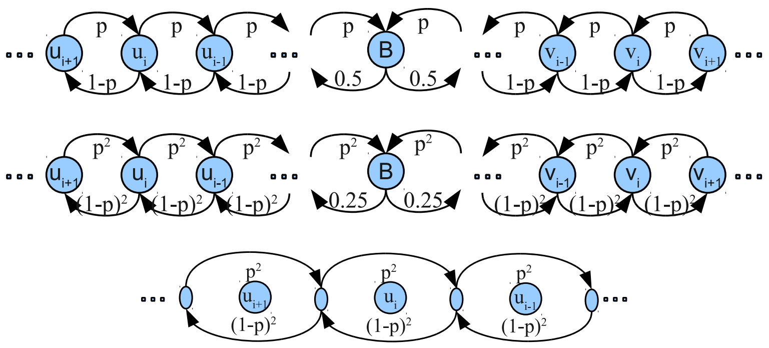


Figure 4.1: (a) A simple directed graph. (b) Corresponding MARW graph for two agent random walk with $l = 0$. (c) Corresponding MARW graph for two agent random walk with $l = 1$. Initially the two agents are placed in two neighbor nodes. Unnamed nodes correspond to center of masses and the self loops are omitted to keep the figures simple.

random walk on some graph G'. For example, it is easy to see that for rope length $l = 0$ and a agents the MARW on G coincides with a simple random walk on the graph G' that has the same vertices as G, but the edge weights w_{ij} are replaced by w_{ij}^a (see Section 4.3.1). More intricate constructions can be made for the case $l > 0$. In the rest of the chapter, whenever we are able to construct a graph G' such that the simple random walk on G' coincides with the MARW on G, we will call G' the *MARW-graph*.

4.3 How the number of agents affects the mixing time

In this section we investigate how the MARW's likelihood to jump outside a cluster behaves in dependency of the number of agents.

4.3.1 Intuitive example: a chain graph

We would like to start this section by considering an intuitive example of MARW on a directed and weighted chain graph as in Figure 4.1a. We choose $p < 0.5$. The graph consists of two clusters on the right and the left side of the chain, and a boundary vertex B in the middle. Inside each region, the transition probabilities are the same.

For simplicity let us start with **case $\mathbf{l = 0}$**. Here we have a agents that need to move simultaneously. The states of the MARW-graph G' coincide with the vertices of G. Assume all agents sit on vertex i. If $P(i, j)$ denotes the probability that one individual random walker goes along edge (i, j), the probability that a agents do it simultaneously is $P(i, j)^a$. Moreover, there is a considerable probability that the a agents do not agree in which direction they want to walk, namely $1 - \sum_j P(i, j)^a$. If we ignore these self-loops, then the loop-free MARW with rope-length 0 can be interpreted as a simple random walk on the graph G with transition probabilities $P(i, j)^a$. In particular, the MARW-graph is just the original graph, but with edge weights taken to the power of a (see Figure 4.1-b). Thus, whenever there is a choice of going in the direction of high or low edge weights, the MARW will choose the high weight edge even more often than the standard RW.

We now want to argue that in the case of small but positive l, the MARW behaves similar to the case $l = 0$. To see this, consider how the center of mass of the agents moves. If all agents move in the same direction, the center of mass does the same. Intuitively, when $l > 0$ we allow the agents to explore the vertices around the mass center. By increasing l, a bigger region will be allowed to get explored. We will now show that if the center of mass is not too close to the boundary (compared to l) and there is a “drift” away from the cluster boundary, then the MARW tends to follow this drift. Consequently, it tends to move away from the boundary towards the center of a cluster.

Consider the same graph as above **with l=1 and a=2**. Assume that both agents are in the right cluster. Each agent jumps with probability p to the left and probability $1 - p$ to the right. Consider the center of mass of the two agents. Depending on the movement of agents it will stay, go to the right or go to the left. If both of agents go to the right, which happens with probability $(1 - p)^2$, the center of mass will also move to the right. Similarly it moves to the left with probability p^2. It stays on the same vertex when the two agents want to move in different directions. They either change their place (with probability $p(1-p)$) or do not move because of violating the condition on l (with probability $(1 - p)p$). Thus, the probability that the mass center stays at the same place is $2p(1 - p)$. This implies that the center of mass of the two agents moves like a simple random walk on another graph G', namely the graph whose vertices correspond to the “mid point” between two consecutive vertices (Figure 4.1-c). In G' we have squared transition probabilities, except for the vertices near to the cluster boundary.

This example shows that, compared to the original random walk, the probability that a MARW walks in a region of low density is the square of the probability of the original random walk. **Increasing l** will increase the size of the boundary, but otherwise the argument will work as well. If we consider **a > 2 agents**, one can show in the same example that the ratio of the transition probabilities away from / towards the cluster boundary change from $\frac{1-p}{p}$ to $(\frac{1-p}{p})^{O(a)}$.

4.3.2 More rigorous: bounding the spectral gap

The main reason that we introduced the MARW is because we want to increase the probability of staying within a community. One way to treat this formally is to analyze the spectral gap of the graph, in particular the second eigenvalue λ_2 of the transition matrix. This eigenvalue is both a measure for how clustered the graph is (for example, it can be used to bound the Cheeger cut) and can also be used to bound the mixing time of the random walk. The smaller λ_2, the more clustered the graph is, and the larger the mixing time is. Thus, if λ_2 of the MARW decreases compared to the standard random walk, then we can conclude that the MARW takes longer to travel between different clusters.

In this section we treat the **special case where $l = 0$.** Our analysis will be mainly useful in weighted graphs. As we have already seen above, for $l = 0$ the MARW-graph G' has the same vertices as G but with edge weights raised to the power of a. We now want to compare the spectral gaps of G and G'. For a fair comparison we need to ensure that we consider the MARW without self-loops (otherwise the mixing time of the MARW increases trivially as the random walk often stays at a vertex without moving). As it is hard to argue about the spectral gap itself, we will compare lower bounds on the spectral gap.

These lower bounds will be provided by means of the Poincaré inequality (see Diaconis and Stroock (1991) for a general introduction to this technique). The outline of this technique is as follows: for each pair of vertices (X, Y) in the graph we need to select a "canonical path" γ_{XY} in the graph that connects these two vertices. Then we need to consider all edges in the graph and investigate how many of the paths γ_{XY} traverse this edge. We need to control the maximum of this "load" over all edges. Intuitively, a high load means that many paths need to traverse the same edges, that the graph has a bottleneck. Formally, the spectral gap β is related to the maximum average load b as follows (cf. Corollary 1 and 2 in Diaconis and Stroock (1991); we adapted their statements to the case of weighted graphs):

$$\beta \geq \frac{\mathrm{vol}(G) w_{min}}{d_{\max}^2 |\gamma_{\max}| b}, \tag{4.1}$$

where $d_i = \sum_j w_{ij}$ is the weighted degree, $\mathrm{vol}(G) = \sum_i d_i$ denotes the volume of the graph, w_{min} the minimal edge weight, d_{max} the maximal degree and b is the maximal load defined by

$$b := \max_{\{e \in E\}} |\{\gamma_{XY} \mid e \in \gamma_{XY}\}|.$$

Here $|\gamma_{\max}|$ is the maximal number of edges on the canonical paths. We will now compare the Poincaré bounds on the spectral gaps of G and G'.

Theorem 4.1 (Spectral gap of the MARW; $l = 0$) *Consider the lower bound on the spectral gap provided by the Poincaré technique. If this bound is u for the standard RW, then it is uC^{a-1} for the MARW with* **a** *agents and rope length $l = 0$. In general, C is a constant with $C \leq 1$ and for weighted graphs with non-uniform weights, it always satisfies $C < 1$.*

Proof. As explained above, the standard random walk and the MARW can both be seen as random walks on the same graph, just the edge weights are different. For this reason, we can use the same set of canonical paths in both graphs (and we do not need to specify it explicitly in our proof). In particular, the maximal load b and the maximal path length $|\gamma_{\max}|$ coincide in both graphs. What is different for both random walks are the minimal and maximal weights and degrees. Denoting the various quantities for G by normal letters and the corresponding ones of the MARW with a tilde on top, we obtain the following relations:

$$\begin{aligned}
\tilde{w}_{min} &= w_{min}^a. && (4.2)\\
\tilde{d}_{max} &\leq d_{max}^a. && (4.3)\\
vol(\tilde{G}) &= \sum_{e \in E} \tilde{w}_e = \sum_{e \in E} w_e^a \\
&\geq \frac{(\sum_{e \in E} w_e)^a}{|E|^{a-1}} = \frac{vol(G)^a}{|E|^{a-1}}. && (4.4)\\
\tilde{\beta} &\geq \frac{\mathrm{vol}(\tilde{G}) \tilde{w}_{min}}{\tilde{d}_{\max}^2 |\gamma_{\max}| b} \\
&\geq \frac{vol(G)^a w_{min}^a}{|E|^{a-1} d_{\max}^{2a} |\gamma_{\max}| b} \\
&= \beta (\frac{vol(G) w_{min}}{|E| d_{\max}^2})^{a-1}. && (4.5)
\end{aligned}$$

If we show that

$$C \quad := \quad \frac{vol(G)w_{min}}{|E|d_{\max}^2} \overset{!}{\leq} 1, \tag{4.6}$$

then the bound for spectral gap of MARW will exponentially decrease with a. Indeed,

$$vol(G)w_{min} = w_{min} \sum_e d_e \leq w_{min}|E|w_{max} \leq |E|d_{max}^2.$$

□

This proposition shows that for a MARW with a agents on a weighted graph, a lower bound on the spectral gap will decrease with "the a-th power" of defined C. In particular, as the gap of the standard random walk is always smaller than 1, this means that the bound on the spectral gap of the MARW is a power of a smaller than the gap in the original graph. Even though this does not prove that the spectral gap itself has the same behavior, it is a strong indication that this might be the case. In this case, we can conclude that the "bottlenecks" in the MARW are much harder to overcome than the ones in the original random walk.

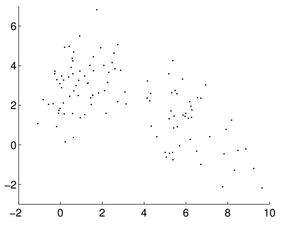
Note that if we consider unweighted graphs (that is $w_{ij} \in \{0, 1\}$)), then the special case of the loop-free MARW with $l = 0$ corresponds to the standard random walk. This just shows that in this case, the technique of bounding the spectral gap the way we do it is suboptimal and leads to the trivial result that both gaps are bounded by the same quantity.

In **case where $\mathbf{l > 0}$** but l is still small we expect a similar behavior as for the case $l = 0$: the mixing time will be much smaller than the one for the original random walk. Only when l becomes bigger, this behavior starts to change. On the other end, when l tends to ∞, the effect is turned around: the random walk mixes much faster than the original random walk, see Alon et al. (2008) and the next section.

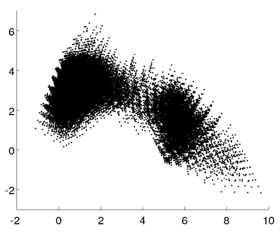
4.4 Analyzing MARWs on ϵ-graphs

In this section we analyze the case of random geometric graphs, in particular ε-graphs. These graphs are widely used in data mining and machine learning, and the effect of replacing the RW by the MARW can be visualized in a very intuitive manner. Assume we are given a sample of points drawn i.i.d. from some underlying probability distribution with density f on $\mathbb{R}^d$. The ε-graph takes these points as vertices and connects two points by an unweighted edge if the points have distance at most ε from each other. Intuitively, the ε-graph is supposed to show the structure of the underlying probability density f. As an example consider the density in Figure 4.2d. It consists of two clearly separated high density regions corresponding to two clusters. A sample of points from this density can be seen in Figure 4.2a and expresses the cluster structure as well. If we now build an ε-graph on this sample (with a reasonable choice of ε, say $\varepsilon = 1$), then this graph consists of two distinct clusters as well: each cluster has many connections inside the cluster and only few connections to the other cluster.

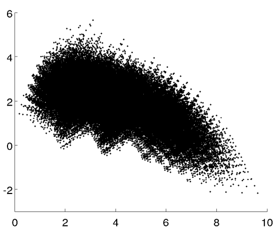
As we are going to explain now, the advantage of the geometric setting is that replacing a standard RW by a MARW can be interpreted as changing the cluster structure of the underlying density. We will see that for reasonably small l, replacing the standard RW by the MARW has a similar effect as replacing a random walk on n points drawn from the original density g by a random walk of $O(\tilde{n})$ points drawn from density g^a. Depending on the structure



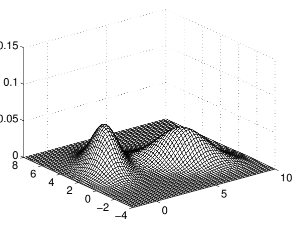
(a) Original sample: Moderate cluster structure.



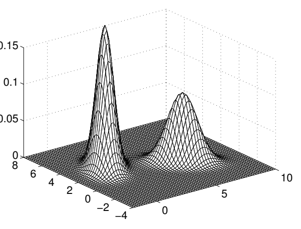
(b) Vertices of the MARW graph for small l: High cluster structure ($a = 3, l = 3$).



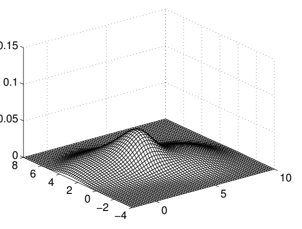
(c) Vertices of the MARW graph for large l: No cluster structure ($a = 3, l = 7$).



(d) Density f.



(e) Density f' for small l ($a = 3, l = 3$).



(f) Density f' for large l ($a = 3, l = 7$). The density smoothes heavily.

Figure 4.2: Density function and samples from a mixture of gaussians and the corresponding MARW's.

of graph, $\tilde{n}$ can vary from n to n^a. In particular, the peaks of g^a will be much sharper than the ones of g, leading to a much lower probability of transitioning between different modes of the underlying distribution.

For the ease of presentation we start with the **case $\mathbf{a = 2}$ and $\mathbf{l} = \varepsilon$**. Here we define the distance between two vertices in the metric of the underlying space (this corresponds to $l = 1$ if the distance between two vertices in the graph is measured by the shortest path distance). The general case will be treated later.

Let us describe the MARW-graph G' for the special case. G' will contain two types of vertices: vertices corresponding to the situation that the two agents sit on the same vertex of G, and vertices corresponding to the situation that the two agents sit at the two ends of an edge of G. We denote the set of first type vertices by $U_1 = \{u_{x,x} \mid x \in V\}$ and the second type of vertices by $U_2 = \{u_{x,y} \mid e_{x,y} \in E\}$. Two vertices u_{x_1,y_1} and u_{x_2,y_2} are connected by a directed edge in G' whenever it is possible to move the two agents from $\{x_1, y_1\}$ to $\{x_2, y_2\}$ in one step in the graph G. The weight of an edge in G' will be defined as the probability that the MARW performs the corresponding step on G. Note that we define G' without self-loops. As the original graph G is unweighted, it is easy to see that the weights of all outgoing edges of a vertex $u_{x,y}$ are identical. We can represent G' as a geometric graph by identifying each vertex $u_{x,y}$ in G' with a point in $\mathbb{R}^d$, namely with the center of mass of x and y. For an example consider Figure 4.2. We sampled 110 random points from a mixture of two Gaussians with different covariances in 2 dimensions and built an unweighted kNN-graph with $k = 15$ on this sample. The sample itself is shown in Figure 4.2a. Now consider a MARW with $a = 3$ and $l = 1$ on G (and shortest path as distance function). We plotted the vertices of the corresponding MARW-graph G' in Figure 4.2b (the vertex $u_{x,y,z}$ is represented by the center of mass of x, y, z).

We now compare the geometric properties of the two graphs G and G'. Consider the ball B of radius $\varepsilon/2$ centered at a particular point $x \in G$. Denote the number of sample points in B by k. By definition, all sample points in B have distance at most ε to each other and thus are connected in G. The graph G' will have k type-1 vertices corresponding to the points in B, and $k(k-1)/2$ type-2 vertices corresponding to pairs of points in B. This shows that the number of vertices in B increases from $O(k)$ in G to $O(k^2)$ in G'.

Now recall that our sample points have been sampled from an underlying density f. Assume that ε is small enough such that the density on an ε-ball can be considered approximately constant. Consider two ε-balls B_1 and B_2 centered x_1, x_2. By the definition of a density, the numbers of sample points in B_1 and B_2 are roughly related by

$$\frac{\#\text{ vertices of } G \text{ in } B_1}{\#\text{ vertices of } G \text{ in } B_2} \approx \frac{f(x_1)}{f(x_2)}.$$

What would be a density f' corresponding to the vertices in G'? As we have seen above, when k is the number of vertices of G inside B, then the number of vertices of G' inside B is of the order k^2. Hence we deduce

$$\frac{f'(x_1)}{f'(x_2)} \approx \frac{\#\text{ vertices of } G' \text{ in } B_1}{\#\text{ vertices of } G' \text{ in } B_2} \approx \frac{f(x_1)^2}{f(x_2)^2}.$$

Consequently, G' can be interpreted as a geometric graph based on the underlying density $f' = f^2$. Squaring a density leads to much sharper peaks in the modes of the distribution and deeper valleys between the modes, that is the cluster structure is much better expressed. For an illustration, compare the densities in Figures 4.2d (original density) and 4.2e (density corresponding to a MARW with small rope length). It is clear that a random walk on a sample of f' (corresponding to the MARW on G) has a much harder time than the original RW on a sample of f (corresponding to the RW on G) to travel from one cluster to the other one. Additionally, the number of vertices in G' corresponds to $|V| + |E|$, that is, it is of the order $O(|E|)$ rather than of the order $O(|V|)$ as in the original graph G. This effect can be seen in Figures 4.2a (vertices of G) and 4.2b (vertices of G').

In **case of general $a > 1$ and rope length $l = \varepsilon$** a similar argument shows that the distribution of vertices in G' is raised to the power of a: denoting the number of cliques of size t by V_t, the number of vertices in G' is $\sum_{i=1}^{a} \binom{a-1}{i-1} |V_i|$. To see this, note that the number of placements for a agents on i vertices with at least one agent at each vertex is $\binom{a-1}{i-1}$. Depending on the exact numbers $|V_i|$, the number of vertices in G' can vary between $O(n)$ to $O(n^a)$.

The above argument still holds **if the rope length l is larger than ε**. In this case, however, we can no longer argue about the densities f and f' themselves, but rather about their smoothed counterparts obtained by convoluting f and f' with the function that is constant on an l-ball. Denote the characteristic function of a ball with radius l and centered at x_0 by $\chi_{B(x_0,l)}$:

$$\chi_{B(x_0,l)}(x) = \begin{cases} 0 & \text{if } d(x, x_0) > l \\ 1 & \text{otherwise.} \end{cases}$$

Then the density of G' satisfies

$$\frac{f'(x_1)}{f'(x_2)} \approx \frac{(\int f(t)\chi_{B(x_1-t,l)}dt)^a}{(\int f(t)\chi_{B(x_2-t,l)}dt)^a}.$$

That means G' can be interpreted as a geometric graph based on the underlying distribution

$$f'(x) \propto (f(x) * \chi_{B(x,l)})^a, \tag{4.7}$$

where $*$ is the convolution operator. For small l these convolutions will be reasonably close to the original density functions, and lead to the same intuition as above. However, as soon as l gets very large (for example, l is as large as the distance between different modes of the underlying distribution), then the interpretation changes completely. In this case, the density f' is smoothed so heavily that the different modes are not visible any more and the distribution becomes unimodal in the extreme case. Then, the random walk on G' mixes very fast as there are no bottlenecks any more, and transitions easily between points corresponding to different clusters of the original density f. In this case, we completely changed the behavior of the MARW: instead of mixing slower than the original RW it now mixes much faster (and this is the case considered in Alon et al. (2008)).

This effect is demonstrated in Figure 4.2. Plot 4.2f shows the smoothed density f'. As we can see, the clusters have vanished completely and are replaced by a unimodal distribution. The same effect can be seen for the points of the MARW-graph G' in Figure 4.2c.

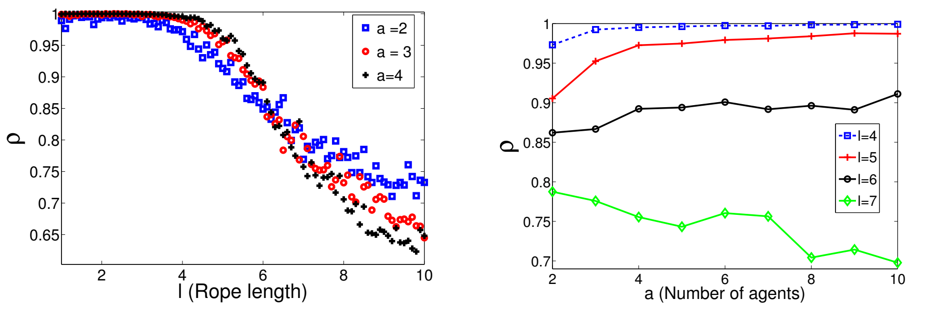
These insights can be used to understand **the choice of l**. There is a trade-off between two goals. The rope length has to be considerably smaller than the diameter of the cluster and the distance between different clusters we want to find (otherwise the smoothed distributions loose their cluster properties as in Figure 4.2f). On the other hand, it should be as large as possible in order to allow G' to have many vertices (otherwise G' will look almost as G, and the effect of sharpening the peaks by squaring the distribution will not occur).

While the above argument was for ε-graphs, one can observe similar effect on other types of random geometric graphs such as kNN graphs or Gaussian neighborhood graphs. By a similar analysis one can also discuss the behavior of a **MARW on a general, unweighted non-metric graph**. Even though we do not have a geometric interpretation in terms of an underlying distribution, the effect of the rope length and the trade-off between the different aims when choosing a good rope length is the same as described above.

4.5 Experiments

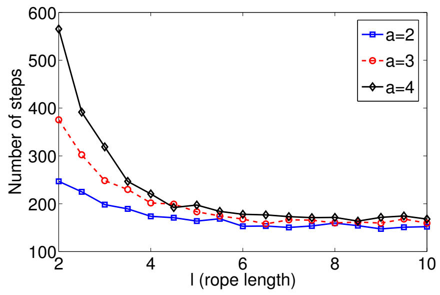
Here we demonstrate different aspects of the MARW in simulations and experiments and compare it to other algorithms from the literature. By using expensive partitioning algorithms, the authors of Leskovec et al. (2008) found that even in massive graphs the typical clusters have a relatively small size. This shows that in real world datasets one should look for clusters in the range of hundreds to thousands of nodes. To validate local clustering algorithms it is thus enough to study graphs up to a couple of thousand vertices, which is what we do in our experiments.

Whenever we know the true cluster labels in a data set, we measure the performance of a local clustering algorithm by the following procedure. We call the starting vertex x_s, the true label



(a) Effect of rope length l on the performance measure ρ

(b) Effect of number of agents a on the performance ρ



(c) Influence of rope length on the speed of MARW

Figure 4.3: Influence of rope length l and number a of agents on the MARW

of vertex x by $y(x)$ and the set of visited vertices by S. Then we define r as the ratio

$$r = \frac{|\{x \mid y(x) = y(x_s)\}|}{|S|},$$

which is the fraction of the visited points belonging to the same cluster as x_s.

4.5.1 Effect of rope length and number of agents

We sample 200 points from a mixture of Gaussians (similar to Figure 4.2a) in $\mathbb{R}^2$ and build the kNN graph for $k = 10$. We run the MARW with $a = [2, 3, 4]$ for rope lengths in the interval $[1, 10]$. We allow the MARW to take 200 steps and record the set of visited points.

In our first experiment we fix the number of agents and investigate the effect of the rope length. We say that a MARW is accepted if $r \geq 0.9$. We repeat this procedure 100 times with random starting vertices. The performance measure ρ is defined by:

$$\rho = \frac{\text{Number of accepted MARW's}}{\text{Number of MARW's}}.$$

The results are shown in the Figure 4.3a. We can observe very good performance of MARW for $l \approx [1, 4]$. In this regime, nearly all MARWs get accepted. The performance drops by increasing l. Note that in this data set, the width of the bottleneck region is about 3 (cf.

Table 4.1: Percentage of correctly visited vertices for different real world datasets. N is the number of vertices, E is the number of edges, D is the dimensionality of data and K is the number of clusters. The number in parentheses shows the cluster size we used as stopping criterion

Dataset	N	E	D	K	Nibble	Apr.PR	MARW ($a = 3$)	MARW ($a = 4$)
Wine (40)	178	1268	13	2	82.31	86.8	88.02	91.76
B.Cancer(160)	683	10022	9	2	93.26	94.66	94.37	96.02
USPS (50)	7291	106722	256	10	88.5	92.78	93.37	97.43
USPS (100)	7291	106722	256	10	81.48	88.15	90.85	92.21

Figure 4.2a). This corroborates our rules of thumb stated in Section 4.4: as soon as l gets considerably wider than the bottleneck, the random walk mixes too fast, hence many MARWs get rejected in our experiment.

As a baseline we also used a simple RW on the same data. It leads to performance $\rho = 55\%$., which is way worse than the MARW results. Additionally we observed that the variance of the value ρ (measured over different starting points) is much higher for RW than for the MARW. This shows that MARWs not only give better performance, but also lead to a more stable behavior.

Next we investigate the effect of the number of agents on the same graph by running the MARW for $a \in [2, 10]$ and $l \in \{4, 5, 6, 7\}$. The results are shown in Figure 4.3b. When l is still small, by increasing a we observe an increase of the performance ρ. That is, in this regime more agents lead to more accepted MARWs. As soon as the rope length gets too long ($l = 7$) the performance decreases with the number of agents. This is the regime where the MARW mixes faster than the orignial RW. All in all, this experiment shows that the rope length l governs whether the cluster structure of the MARW is more or less expressed than in the original density. The number of agents acts as an amplifier.

Note that while increasing the number of agents amplifies the performance of the MARW, it also has a negative side effect. The larger a, the slower the MARW moves. This has two effects. First, when we implement the MARW, much more proposal steps get rejected for violating the rope length condition (leading to an increase in running time). Second, the tendency of staying in a high density region might be so pronounced in a MARW with large a that it barely moves around but often visits vertices it has already seen. This effect is demonstrated in Figure 4.3c. We run MARW with different number of agents and rope lengths until they visit 80 different nodes. The number of steps during the visit shows that by increasing the number of agents, they tend to stay in local neighborhoods. This effect vanishes when we increase the rope length. When the rope length gets long enough (in this experiment, around 6), agents move with more freedom and less tendency to stay in the same place.

4.5.2 Real world examples

We now compare performance of various algorithms on real world datasets: our MARW, the *Nibble* algorithm from Spielman and Teng (2008) and approximate PageRank (*Apr.PR*) from Andersen et al. (2006). Note that all algorithms have different stopping criteria. In order to make them comparable, we simply run all algorithms from the same starting point until they

Table 4.2: We fixed the number of *different* vertices we wanted to visit (reported in parentheses). The numbers in the table show how many *steps* the MARW needed to make in order to visit this number of different vertices, on average over different starting points. The rope length l was fixed to $l = 1$.

# of agents	Wine (40)	Breast Cancer (100)	USPS (50)	USPS (100)
$a = 3$	487	2417	156	361
$a = 4$	1142	19798	281	752

discovered a given number of vertices. Then we analyze the quality of the clusters that are discovered by the algorithms. We set the parameters in the algorithms as follows. In *Nibble* we optimize the threshold parameter by approximately finding smallest threshold that allows us to discover the mentioned number of vertices. For approximate PageRank, using the value of α suggested by the authors led to poor results. So instead of using their values we report the best result obtained for a range of α.

We use the *Wine* and *Breast Cancer* datasets from UCI repository, the USPS handwritten digit recognition dataset (Hull, 1994) and a collaboration network of arxiv papers (Leskovec et al., 2007).

For the first three data sets the true cluster labels are known. For these data sets we first constructed an unweighted kNN graph ($k = 10$). We started the algorithm from a random starting point. We ran each algorithm until visiting a given number of points: approximately $\frac{2}{3} \times$ *size of smallest cluster* vertices in Wine and Breast Cancer datasets, and 50 resp. 100 points in the USPS data set. Then we used r, the percentage of visited points from the correct cluster, as our performance measure. We repeated the experiments for 100 random starting points. For our algorithm we simply set the rope length to $l = 1$ (we did not optimize over l or a in any way). For the other algorithms we chose the parameters as described above. The results for $a = 3, 4$ are shown in Table 4.1. We can see that our algorithm achieves the best results on all data sets.

It worth to note that even though using more agents improves the performance, it also increases the running time of the algorithm. There is a trade off between performance versus running time and visiting majority of points in the cluster. MARW with big a will move slower and will mainly remain in the center of a cluster. This effect is demonstrated in Table 4.2. In this table we fix the number of *different* vertices we want to visit. We then report how many steps the random walk had to perform to achieve this (a step is simply one "hop" of the random walk). For example, in the Wine data set we can see that in order to visit 40 different vertices, the MARW with $a = 3$ had to take about 500 steps while the MARW with $a = 4$ already needed about 1100 steps. This shows that for larger a, the random walk tends to visit the same vertices more often and is more reluctant to move away from high-density regions.

The *Collaboration Network* dataset (*Collab* for short) consists of an undirected non-metric graph with about 12000 vertices. As ground truth is not known on this data set, we use the conductance of the set of visited vertices as a performance measure. The conductance of a cluster is defined as the ratio of the number of its external connections to the number of its total connections. Formally, the conductance of a subgraph S from graph $G = \{V, E\}$ is defined as:

$$\Phi(S) = \frac{\sum_{i \in S, j \in V-S} w_{ij}}{\min(\mathrm{vol}(S), \mathrm{vol}(V - S))},$$

Table 4.3: Conductance of the set of visited points in the collaboration dataset. N is the number of vertices and E is the number of edges.

Dataset	N	E	Nibble	Apr.PR	MARW $a = 3$
Collab. (200), $l = 1$	12008	237010	0.5068	0.4916	0.4342
Collab. (500), $l = 2$	12008	237010	0.4703	0.4565	0.194

where $\text{vol}(A) = \sum_{i \in A, j \in V} w_{ij}$. The conductance is a popular quantity to measure the quality of a cluster in a graph. The aim is to visit a set with small conductance.

We run two rounds of experiments, one aiming for 200 and one for 500 vertices. For the MARW we use the shortest path distance as underlying distance function and rope lengths $l = 1$ (for the 200 vertices) and $l = 2$ (for the 500 vertices). We set the number of agents to 3. The results are shown in Table 4.3. In this example by visiting 500 vertices, MARW works significantly better than other algorithms and the average conductance is less than half of conductance reached in other algorithms.

4.6 Discussion

In this chapter we introduced the new concept of the multi-agent random walk. We provided several ways to analyze the behavior of the MARW. The bottom line of these analyses is always the same: using the MARW instead of the standard RW enhances the cluster properties of the graph. Hence, the MARW is well-suited for local clustering. Our experiments show that it outperforms the major other approaches from the literature.

The concept of a MARW is new, and we believe that it has many advantages and can be applied to other data mining problems. For example, one might be able to derive a spectral clustering algorithm based on the MARW-Laplacian instead of the standard random walk Laplacian, and similar ideas might work for semi-supervised learning. Essentially, any algorithm that works with random walks or related matrices might be adapted to the MARW. It will be a matter of future work to fully explore the possibilities of multi-agent random walks in data mining and machine learning.

Chapter 5

Friend recommendation with local distances

5.1 Introduction

Fundamental to all Online Social Networks (OSN's) is the goal to effectively predict and recommend friendships between users. OSN's facilitate link formation and friendships among users and thus increase the value of the site for it's members. Users will subsequently spend more time on the network and thus increase the site's traffic and the potential for monetization. Moreover more users will recognize the value of the network and will join the site. Typically friend recommendation systems in OSN's are responsible for a large fraction of the created edges in the social graph.

One of the main ingredients of the success of OSN's is the ease with which friendship groups and communities arise. These groups often arise among like-minded users, i.e. users that share the same interests. Taping into the principal of "homophily" (McPherson et al., 2001) these networks provide a rich source of user behavior and preferences (Yang et al., 2011). By exploiting this principal the network itself can be used as a recommendation engine by essentially recommending items to users that their friends liked. These recommendations further increase the value and popularity of the network to both user and vendors.

Related work Friend recommendation can be seen as a type of link prediction problem in a network (Liben-Nowell and Kleinberg, 2003). Although a general link prediction algorithm can be used as a friend recommender, we can implicitly model user behavior in order to build a customized algorithm for the specific task of friend recommendation in social networks. Most friend recommendation algorithms are based on evaluating a similarity score between vertices in the social graph (Backstrom and Leskovec, 2011, Sarkar et al., 2012). To suggest a new friend for user u, potential friends of u are ranked with respect to the similarity scores. Users that end up at the top of the ranking list are suggested to the user.

The similarity scores can be categorized based on the amount of data around u that they use. Lü and Zhou (2011) consider three classes of local, global and quasi-local scores.

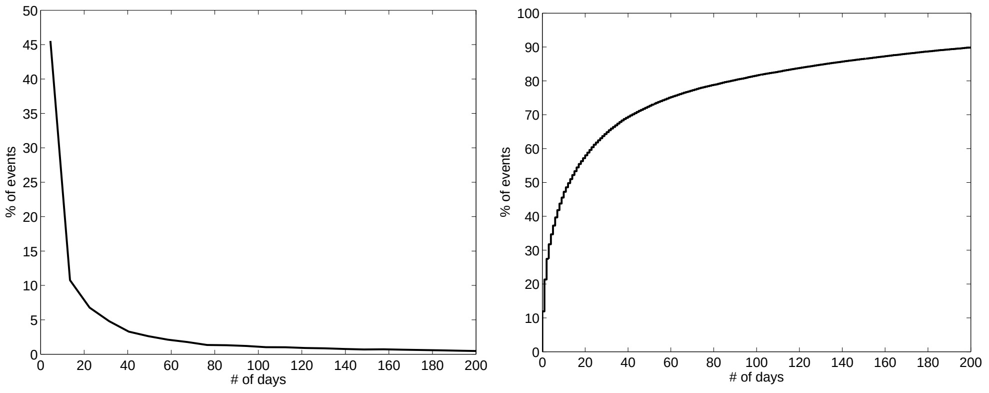


Figure 5.1: Temporal effects of link creation in a real OSN. Assuming a link is created between user u and user v: (a) the distribution of time difference between this event and a subsequent link creation betwen user u and one of users v friends. (b) the cumulative distribution. Figures by Dionysios Logothetis.

Popular local methods include *Common Neighbors* (CN) where potential friends are ranked in descending order of the number of common neighbors (Liben-Nowell and Kleinberg, 2003). The *Resource Allocation* (RA) Index, score is based on common neighbors weighted by the inverse of their degrees (Zhou et al., 2009). The *Adamic-Adar* (AA) Index is similar to RA, but uses the logarithm of the degrees to decrease the effect of common friends with many friendships:

$$S^{AA}(u,v) = \sum_{x \sim u,v} \frac{1}{log(d_x)},$$

where d_x denotes the degree of x (Adamic and Adar, 2003).

Quasi-local scores like the Local Path (LP) Index (Lü et al., 2009) use a wider area of the graph than the second layer neighborhood. Methods modeling the temporal characteristics of social neighborhoods (Sarkar et al., 2012) can also be seen as quasi-local.

The Katz (Katz, 1953) measure is a score that exploits the global structure of the graph. Although the Katz index performs well in small-scale datasets (Lü and Zhou, 2011), there exist theoretical doubts on its applicability on massive graphs (von Luxburg et al., 2010). Factor models such as Backstrom and Leskovec (2011), Miller et al. (2010), Menon and Elkan (2011) fall also into the global scores category and have been used on different types of networks.

However in real networks with tens of millions of nodes and billions of edges, global and even quasi-local scores are infeasible to compute. Local scores are thus the only remaining category that can be easily used in a real world use scenario.

Temporal dynamics in a real OSN To motivate the use of temporal dynamics in the distance measure, we analyze the temporal characteristics of link additions on a graph from a real OSN. The social graph used is from the *Tuenti* OSN Spain's leading social network described in more detail in Section 5.3. We show that once a new friendship is established, a user is likely to connect with another user through this new friend within a short amount of time. More specifically, we collect all new link additions during an entire day which amount to approximately 380K of new links. For every new link we discover all previous common friends of the linked nodes. Among those, we calculate the time difference between the most recent link addition and the new addition. Figure 5.1b shows the cumulative distribution of the time difference across all new link additions. We observe that 50% of the new link additions occur within less than 12 days from a previous link addition. In Figure 5.1a , we show the normalized distribution of the time difference in link creation in days and notice that the percentage drops exponentially.

The method proposed in this work is motivated by the above observations and also falls into the category of local methods. In this work:

- We propose a novel local approximation of a diffusion kernel which takes the time and the direction of link creation into account.
- We show in Section 5.3 that exploiting the temporal information in the link creation process boosts performance in terms of IR metrics by a significant amount compared to standard link prediction methods with only minimal computational overhead.
- The proposed method scales with ease to social networks with several millions of users.

5.1.1 Notation

We represent the social network by a graph G which consists of the vertex set V with $|V| = n$ and the set of directed edges E. The edge creation function is $T : V \times V \to \mathbb{R}^+$ where $T(u, v)$ represents the time u and v acquire their friendship in the network. If they never get connected, we set $T(u, v) = \infty$. We define $\mathcal{N}_t(u)$, the first layer neighborhood of a vertex u at time t as the set of vertices directly connected to u at time t, i.e. his friends. Similarly $\mathscr{N}_t(u)$, the second layer neighborhood of u is defined as the set of vertices with distance two from u, i.e. his friends of friends. We call each member of set $\mathscr{N}_t(u)$ a *candidate friend*.

The information of edge creation time up to time t is provided. The friend recommendation task boils down to predicting the next vertex that is going to get connected to u. Standard approaches dealing with friend recommendation are based on the assignment of similarity scores $s(x, y)$ between vertices x and y. This score is not necessarily a symmetric score. Temporal information is usually ignored or not considered in these scores. Scores are typically computed for the vertices in $\mathscr{N}(x)$ and the nodes in the second layer neighborhood are ranked based on their scores. Typically the top-N scored candidate friends (where N is usually between 3-10) are recommended. We evaluate the proposed algorithm on data from a real world social network using ranking metrics routinely used in Information Retrieval.

5.2 Heat diffusion in a network

In this chapter, we propose to use a *time kernel* (a decreasing function of time) to model the appearance of a link. The intuition behind this time kernel is the following: When a user u acquires a friendship with user v at time t, their friendship gets hot. This means that it would be more likely that u also acquires friendship with friends of v in the “near” future. As time passes, the temperature of this friendship decreases and the friendship circle between u and v

gets saturated. Then their friendship colds down and u receives fewer recommendations from friends of v.

This procedure is modeled with a decreasing function of time called a time kernel. Here we derive a function modeling the heat diffusion in the network.

Motivation The physical model of heat diffusion in an object has inspired a family of algorithms in graph analysis and machine learning (Perona and Malik, 1990). In physics, the diffusion of heat is described by a partial differential equation subject to initial and boundary conditions (Lafferty and Lebanon, 2005). We sketch the concept with an example.

Consider a metal wire with length ℓ, aligned along the x-axis. Assume that at time $t = 0$, the wire at point x has the temperature $u(x)$ (initial condition). The solution of the heat equation $\varphi_t(x)$ describes the temperature of point x at time t. As time passes and when there are no heat sources and sinks (no boundary condition), the temperature stabilizes and we get the same temperature everywhere as $t \to \infty$.

Related work The heat equation has been used in Perona and Malik (1990) for image analysis and denoising while Kondor and Lafferty (2002) presented diffusion kernels on graphs and other discrete structures which can be seen as a discretization of the familiar Gaussian kernel. Laplacian regularization for semi-supervised learning (Zhu et al., 2003) is also equivalent to solving a heat equation subject to initial conditions imposed by labels. Moreover, recent works has been devoted to the diffusion of information or disease in a network e.g. Gomez-Rodriguez et al. (2011), although diffusion as defined there has different dynamics compared to the heat diffusion. In this work we use the heat diffusion process to model the creation of the network itself.

We first define the diffusion distance and run through existing results. Then we define the random walk diffusion distance, which is a modified version of the heat kernel defined in Kondor and Lafferty (2002). The natural involvement of time in this process is a motivation point for us to study the model. Inspired by these results, we propose a local approximation of the random walk diffusion distance with its application for friend recommendation in our mind.

5.2.1 Heat diffusion

The social graph G consists of the vertex set V and the set of directed edges E. We are given a function T defined over edges where $T(u, v)$ represents the time this edge appears in the network. Denote the degree of vertex i by d_i. The (unnormalized) graph Laplacian matrix $\mathcal{L}_\mathrm{u}$ is defined as

$$\mathcal{L}_\mathrm{u}(i,j) = \begin{cases} -1 & \text{for } i \sim j \\ d_i & \text{for } i = j \\ 0 & \text{otherwise.} \end{cases}$$

The matrix $-\mathcal{L}_\mathrm{u}$ is a discrete counterpart of the weighted Laplace-Beltrami operator (Hein et al., 2007).

The diffusion of heat through a continuous material during the time is described by the heat equation

$$\frac{\partial}{\partial t}u(x,t) = k\Delta u(x,t), \tag{5.1}$$

where k is the thermal diffusivity and Δ is the Laplace-Beltrami operator

$$\Delta u(x,t) = \nabla^2 u(x,t).$$

The initial state of the system at time $t = 0$ is specified by initial conditions in the form $u(x,0) = f(x)$. A heat kernel or a fundamental solution is the solution of Equation 5.1 with initial condition of a point heat source at point x_0:

$$\begin{cases} \frac{\partial}{\partial t}u(x,t) = k\Delta u(x,t) & 0 < t < \infty \\ u(x,0) = \delta(x - x_0) & , \end{cases} \tag{5.2}$$

where δ is the Dirac delta function.

Denote the fundamental solution by $u^*_{x_0}$. With fixing a time t, we can define a distance function between points using this fundamental solution $U_t(x,y) = u^*_y(x,t)$. This distance can be interpreted as the temperature of point x at time t when we have a point source of heat at y.

The heat equation can be translated to the discrete domain with the geometry induced by the graph structure in the following form:

$$\frac{d}{dt}K(x,\cdot) = -k\Delta K(x,\cdot). \tag{5.3}$$

Assuming that the thermal diffusivity is one ($k = 1$) and the convergence of the discrete unnormalized Laplacian to the continuous Laplacian $\mathcal{L}_{\mathrm{u}}K(x,\cdot) = \Delta K(x,\cdot)$ (see Hein et al. (2007), Section 3.3 for conditions under which this assumption holds), Kondor and Lafferty (2002) ended up with the graph diffusion kernel:

$$K_t = e^{-t\mathcal{L}_{\mathrm{u}}}. \tag{5.4}$$

It is easy to check that this kernel is a solution of Equation 5.3. We can interpret $K_t(u,v)$ as the temperature of vertex v at time t when we initialize the networks temperature with 1 at vertex u and 0 elsewhere. The long term behavior of K can be seen by allowing $t \to \infty$, which gives $K(u,v) \to 1/n$, where n is the number of nodes in our graph. This shows that eventually the network will stabilize and attain the same temperature everywhere when there is no energy loss.

5.2.2 Time-aware friend recommendation

The graph heat diffusion kernel K_t can be calculated by diagonalizing the generator matrix

$$\mathcal{L}_{\mathrm{u}} = SAS^T,$$

where S is a unitary (rotation) matrix and A is a diagonal matrix with elements a_i (eigenvalues) on its diagonal. Then we get

$$K_t = Se^{-tA}S^T,$$

where e^{-tA} is the diagonal matrix with elements e^{-ta_i} on its diagonal. This computation is expensive even for sparse graphs.

Although the utility of this kernel has been shown in many applications (Belkin et al., 2006, Zhu, 2005), the computational complexity prevents us from applying the heat kernel on graphs with more than thousands of nodes. Online social networks tend to have millions of nodes, while the underlying social graph dynamically changes over the time. Moreover, in an online recommendation system we need to query the distance between specific vertices in the current graph several times in a second in order to provide recommendations to the users of the OSN. Our goal is to find a fast approximation for this kernel.

The unnormalized graph Laplacian approximates the Laplace-Beltrami operator only for uniform measures. Hein et al. (2007) show that the random walk Laplacian $\mathcal{L}$ defined as

$$\mathcal{L}(i,j) = \begin{cases} -\frac{1}{d_i} & \text{for } i \sim j \\ 1 & \text{for } i = j \\ 0 & \text{otherwise,} \end{cases}$$

converges to the Laplace-Beltrami operator, even when our underlying space is equipped with a non-uniform probability measure. On the other hand, social networks are highly non-uniform in degree distribution (Barabasi and Albert, 1999). This encourages us to use the random walk graph Laplacian $\mathcal{L}$ instead of the unnormalized Laplacian $\mathcal{L}_{\mathrm{u}}$. We call the resulting kernel $K_t^{\mathrm{rw}} = \exp(-tk\mathcal{L})$ a *random walk diffusion kernel.*

In this work we find a linear time local approximation of $K_t^{\mathrm{rw}}(x,y)$ which makes possible to apply it for friend recommendation in massive dynamic graphs. Fixing a vertex x, the local approximation is defined for nodes in the first and the second layer neighborhood of x.

The diffusion distance is a global measure of distance in the sense that $K(x,y)$ takes all different paths between x and y into account, and weights them exponentially decreasing with their length. To see this, we can write the tailor expansion of matrix exponentiation:

$$e^{-tL} = I + \sum_{i=1}^{\infty} \frac{(-tL)^i}{i!},$$

and observe that $(\mathcal{L})^i$ is related to paths with length i.

This suggests that when x and y are spatially "near" to each other (e.g. they are neighbors), long paths will not play a big rule in the actual diffusion distance between x and y. The fundamental solution of Equation 5.2 can be written in a closed from. To find a plausible approximation for the random walk diffusion kernel in local neighborhoods, we benefit from this solution which is:

$$\Phi(x,t) = \frac{1}{\sqrt{4\pi kt}} \exp\left(-\frac{(x-x_0)^2}{4kt}\right).$$

We can transfer this result to discrete domains by considering the fundamental solution of Equation 5.3 on infinite grids:

$$\varphi_t(x,y) = \frac{1}{\sqrt{4\pi kt}} \exp\left(-\frac{(x-y)^T(x-y)}{4kt}\right). \tag{5.5}$$

This motivate us to use the following approximate distance for two neighboring vertices x and y:

$$\overline{\varphi}_t(x,y) = \frac{1}{\sqrt{4\pi k(t - T(x,y))}} \exp\left(-\frac{1}{4k(t - T(x,y))}\right). \tag{5.6}$$

where $T(x,y)$ is the time in which x and y become connected in the network (i.e. become friends).

To approximate the temperature at vertex y in the second layer neighborhood of x, we use a lazy random walk interpretation of diffusion (see also Section 3.3 in Kondor and Lafferty, 2002). A lazy random walk on graph G with parameter β will jump from vertex i to vertex j with probability β/d_i and will stay in i with probability $1-\beta$. It is easy to show that this random walk is related to the random walk diffusion kernel K_t^{rw}. This leads us to the following approximation for any $y \in \mathbb{N}_t(x)$:

$$\overline{\varphi}_t(x,y) = \sum_{v \in \mathcal{N}(x) \cap \mathcal{N}(y)} \frac{\beta \overline{\varphi}_t(x,v)}{d_v}. \tag{5.7}$$

The parameter β appears in all distances between candidate friends, so we can ignore it in computing scores.

Note that the contribution of old friends gets toward zero in Equation 5.7, which is an undesirable property. This is caused by the assumption that the underlying graph is an infinite grid. In infinite grids, the temperature will go towards zero when there is no permanent heat source.

We remedy this effect by adding an extra assumption that all vertices are connected to a heat source with temperature 1. Using the notation $\Delta_{x,y}t = t - T(e_{x,y})$, our random walk diffusion score between x and y takes the form:

$$S_t(x,y;k) = \sum_{v \in \mathcal{N}(x) \cap \mathcal{N}(y)} \frac{1 + e^{-\frac{1}{k\Delta_{x,v}t}} / \sqrt{k\Delta_{x,v}t}}{d_v}. \tag{5.8}$$

5.2.3 Learning the thermal diffusivity

The diffusivity parameter k balances the preference between candidate friends with few common recent friends and ones with many common old friends. Here, we describe a simple but effective procedure to learn the thermal diffusivity parameter k in an online-learning scenario.

Friend recommendation can be formulated as the following online learning problem: We start with a graph G_0, and assume that at time t_i node x_i acquires a new friendship. We predict (recommend) new friendships using a ranking function r_{x_i} which produces a score for each candidate friend x_i according to which the candidate friends are ranked. We then observe y_i and the algorithm incurs the reciprocal rank loss:

$$l(y_i, r_{x_i}) = 1 - \frac{1}{r_{x_i}[y_i]}.$$

where $r_{x_i}[y_i]$ is the position of the observed friendship y_i given the ranking induced by the scores r_{x_i}. Then we update our graph G_0 and add this new edge. Note that the ranking induced by the function r_{x_i} would depend on time t_i, but we drop the time index when it is clear from the context.

Lets first consider the separable case where we can choose the parameter k such that the reciprocal rank loss is zero for all i. Define $V_{x_i,t_i} := V - \{\mathcal{N}_{t_i}(x_i) \cup y_i\}$, the set of all potential friends of x_i at time t_i. The zero loss implies

$$\forall i : S_{t_i}(x_i, y_i; k) > \max_{y \in V_{x_i,t_i}} S_{t_i}(x_i, y; k).$$

We set the parameter k to maximize the minimal difference between the score of the next friend y_i and the closest runner up. This is in analogy to the max margin principle in Vapnik (1998).

For soft margin maximization in the batch case, we allow constraints to be violated by minimizing:

$$J = \sum_{i=1}^{n} \left(S_{t_i}(x_i, y_i; k) - \max_{y \in V_{x_i,t_i}} S_{t_i}(x_i, y; k) \right)^2. \tag{5.9}$$

In the online-learning scenario, we can replace k with k_{t_i}. To be able to write the solution in a closed form, we do another step of approximation for neighboring vertices:

$$\hat{\varphi}_t(x, y) = \frac{1}{\sqrt{k_t \Delta_{x,y} t}} \exp\left(-\frac{1}{k_{t-1} \Delta_{x,y} t} \right).$$

In another word, we fix the k in the exponent to its previous value to get a quadratic minimization problem. Also for the sake of notation simplicity, we denote the closest runner up by $\check{y}_i$. After taking derivatives and setting them to zero, we end up with the following update:

$$k_{t_n} = \left(\frac{\sum_{i=1}^{n} \left(a_{x_i}^{y_i}(t_n) - a_{x_i}^{\check{y}_i}(t_n) \right)^2}{\sum_{i=1}^{n} \left(a_{x_i}^{y_i}(t_n) - a_{x_i}^{\check{y}_i}(t_n) \right) \left(b_{x_i}^{y_i}(t_n) - b_{x_i}^{\check{y}_i}(t_n) \right)} \right)^2 \tag{5.10}$$

where

$$\begin{aligned} a_{x_i}^{y_i}(t) &= \sum_{v \in \mathcal{N}_t(x_i) \cap \mathcal{N}_t(y_i)} \frac{\exp(-\frac{1}{k_{t-1} \Delta_{x_i,v} t})}{d_v \sqrt{\Delta_{x_i,v} t}} \\ b_{x_i}^{y_i}(t) &= \sum_{v \in \mathcal{N}_t(x_i) \cap \mathcal{N}_t(y_i)} \frac{1}{d_v}. \end{aligned}$$

$a_{x_i}^{\check{y}_i}$ and $b_{x_i}^{\check{y}_i}$ are also defined in the same way with respect to $\check{y}_i$. The online learning procedure is summarized in Algorithm 1.

The condition in line 7 is checked to prevent unnecessary updates in places where we would not be able to push the rank of y_i to the top for whatever value of k we choose.

5.2.4 Complexity analysis

The complexity of our algorithm is the same as the complexity of other local scores like CN, RA and AA. In contrast to to RA, every edge connected to x contributes $1 + \hat{\varphi}_t(x, y)$ instead

Algorithm 1 Online learning of the parameter

```
1: **Input:** graph G, data $x_i$
2: Initialize $p = q = 0.1$
3: **for** $i = 1, ...$ **do**
4:     Receive $x_i$ at time $t_i$
5:     Calculate rank $r_{x_i}$
6:     Suffer loss $l(y_i, r_{x_i})$
7:     **if** $\left(a^{y_i}_{x_i}(t_n) - a^{\check{y}_i}_{x_i}(t_n)\right)\left(b^{y_i}_{x_i}(t_n) - b^{\check{y}_i}_{x_i}(t_n)\right) < 0$ **then**
8:         $p = p + \left(a^{y_i}_{x_i}(t_n) - a^{\check{y}_i}_{x_i}(t_n)\right)^2$
9:         $q = q - \left(a^{y_i}_{x_i}(t_n) - a^{\check{y}_i}_{x_i}(t_n)\right)\left(b^{y_i}_{x_i}(t_n) - b^{\check{y}_i}_{x_i}(t_n)\right)$
10:        $k_{t_i} = (p/q)^2$
11:    **end if**
12: **end for**
```

of 1. The updates in Algorithm 1 are done in $O(1)$ when we store $a^{\check{y}_i}_{x_i}$ and $b^{\check{y}_i}_{x_i}$ during the rank computation. The complexity of all the aforementioned local scores are the same and will depend on local degrees. For a typical vertex x with max degree d between him and his neighbors, we would need $O(d^2)$ operations. This algorithm is trivial to parallelize in most distributed computing architectures e.g. shared memory, map-reduce etc.

5.3 Experiments

First we describe the dataset and the evaluation metric used for experiments. For the evaluation we use the friendship graph of the *Tuenti* Social Network.

Data *Tuenti* is Spain's leading online Social Network in terms of traffic. Over 80% of Spaniards aged 14-27 actively use the service and today (2013) it counts more than 14M users and over a billion daily page views. *Tuenti* allows users to chat with friends, post photos, share videos and play games. The network was launched in April 2006. Up until the summer of 2013 *Tuenti* was an invitation only network. We use a snapshot of the network from April 2012 which includes the timestamps of the friendship creation. The network attracted more than 12 million users in this period, and formed around 760 million friendships(edges). We will compare our algorithm to the baseline algorithms CN, AA and RA.

We tried hard to find relevant public datasets, but unfortunately none of them was suited for our application. There are few small to medium sized publicly available social network datasets containing the formation time of friendships, but come from a crawling process. As a side-effect, these graphs are incomplete, not fully connected and missing a big fraction of nodes.

Procedure and Metrics In the experiments we use Reciprocal Rank (RR) as an evaluation metric. Since we have access to the exact appearance time of each edge in the *Tuenti* graph, we can use a more precise evaluation procedure than the ones typically used in the literature. Assume that at time t, user u acquires friendship with user v. We evaluate each algorithm by assigning a score to each candidate friend of u and ranking them by this score. In this on-line

Table 5.1: Reciprocal Rank results on 200,000 new edges at given periods of time

	CN	AA	RA	TR	TR Log
Dec 2011	3.61	3.85	4.08	4.43	4.60
March 2012	3.68	3.94	4.19	4.67	4.85

evaluation scenario each algorithm gains the reciprocal rank $1/r[v]$ where $r[v]$ is the ranking, induced by the scores of the actual user v that became friend with u at time t.

In this scenario, other evaluation metrics like precision will not be meaningful. Given the ranking of candidate friends, precision is defined as the ratio of real-world friends to the number of friends suggested (Lü and Zhou, 2011). Essentially the measure captures the fraction of relevant users v (friends) in a ranked list L. In our on-line evaluation scenario, it is very likely that user u acquired friendships to users in the list L at different times. For example u can meet a person from L and acquire friendship a year after this suggestion happens. So this make the metric problematic. One possible solution is to only consider the first k future friends of u and find its intersection with L. This correction is also disputable. For a non-active user, the acquirement of k friends will take a long time. However an active one can acquire all k friends in a day.

AUC is another metric used in the literature (Lü and Zhou, 2011). It can be interpreted as the probability that a randomly chosen edge gets higher score than a ground truth edge. For large sparse networks, the AUC measure is not a suitable measure. In our data set more than 0.99999 percent of users look irrelevant for prediction, and the AUC metric will not be interpretable.

Results We evaluate all algorithms for predicting new friendship at two different times: December 2011 and March 2012. At each period we test over 200,000 recent edges from the social graph. We run experiment with two versions of our score function. In the first version called *TR*, we use the score function from Equation 5.8. In the second version, inspired from AA scores, we replace the weight factor d_v by factor $log(d_v)$ and call it *TR_Log*. We report the normalized cumulative reciprocal rank over all users in the test set $\mathcal{T}$ in a scale of 0-100 i.e.:

$$RR = \sum_{v \in \mathcal{T}} \frac{100}{r[v]|\mathcal{T}|}.$$

Results in Table 5.1 show the significant improvement of performance that our method achieves over the baseline methods.

It is worth mentioning that although the baseline methods seem simple, they are very hard to beat, particularly in the context of social networks. We observe that in Sarkar et al. (2012), for non-seasonal networks, these baselines perform similar to their proposed algorithm in a range of different datasets.

In evaluating two algorithms A and B using reciprocal rank, the following issue can occur: Although B has a larger reciprocal rank than A, they do perform similarly in real world applications: the superiority of B in recommending friends to user u is only when its subsequent friend v is at the bottom of the ranking list (which is never used for recommendation in practice). To demonstrate the superiority of our approach in recommending new friends at the top of the list, we show the distribution of predicted friends at the top-5 position of

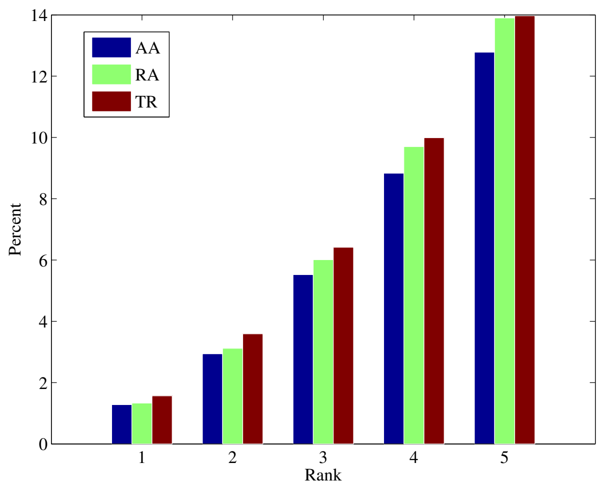


Figure 5.2: Frequency of hitting top 5 ranks

the recommendation list as generated by each method. Figure 5.2, shows that our method outperforms the alternative methods in recommending new friends at the top of the list. We plot the frequency with which new friends v appear in the positions 1 to 5 of the lists generated using the corresponding methods. We observe that our algorithm hits the top-5 rankings more often than others.

Local scores are only defined for users in the second layer neighborhood. We perform an experiment to check the soundness of this assumption. We count how often a friendship happens between a person u and a user from $\mathscr{N}(u)$. At the first part of experiment in December 2011, we observed that 89% of friendships appeared between users and their friend of friends. In March 2012 this ratio was 90%, which confirms the use of local scores.

Overall we observe that our method outperforms the next best method (RA) by 13-16% in terms of reciprocal rank while also providing better performance at the top-5 positions of the list. This validates our approach for taking the temporal information of the link creation into account in the diffusion distance. Note that this performance improvements are delivered without any significant additional computational costs compared to the methods in test (CN, AA, RA).

5.4 Conclusions

We proposed a novel diffusion distance that takes into account temporal information in the edge creation process. We derive a local approximation of the diffusion distance and an algorithm for computing the diffusivity parameter. The kernel can be computed very efficiently. We conducted experiments on the *Tuenti* social graph which clearly demonstrate that using the diffusion kernel as a distance measure improves upon other local measures such as CN, AA and RA.

Part III

Downsampling Random Geometric Graphs

Chapter 6

Downsampling a neighborhood graph

6.1 Introduction

Large graphs are hard to process, analyze and visualize. To overcome this problem, many attempts have been made to reduce the size of the graph. Examples are found in many areas, including machine learning (Liu et al., 2010,de Silva and Tenenbaum, 2002), social network analysis (Leskovec and Faloutsos, 2006, Krishnamurthy et al., 2005) and visualization (Rafiei and Curial, 2005). See also Section 6.5 for many more references and related work. Typically, the size of the graph is reduced by removing edges or vertices, collapsing edges or vertices or alternatively building another graph that reflects some aspects of the original graph.

We say a graph G' is downsampled from G whenever G' has fewer vertices and for every vertex $v' \in G'$ exists at least one corresponding vertex $v \in G$. The ultimate goal in downsampling is to find a procedure that reduces the size of the graph, but at the same time keeps invariant those properties that are essential for our application. For example, in visualization the downsampled graph should "look like" the original one. In community detection, two vertices in the downsampled graph should belong to the same "community" whenever they are in the same community in G.

It is rather obvious that there cannot exist a single downsampling procedure that is "best" for all applications. However, it would be tedious to find a procedure for every single purpose. There are a couple of basic graph properties that are commonly used in many algorithms: degrees, distances between vertices, properties of random walks on the graph, spectral properties, and so on. Is it possible to construct a downsampling procedure that keeps many of these properties intact simultaneously? If we do not make any assumptions on the graph at hand, the answer will be no. However, in this chapter we show that we can achieve this goal if we have extra knowledge about the graph at hand, namely if it comes from a certain class of graphs.

In this chapter we focus on the class of graphs that is most important for machine learning, namely neighborhood graphs like k-nearest neighbor graphs or ε-graphs. Mathematically, such graphs can be modeled by the family of random geometric graphs (RGGs). We define a strong notion of similarity between a neighborhood graph G and its downsampled version G' called geometry-preserving downsampling. We prove that it simultaneously keeps many basic properties of the original graph intact. We give a simple downsampling procedure for neighborhood graphs that can be proved to be geometry-preserving.

Finally, we investigate what happens if we apply other common graph sampling procedures to neighborhood graphs, such as the downsampling procedures that are often used for power-law graphs. We show that any such procedure that keeps the degree distribution invariant necessarily has to destroy the geometric information in the neighborhood graph.

6.2 Problem setting and basic definitions

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be a closed connected set endowed with a density function p with respect to the Lebesgue measure, $X_1, ..., X_n$ an i.i.d. sample from $\mathcal{X}$, and $D : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a metric on $\mathcal{X}$. We say the set $V := \{X_1, ..., X_n\}$ satisfies the **dense sampling assumption** with neighborhood size ς, if for all $x \in \mathcal{X}$ there exists a $y \in V$ with $D(x, y) \leq \varsigma$.

Consider the random geometric graph $G_{p,n,D,r} := (V_{p,n}, E_{V,D,r})$ that has the vertex set $V_{p,n} = \{X_1, ..., X_n\}$ and edge set

$$E_{V,D,r} := \{\{X_i, X_j\} \mid D(X_i, X_j) \leq r\}$$

for some parameter $r > 0$. This family of graphs not only contains the well-known ε-graphs, but also by an appropriate choice of D we can model k-nearest neighbor graphs. In this chapter, all graphs are undirected and unweighted. All results can directly be carried over to the case of graphs whose edges are weighed by the distance $D(X_i, X_j)$ of the adjacent points, but in order to keep notation simple we stick to unweighted graphs in this chapter. We define the shortest path distance $D^G_{sp}(x, y)$ between two vertices in graph G as the sum of the number of edges in the shortest path. For an edge set E and a vertex v we denote by $E(v)$ the set of edges that are adjacent to v.

Now assume that we are given the adjacency matrix of the graph $G_{p,n,D,r}$ but we do not know the point locations $X_i \in \mathbb{R}^d$, nor do we know the underlying distances $D(X_i, X_j)$. Our goal is to construct a downsampled graph G' with $n' < n$ vertices that "looks like" a graph of the form $G_{p,n',D,\tau r}$. In other words, given a large graph G from some unknown latent space model, find a downsampled small graph G' that behaves as if it were constructed from the same latent space model directly.

Definition 6.1 (Geometry-preserving downsample of G) *Consider a random geometric graph $G_{p,n,D,r}$. We say that a graph $G' = (V', E')$ is a δ-geometry preserving downsample ("δ-GPD") of G if the following conditions are satisfied:*

1. The new vertex set is a subsample of the old one: $V' \subset V$.

2. The vertices in V' are distributed according to the original density p:
For all Borel-sets $A \subseteq \mathcal{X}$ and all $v \in V'$ we have

$$P(v \in A) = \int_A p(x)dx.$$

3. The neighborhoods in E' approximate the ones of a neighborhood graph that is built on V' directly: There exists some $\tilde{r} > 0$ such that the graph $\tilde{G} = (V', \tilde{E})$ with $\tilde{E} := E_{V',D,\tilde{r}}$ satisfies *that for all vertices $v \in V'$*

$$1 - \delta \le \frac{|E'(v) \cap \tilde{E}(v)|}{|E'(v) \cup \tilde{E}(v)|}.$$

In Section 6.4 we show that the notion of geometry-preserving downsampling is very strong. It implies that both local and global properties of the graph are kept invariant. In Section 6.3 we show that one can construct geometry-preserving downsamples by drawing a uniform subsample of V and connecting the new vertices whenever their shortest path distance in the original graph is smaller than some parameter $\tau > 0$. This result applies to ε-graphs (rather obvious) and to k-nearest neighbor graphs (surprising).

Let $f : \mathcal{X} \to \mathbb{R}^+$ be a Lipschitz continuous scalar function with Lipschitz constant L_2. Consider a path $\gamma : [0,1] \to \mathcal{X}$ parametrized by t that connects x to y. We define the f-length of this path as

$$D_{f,\gamma} = \int_0^1 f(\gamma(t))|\gamma'(t)|dt.$$

This is also called as the *line integral* along γ with respect to f (see Section 2.2).

To keep proofs simple, we make the following general assumptions in the rest of the chapter. The density p is Lipschitz continuous with Lipschitz constant L_1 and bounded away from 0 by $p_{\min}$. The density dependent function f that defines the distance D_f is Lipschitz continuous with Lipschitz constant L_2 and $\tilde{f}(p_{\min}) > 0$. The volume of the unit ball in $\mathbb{R}^d$ will be denoted by η_d.

6.3 Constructing a geometry-preserving downsample

6.3.1 Vertex selection

In this section we study different approaches for the vertex selection step. The easiest one is to select vertices uniformly random from the original graph. This procedure is super fast and can applied on very large graphs. It obviously satisfies the first two conditions on geometry-preserving downsampling, and this is also the one we are going to use later on.

Proposition 6.2 (Uniform vertex selection) *Let $V = \{X_1, ..., X_n\}$ be an i.i.d. sample from density p, and let $V' = \{Y_1, ..., Y_{n'}\}$ be a subsample from V that has been drawn uniformly at random without replacement. Then the vertices in V' are distributed as the ones in V, that is $Y_1, ..., Y_{n'}$ is distributed as an i.i.d. n'-sample from p.*

Proof. Let σ be a permutation of $\{1, ..., n\}$ such that $Y_j = X_{\sigma(j)}$. Now consider any measurable set $A \subseteq \mathcal{X}$. We have

$$P(Y_i \in A) = P(X_{\sigma(i)} \in A) \overset{a}{=} P(X_i \in A),$$

where in a we use the assumption that X_i's are i.i.d. samples. □

There exist several approaches for the vertex selection in the literature, for example procedures based on landmarks (Goldberg and Harrelson, 2005, Silva et al., 2005), graph coarsening (Karypis and Kumar, 1999, Satuluri and Parthasarathy, 2009), based on clustering (Zhang et al., 2009) or based on some random-walk based exploration of the graph (Leskovec and

Faloutsos, 2006). In general, these procedures change the density of the selected vertices, so Property 2 of geometry-preserving downsampling will not be satisfied.

In Chapter 7 we propose a new procedure for vertex selection that enjoys an interesting property: Even if we choose the neighborhood size r' "very small", the resulting graph G' would be connected. However in chapter, we stick with the simple random selection method.

6.3.2 Edge construction

Given a set $V' \subset V$ of vertices, the question is how edges should be defined on V'. We use the following simple rule based on shortest paths: Fix an integer $\tau > 0$. For any two points $v_i, v_j \in V'$ put an unweighted edge between them if and only if $D^G_{sp}(v_i, v_j) \leq \tau$. We denote the resulting edge set by $\hat{E}_G(V', \tau)$.

In the next theorem we prove that the graph constructed by selecting vertices uniformly at random and connecting them by the shortest path rule is geometry-preserving.

Theorem 6.3 (Geometry-preserving downsampling) *Consider a random geometric graph $G = (V_{p,n}, E_{V,D,r})$. Let $V' \subseteq V_{p,n}$ be a sample from $V_{p,n}$ that has been drawn uniformly at random without replacement. Build a graph $G' = (V', E')$ by the shortest path based graph construction procedure $E' = \hat{E}_G(V', \tau)$ and set*

$$\lambda = \frac{L_1 L_2 \tau r}{f_{\min}} \ , \ \lambda' = \frac{\varsigma L_1 L_2}{f^2_{\min}} \ , \ c_1 = \frac{\varphi(1+\lambda')^d 2^d}{(1-\lambda')^{2d}\eta_d} \ , \ c_2 = \frac{p_{\min}\eta_d}{3(1+\lambda')^d f^d_{\max}} \ ,$$
$$c_3 = \frac{f_{\min}}{p_{\min} L_2} \ , \ c_4 = \frac{1}{3} p_{\max}(1 - c_3\lambda)\Big(f_{\max}(1-\lambda)(1 - 2\frac{\varsigma}{r})\Big)^d \eta_d \ ,$$
$$\delta = \frac{2\kappa}{1+\kappa} + \frac{2c_3\lambda}{1+c_3\lambda} + \frac{2d\lambda}{1+\lambda} + \frac{2d\varsigma}{r}.$$

Then with probability at least

$$1 - 2n' e^{-c_4 \kappa^2 n' \tau^d r^d} - c_1 \varsigma^{-d} e^{-c_2 n \varsigma^d},$$

the graph G' is a δ-GPD of G.

Moreover as $n, n' \to \infty$, $r, \tau r \to 0, r^d n/\log(n) \to \infty, \tau^d r^d n'/\log(n') \to \infty$ and by selecting $\varsigma = (\log(n)/n)^{1/d}$, the probability converges to 1 and δ converges to 0.

Before proving the theorem we need to state a couple of lemmas and propositions. The next lemma uses the Lipschitz continuity of p and f to show that $D_f(x, y)$ can be approximated by $f(x)\|x - y\|$ in small intervals. Its proof is a direct generalization of Lemma 2.5.

Lemma 6.4 (Approximating D_f in small balls) *Consider any given $\lambda < 1$. If $\|x-y\| \leq \frac{f_{\min}\lambda}{L_1 L_2}$ then the following statements hold:*

1. We can approximate $f(y)$ by $f(x)$:

$$f(x)(1-\lambda) \leq f(y) \leq f(x)(1+\lambda).$$

2. We can approximate $D_f(x, y)$ by $f(x)\|x - y\|$:

$$(1-\lambda) f(x)\|x-y\| \leq D_f(x, y) \leq (1+\lambda) f(x)\|x - y\|.$$

Now we establish a relation between the parameter ς of dense sampling assumption and the probability that this assumption holds. By carefully choosing ς, the dense sampling assumption will hold with "high" probability.

Lemma 6.5 (Sampling lemma) *Assume $X_1, ..., X_n$ are sampled i.i.d. from density p and a constant $\varsigma < f_{\min}^2/(L_1 L_2)$ is given. Set*

$$\lambda = \frac{\varsigma L_1 L_2}{f_{\min}^2} \quad , \quad c_1 = \frac{\varphi(1+\lambda)^d 2^d}{(1-\lambda)^{2d}\eta_d} \quad , \quad c_2 = \frac{p_{\min}\eta_d}{3(1+\lambda)^d f_{\max}^d}$$

Then with probability at least $1 - c_1\varsigma^{-d}e^{-c_2 n\varsigma^d}$, for every $x \in X$ exists a $y \in X_1, ..., X_n$ such that $D_f(x, y) \leq \varsigma$.

Proof. Define the f-mass of a set $\mathcal{W}$ as $\mathcal{V}_f(\mathcal{W}) = \int_{\mathcal{W}} f(x)^d dx$ and set $\varphi = \mathcal{V}_f(\mathcal{X})$, the f-mass of our underlying space $\mathcal{X}$. Denote the probability mass of a volume $\mathcal{W}$ by $\mathcal{V}(\mathcal{W})$. We are going to prove that for every $y \in \mathcal{X}$ there exists a $x \in X_1, ..., X_n$ such that $\|x-y\| \leq \big(\varsigma/(1+\lambda)f(x)\big)$. Applying Lemma 6.4 will then lead to the result. The proof is a generalization of the covering argument in Section 2.7. We first cover $\mathcal{X}$ by balls with approximately equal f-masses. The centers of balls are chosen iteratively such that every center is outside of the balls we have so far. The ball $B_f(x, \varsigma)$ at point x has the radius $\frac{\varsigma}{(1+\lambda)f(x)}$. The f-mass of this ball is bounded by

$$\mathcal{V}_f(B_f(x,\varsigma)) \geq \frac{(1-\lambda)^d}{(1+\lambda)^d}\varsigma^d \eta_d.$$

Smaller balls $B_f(x, (1-\lambda)\varsigma/2)$ are all disjoint, so we can bound the total number of balls by

$$S \leq \frac{\varphi}{\mathcal{V}_f(B_f(x,(1-\lambda)\varsigma/2))} \leq \frac{\varphi(1+\lambda)^d 2^d}{(1-\lambda)^{2d}\varsigma^d\eta_d} = c_1\varsigma^{-d}.$$

Now sample from the underlying space $\mathcal{X}$. The probability mass of ball $B_f(x, \varsigma)$ is bounded by

$$\mathcal{V}(B_f(x,\varsigma)) \geq \frac{p_{\min}\varsigma^d\eta_d}{(1+\lambda)^d f(x)^d} \geq \frac{p_{\min}\varsigma^d\eta_d}{(1+\lambda)^d f_{\max}^d} = 3c_2\varsigma^d.$$

By applying the concentration inequality for binomial variables (Prop. 28 in von Luxburg et al., 2010), we bound the probability that a ball $B_f(x, \varsigma)$ does not contain any sample point ("is empty")

$$Pr(\text{ball } i \text{ is empty}) \leq e^{-c_2 n\varsigma^d}.$$

Finally using a union bound results in

$$Pr(\text{no ball is empty}) \geq 1 - \sum_i Pr(\text{ball } i \text{ is empty}) \geq 1 - c_1\varsigma^{-d}e^{-c_2 n\varsigma^d}.$$

□

The following proposition shows that $D_f(x, y)$ can be approximated by the rescaled graph shortest path distance and is a straightforward adaptation of Proposition 2.4 (see also Tenenbaum et al., 2000).

Proposition 6.6 (Approximating D_f by D_{sp}) *Consider a random geometric graph $G_{p,n,D_f,r} = (V_{p,n}, E_{V,D_f,r})$ that satisfies the dense sampling assumption with neighborhood size ς. Fix two vertices x and y of the graph. Then the following statement holds:*

$$D_f(x,y) \le rD^G_{sp}(x,y) \le \frac{D_f(x,y)}{1-2\varsigma/r}.$$

Proof of Theorem 6.3 . The sampling lemma shows that the dense sampling assumption with neighborhood size ς holds with probability at least $1 - c_1\varsigma^{-d}e^{-c_2 n\varsigma^d}$.

Let $\tilde{E} := E_{V',D,\tau r}$. Fix a vertex $x \in V'$. For any neighbor $y \in E'(x)$ of x, it follows from Proposition 6.6 and the construction rule of E' that $D_f(x,y) \le rD^G_{sp}(x,y) \le \tau r$. This shows that $E'(x) \subseteq \tilde{E}(x)$ and $E'(x) \cap \tilde{E}(x) = E'(x)$ and $E'(x) \cup \tilde{E}(x) = \tilde{E}(x)$. To have a bound on $|\tilde{E}(x)|$, note that all neighbors of x are in the set $\mathcal{T} = \{u \mid D_f(x,u) \le \tau r\}$ with probability mass

$$\mathcal{V}(\mathcal{T}) \le p(x)(1 + c_3\lambda)\Big(f(x)(1+\lambda)\tau r\Big)^d \eta_d.$$

Now consider a neighbor $y \in \tilde{E}(x)$ of x which is not in $E'(x)$ ($D^G_{sp}(x,y) > \tau$), then

$$D_f(x,y) \ge rD^G_{sp}(x,y)(1-2\varsigma/r) > \tau r(1-2\varsigma/r).$$

This means that in E', x is connected to all vertices in the set

$$\mathcal{Q} = \{u \mid D_f(x,u) \le \tau r(1-2\varsigma/r)\}.$$

The probability mass of $\mathcal{Q}$ is bounded by

$$\mathcal{V}(\mathcal{Q}) \ge p(x)(1 - c_3\lambda)\Big(f(x)(1-\lambda)\tau r(1-2\varsigma/r)\Big)^d \eta_d.$$

Let $Q \approx Binomial(n', \mathcal{V}(\mathcal{Q}))$ and $T \approx Binomial(n', \mathcal{V}(\mathcal{T}))$. Apply a concentration inequality for binomial random variables (see Proposition 28 in von Luxburg et al., 2010) to get

$$P\Big(Q \le (1-\kappa)\mathbb{E}(Q)\Big) \le \exp\Big(-\frac{1}{3}\kappa^2\mathbb{E}(Q)\Big)$$
$$P\Big(T \ge (1+\kappa)\mathbb{E}(T)\Big) \le \exp\Big(-\frac{1}{3}\kappa^2\mathbb{E}(T)\Big),$$

where $\mathbb{E}(Q) = n'\mathcal{V}(\mathcal{Q})$ and $\mathbb{E}(T) = n'\mathcal{V}(\mathcal{T})$. This implies that with probability at least

$$1 - \exp\Big(-\frac{1}{3}\kappa^2\mathbb{E}(Q)\Big) - \exp\Big(-\frac{1}{3}\kappa^2\mathbb{E}(T)\Big) \ge 1 - 2\exp\Big(-\frac{1}{3}\kappa^2\mathbb{E}(Q)\Big)$$

we have

$$\frac{|E'(x) \cap \tilde{E}(x)|}{|E'(x) \cup \tilde{E}(x)|} = \frac{|E'(x)|}{|\tilde{E}(x)|} = \frac{Q}{T} \ge \frac{(1-\kappa)\mathbb{E}(Q)}{(1+\kappa)\mathbb{E}(T)} = \frac{(1-\kappa)\mathcal{V}(\mathcal{Q})}{(1+\kappa)\mathcal{V}(\mathcal{T})} \overset{a}{\ge} 1 - \delta, \tag{6.1}$$

where in a we use the Bernoulli inequality $(1-x)^d \ge 1 - dx$. By applying a union bound over n' vertices in V', the Equation 6.1 will hold uniformly over all vertices in G'. For the convergence condition, note that if we set $\varsigma = (\log(n)/n)^{1/d}$, then $\varsigma/r \to 0$. The remaining can easily checked by simply replacing values. □

6.4 Properties of geometry preserving downsampling

In the last section we have seen how to construct a GPD for a given random geometric graph. Now we discuss the mathematical properties of a graph that has been constructed by a GPD procedure. In this whole section, we use the following notation: $G := G_{p,n,D,r}$ denotes the large geometric graph, $G' = (V', E')$ with $E' = \hat{E}_G(V', \tau)$ denotes its downsampled graph, and $\tilde{G} = (V', \tilde{E})$ denotes the corresponding small geometric graph with $\tilde{E} = E_{V',D,\tau r}$.

6.4.1 Positive results

The following corollary is a direct consequence of Theorem 6.3.

Corollary 6.7 (Invariants in geometry preserving downsampling)
Consider any three vertices $v_1', v_2', v_3' \in V'$ in G' and their corresponding vertices $v_1, v_2, v_3 \in V$ in graph G. Also denote the steady state distribution of a random walk on G by π and on G' by π'. If the convergence conditions in Theorem 6.3 hold, with probability converging to 1 we have:

1. *The relative degrees converge and stationary distributions converge to each other:*

$$\left|\frac{\deg(v_1')}{\deg(v_2')} - \frac{\deg(v_1)}{\deg(v_2)}\right| \to 0 \qquad \textit{and} \qquad \left|\frac{\pi'(v_1')}{\pi'(v_2')} - \frac{\pi(v_1)}{\pi(v_2)}\right| \to 0.$$

2. *The relative shortest path distances converge to each other:*

$$\left|\frac{D_{sp}^{G'}(v_1', v_2')}{D_{sp}^{G'}(v_1', v_3')} - \frac{D_{sp}^{G}(v_1, v_2)}{D_{sp}^{G}(v_1, v_3)}\right| \to 0.$$

Proof. Part 1 is a direct consequence of Theorem 6.3, the statement about the stationary distribution comes from the fact that $\pi(v)$ is proportional to $\deg(v)$. Finally, Part 2 can proved by combining the result of Theorem 6.3 and the relation between the shortest path distance and the f-distance in Proposition 6.6. □

The first two statements in this corollary only concern local quantities and are not particularly strong, it is relatively easy to come up with downsampling procedures that keep the relative degrees intact. The third statement is pretty strong because it concerns the global geometry of the graph. Intuitively, it says that the shortest path distances in the original graph and the downsampled graph behave similar.

On top of that, we can get even stronger results on the global geometry of the graphs. To this end, it is important to realize that Property 3 of the definition of geometry-preserving downsampling is a very strong property. If two graphs share the same neighborhoods, the overall geometric structure of the graphs is very similar. One way to prove this is to consider the eigenvalue structure of the adjacency matrix or the graph Laplacian matrix. It is well known that these spectra encode many of the geometric properties of a graph, for example how clustered it is. The following proposition shows that the spectrum of the downsampled graph G' is similar to the small geometric graph $\tilde{G}$.

Proposition 6.8 (Invariances in the spectrum) *Consider two graphs $G' = (V', E')$ and $\tilde{G} = (V', \tilde{E})$ with $|V| = n'$ whose edges satisfy*

$$1 - \delta \le |E'(v) \cap \tilde{E}(v)| / |E'(v) \cup \tilde{E}(v)|,$$

and whose degrees are of the order $\Theta(\log n')$. Denote the adjacency matrix of G' by A', its degree matrix with the degrees on the diagonal by D', and its graph Laplacian by $L' = D' - A'$. Let σ'_i be the i-th eigenvalue of the adjacency matrix and λ'_i the one of the graph Laplacian. Use the analogous notation for $\tilde{G}$. Then the average deviations between the spectra of G' and $\tilde{G}$ satisfy

$$\frac{1}{n'}\sum_{i=1}^{n'}(\tilde{\sigma}_i - \sigma'_i)^2 = O(\delta \log n') \quad \textit{and} \quad \frac{1}{n'}\sum_{i=1}^{n'}(\tilde{\lambda}_i - \lambda'_i)^2 = O(\delta \log n').$$

In our case, it can be seen that this proposition applies if we choose $\tau^d \approx \log n'/(n'r^d)$, in which case the quantity δ in Theorem 6.3 is of the order $O(\log n'/n')^{1/d}$. This shows that both average differences converge to 0 for large n'. Note that this is a non-trivial result, because the eigenvalues of A' and L' scale with n'. In particular, the graph Laplacian matrix has only a few eigenvalues close to 0, and most of them are much larger.

Proof. (Sketch) To prove this proposition we use the eigenvalue perturbation theorem by Wielandt-Hoffman (e.g., Golub and Van Loan, 1996 Section 8.1.2.) which bounds the deviation of eigenvalues in terms of the Frobenius norm of the perturbation. Precisely, for symmetric matrices A' and $\tilde{A}$ it holds that

$$\textstyle\sum_i(\tilde{\sigma}_i - \sigma'_i)^2 \leq \|\tilde{A} - A'\|_F^2 := \sum_{ij}(\tilde{a}_{ij} - a'_{ij})^2.$$

In case of the adjacency matrix, the right hand side can now bounded by exploiting the result of Theorem 6.3 on the intersection of edge sets in G and G', which gives

$$\|\tilde{A} - A'\|_F^2 \leq \delta\|\tilde{A}\|_F^2 = O(\delta n' \log n').$$

The same bound holds for $\|\tilde{D} - D'\|$, which then leads to the result for the graph Laplacians. □

6.4.2 Negative results

Another common family of random graphs are power-law graphs. Their main property is that their degree distribution follows a power law, that is the number of vertices with degree t is proportional to t^a for some parameter a. When downsampling a power law graph, one would like to keep this distribution invariant, that is we want a smaller graph whose degrees still follow the same power law. Denote by $n(t)$ the number of vertices in G that have degree t, and analogously $n'(t)$ in G'. Intuitively, the vertices in G' still follow the same power law as the ones in G if $n(t)/n'(t) \approx n/n'$.

Several procedures to achieve such a relationship have been suggested in the literature, see for example Leskovec and Faloutsos (2006). We now show that if we apply such a procedure to a geometric graph, we loose information about its geometry.

Proposition 6.9 (Downsampling that is not a GPD) *Consider an ε-graph G built on the random sample $V = \{X_1, ..., X_n\}$ from some non-uniform density on $\mathbb{R}^d$. Consider a graph G' that satisfies $n(t)/n'(t) \approx n/n'$. Then G' is not a GPD of G.*

Proof. (Sketch) The expected degree of a vertex $x \in G$ is the expected number of points in an ε-ball around x. For ε small enough, it can be approximated by $\mathbb{E}(\deg x) \approx np(x)\varepsilon^d\eta_d$, where η_d denotes the volume of the d-dimensional unit ball.

Fix a natural number t and consider the set $S \subseteq \mathcal{X}$ that contains all points with expected degree t. The expected number of vertices in G that have degree t is given by $\mathbb{E}(n(t)) = n\mathcal{V}(S)$ where $\mathcal{V}(S)$ is the probability mass of S.

Assume G' is a geometry preserving downsample of G with n' vertices and connectivity parameter ε'. Denote $\alpha := n'\varepsilon'^d/(n\varepsilon^d)$. In G', every vertex $x \in S$ has the expected degree $\mathbb{E}(\deg x) = t \cdot (n'\varepsilon'^d)/(n\varepsilon^d) = t\alpha$. Similarly as above, $\mathbb{E}(n'(t\alpha)) = n'\mathcal{V}(S)$.

Together, this shows that for every t

$$\mathbb{E}(n(t))/\mathbb{E}(n'(t\alpha)) \approx n(t)/n'(t)$$

or $\mathbb{E}(n'(t\alpha)) \approx n'(t)$. This can only happen when the density is uniform. □

6.4.3 Consequences for practice

Parameter choice. Let us address the issue of choosing the connectivity parameter in the downsampling procedure. Assume we are given large graph $G := G_{p,n,D,r}$ and want to downsample it to n' vertices. On an abstract level, Theorem 6.3 requests that τ satisfies $n'\tau^d r^d/\log n' \to \infty$. Also by this condition, the sampled graph will stay connected. Hence, choosing $\tau = (n \log n'/n' \log n)^{1/d}$ or any other value of the order larger than that will work. In practice we can approximate it by choosing $\tau = (n/n')^{1/d}$.

Special case ε-graph. Consider the case of an ε-graph with n vertices that has been built by connecting all points that have Euclidean distance smaller than ε. Our results say that if we subsample n' vertices uniformly at random, and we connect them whenever their shortest path distance D^G_{sp} in the original graph was smaller than $(n/n')^{1/d}$, then we obtain a graph G' that is very similar to the ε'-graph on the subsampled vertices, with $\varepsilon' \approx \varepsilon(n/n')^{1/d}$.

Special case kNN graph. For an unweighted kNN graph the story is similar: Start with a kNN graph with n vertices, where the nearest neighbors have been computed according to the Euclidean distance in the underlying space. Now downsample to n' vertices and connect the new vertices whenever their shortest path distance in the original graph was smaller than $(n/n')^{1/d}$. Then the resulting graph resembles the kNN-graph on V' with parameter $k' \approx k$.

Case of weighted graphs. All our results can also be extended to the case of graphs whose edges are weighted by the D-distances of the adjacent points. In this case, to downsample such a weighted graph connect two vertices in the new graph whenever the weighted shortest path distance in the old graph is less than $r(n \log n'/n' \log n)^{1/d}$ (r is the connectivity parameter in the original graph).

Finally, consider the choice of n'. In principle, our theorems hold for any values of n and n'. However, if n' is too small compared to n, then two things will happen. First, the quantity δ that controls the deviations in the $\delta - GPD$ will get very large, leading to large differences between the original graph and the downsampled graph. Second, the probabilities that our bounds hold gets smaller, meaning that there is a higher risk that the procedure fails.

6.5 Related work and discussion

Sampling from a graph refers to the following problem: we are interested in a certain quantity of the graph, say the average vertex degree or the diameter of the graph. But because the graph is so big we cannot compute it exactly, but want to estimate it by a sampling procedure. For example, we could sample a couple of vertices and compute the empirical average degree or the diameter based on this sample. This problem is very different from graph downsampling, see Chapter 5 of Kolaczyk (2009) for an overview and references.

Graph coarsening. There exists a huge variety of papers that first "coarsen" a graph (reduce number of vertices and edges), then apply a particular algorithm to the graph, and in the end extend the results to the large graph ("refinement"). In all of this work, the focus is not so much on the properties of the downsampled graph, but on *enabling a particular algorithm of interest to achieve results on the small graph that "resemble" the results one would obtain on the large graph.* As examples consider the METIS algorithm (Karypis and Kumar, 1999) or fast approximate spectral clustering (Yan et al., 2009).

Graph sparsification. Given a large graph, the goal is to *construct a smaller graph such that certain properties of interest stay invariant.* Such properties could be the cut values of particular partitions (Benczúr and Karger, 1996), the values of certain flows (Leighton and Moitra, 2010), the spectrum of the graph (Spielman and Srivastava, 2008), or shortest path distances in the graph (Kleinberg et al., 2004, Ruan and Jin, 2011). Most of the literature is concerned with edge sparsification (keep all vertices, reduce the number of edges), but there exist also papers on vertex sparsification (reduce the number of vertices and edges). The literature on graph sparsification usually comes with strong theoretical guarantees, but sometimes the suggested procedure is not practical.

Model-preserving graph downsampling is the line of work discussed in this chapter. Its connotation is different from graph sparsification. We are given a large graph that comes from some known family $\mathcal{M}(n)$ of random graphs, but with unknown parameters. The goal is to *downsample the graph such that the new graph resembles a small graph that comes from the same family* $\mathcal{M}(n')$. The general intuition is that the downsampled graph should preserve as many as possible of the characteristic properties of the graphs in the model class $\mathcal{M}$, not just a particular one. Such a procedure is preferable over graph sparsification algorithms when we want to apply several different algorithms to the downsampled graph, at least if we want to keep as much flexibility and loose as little information as possible. It is about the best we can do if we want to reduce the size of the graph but want to keep its "geometry".

Additional to our current work, there are some other relevant papers that fall into this framework, mainly in the context of downsampling power-law graphs. See Krishnamurthy et al. (2005), Leskovec and Faloutsos (2006) and references therein.

6.6 Conclusions

This chapter investigates the framework of geometry-preserving downsampling of random geometric graphs. Being geometry-preserving is a pretty strong property, because it implies that both local and global geometric properties of the graphs stay intact after downsampling. Our downsampling procedure is simple, yet implies strong theoretical guarantees. Our framework can be applied to all kinds of neighborhood graphs, such as ε-graphs and kNN graphs.

Chapter 7

Density-preserving quantization

7.1 Introduction

Vector quantization is the task of compressing a large set of data points into a small set of representatives called centroids or centers. Its applications are abundant, just consider the examples of visual word models in computer vision (Leung and Malik, 2001, Csurka et al., 2004), color quantization in image processing (Heckbert, 1982), or selecting landmark points (de Silva and Tenenbaum, 2004). For data points in $\mathbb{R}^d$, the standard approach is to minimize a quantization error measured in terms of the Euclidean distance. The most well-known algorithm is the k-means algorithm (where k is large, as opposed to clustering problems, where k is usually chosen much smaller). Because the selected centers are supposed to be a "faithful representation" of the original data points, it is a highly desirable property that the centroids have the same distribution as the original data points. However, it has been proved that this is not the case if we minimize the quantization error with respect to the Euclidean distance (Graf and Luschgy, 2000). The optimal centers that minimize the quantization error with respect to the Euclidean distance are distributed as a power of the original density instead of the density itself, unless the underlying density is uniform.

How can we find a set of centroids that matches the underlying density of our data? This is the key problem we study in this chapter. In an early work, Delp and Mitchell (1991) looked for a weaker property of centers, which is to match the moments of the underlying density. Hegde et al. (2004) explicitly minimize the KL-divergence between a kernel density estimation of the original data and the estimated density of the centroids by gradient descent. Hulle (1999) and Meinicke and Ritter (2001) use a similar approach to build a compact density estimator. Instead of relying on all samples to estimate the density, they select few points that best describes the density. Recently, Li et al. (2011) considered using the Rosenblatt transformation which transforms an arbitrary distribution to a uniform distribution. After quantizing the transformed data, one can use the inverse transformation to get back to the original density.

In this chapter, we introduce a completely new approach to the problem. The algorithm we suggest is conceptually simple: we construct an unweighted k-nearest neighbor graph on the sample points and use the k-mediod algorithm with respect to the shortest path distance on this graph for constructing the representatives. Quite surprisingly, we do not even need to have access to the data points themselves or to their Euclidean distances, our algorithm works as soon as we know who are the k-nearest neighbors of each data point. It also works for manifolds, and we do not even need to know the intrinsic dimension of the data. In Section 7.5, we present an application of our proposed algorithm in downsampling random geometric graphs.

The major part of this chapter is devoted to a thorough statistical analysis of our algorithm. We first introduce a new distance function D_{PD} on $\mathbb{R}^d$ which depends on the data density p. We then show that this distance function plays the role of a "uniformizing transformation" of the space: Quantizing p-distributed data with respect to the distance function D_{PD} behaves as quantizing uniform data with respect to the Euclidean norm. As a second step we then exploit that the D_{PD}-distance on $\mathbb{R}^d$ can be approximated by the shortest path distance in unweighted kNN graphs. Our main result is that if the original data points are distributed according to the density p, then the representatives selected by our algorithm will follow the same distribution (as the sample size and the number of centers converge to infinity).

7.2 Definitions and formal setup

Consider a connected and compact subset $\mathcal{X} \subset \mathbb{R}^d$ endowed with a probability measure P which has a Lipschitz continuous density p with Lipschitz constant L. Assume the set $V_n = \{X_1, ..., X_n\} \subset \mathcal{X}$ has been drawn i.i.d. according to p. Denote by P_n the empirical measure of the sample. The unweighted, undirected k-nearest neighbor (kNN) graph G_n is the graph with vertex set V_n where we connect X_j to X_i by an undirected edge if X_j is among k nearest neighbor of X_i or vice versa (according to the Euclidean metric). In the following, we use the letter k to denote the parameter of the kNN graph and the letter κ to denote the number of representatives for κ-means and κ-medoids. For two vertices $x, y \in V_n$, the shortest path distance $D_{sp}^{G_n}(x, y)$ is the length of the shortest path connecting x to y in G_n. When the underlying graph is clear from the context, we drop the index G_n and simply use the notation $D_{sp}(x, y)$.

Definition 7.1 (Following / resembling a density) *Let $(A_n)_{n\in\mathbb{N}}$ be a sequence of sets with $A_n \subset \mathcal{X}$, $|A_n| = n$. We say that A_n follows or resembles the density function p if for any measurable set $S \subset \mathcal{X}$, the fraction of points of A_n which lie inside S converges to the probability mass of S: $\frac{1}{n}|S \cap A_n| \to \int_S p(x)dx$ as $n \to \infty$ (Gruber, 2004).*

Let f be a positive, continuous, real-valued function defined on $\mathcal{X}$. We define the f-weighted length or f-length of a curve $\gamma : [0, 1] \to \mathcal{X}$ as

$$D_f(\gamma) = \int_0^1 f(\gamma(t))|\gamma'(t)|dt.$$

The f-distance between x and y is defined as $D_f(x, y) := \inf_\gamma D_f(\gamma)$ where the infimum is over all rectifiable paths γ that connect x to y (see also Chapter 2). Following the notations used in Chapter 2, we use the shorthand notation PD-distance for the f-distance with $f(x) = p(x)^{1/d}$. In Lemma 7.3 we show that the PD-distance induces a uniform structure on non-uniform densities.

7.3 Vector quantization

A vector quantizer maps vectors in $\mathbb{R}^d$ to a set $A = \{a_1, \cdots, a_\kappa\} \subset \mathbb{R}^d$. Each element a_i is called a centroid, a center or a representative. The set $\mathcal{V}_i \subset \mathbb{R}^d$ of vectors that are assigned to the centroid a_i is called a (Voronoi) cell. A common assignment is based on Euclidean distance: every point x is assigned to its nearest centroid with respect to the Euclidean distance. A widely used example of Euclidean distance quantizers is the least square quantization, better known as k-means (Lloyd, 1982). As we will work with other metrics as well, we formally define the centroid assignment function $\mathcal{C}_A : \mathbb{R}^d \to A$ which is the one that determines the cells $\mathcal{V}_i = \{x \in \mathbb{R}^d | \ \mathcal{C}_A(x) = a_i\}$. In particular, we consider the functions $\mathcal{C}_{A,\|\cdot\|}$, $\mathcal{C}_{A,\text{PD}}$ and $\mathcal{C}_{A,sp}$ which assign points to the closest center according to the Euclidean distance, the PD-distance and the shortest path distance, respectively.

The representation error $g(x, y) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ quantifies the error of representing y by x. In κ-means we have $g(x, y) = \|x - y\|^2$. The quality Φ of a set of centroids is measured by the expected representation error of the centroid mapping function

$$\Phi(g, \mathcal{C}_A, P) := \int g(\mathcal{C}_A(x), x) P(dx).$$

The set of optimal centroids of size κ with respect to the error function $\Phi(g, \mathcal{C}_A, P)$ is defined as the set A that minimizes the expected error

$$A^* = \underset{A, |A|=\kappa}{\operatorname{argmin}} \Phi(g, \mathcal{C}_A, P).$$

The set of empirical optimal centroids A_n is defined analogously with respect to $\Phi(g, \mathcal{C}_A, P_n)$. Finding the set of empirical optimal centroids A_n is an NP-complete problem in general. However, there exist EM type algorithms and several heuristics to find a good local optimum of $\Phi(g, \mathcal{C}_A, P_n)$. In this work we will not discuss details of these algorithms.

The topology of optimal centroids and shape of their corresponding cells are studied in several works, see Graf and Luschgy (2000) and Gruber (2004) for references. As a brief summary of their results, the optimal centroids in the case $g(x, y) = \|x - y\|^\alpha$ are distributed with density $p^{d/(d+\alpha)}$ when the number of centroids κ is large enough. A conjecture of Gersho (1979) states that for $g(x, y) = \|x - y\|^2$ and uniform p, there exists a convex polytope such that all cells are congruent to it as $\kappa \to \infty$. This conjecture is only proved for $d = 2$ where the convex polytope is a regular hexagon (Fejes Tóth, 2001, Gruber, 2001).

7.4 Quantization with the pd-distance

In this section, we introduce a new technique to transform a space with a non-uniform density to a space with uniform density. The PD-distance plays a key role in this construction. This transformation is defined for all smooth compact Riemannian manifolds, but later we are only going to use it for full-dimensional subsets of $\mathbb{R}^d$. Then we use our new transformation to construct a quantization procedure which keeps the distribution of centers correct. In Section 7.4.3 we then show how to approximate the quantization procedure and why this approximation works.

7.4.1 Uniformizing metric

Imagine a uniform elastic rubber band with printed lines graduated in centimeters. Now stretch the band in different places to get a non-uniform band. The printed lines will also displace and their Euclidean distances will change. However, these displaced lines show a uniformity property: The mass of the elastic band between successive lines is the same all over the stretched band. We show that PD-distance corresponds to such a stretched distance when the density function p corresponds to the density of the rubber at each point on the band. If an ant walks on this band such that it passes a fixed number of lines in a second, it would need to walk faster in low density regions and slower in high density regions.

Definition 7.2 (Uniformizing Metric) *Let (M, h) be a smooth compact Riemannian manifold with intrinsic dimension d, where h is the standard Riemannian metric. Let $p(x)$ be a continuous and smooth density defined on M. Consider the smooth Riemannian manifold (M, h^p) where*

$$h^p_x(u, v) := h_x(p(x)^{1/d}u, p(x)^{1/d}v)$$

for tangent vectors u and v at x. We call the metric h^p a metric uniformizing the density p.

Because the density is assumed to be strictly positive, the uniformizing transformation can be interpreted as a conformal change of metric with $h^p_x = p(x)^{2/d}h_x$. Let ϱ (resp. ϱ_p) and w (resp. w_p) denote the geodesic distance and the volume element on (M, h) (resp. on (M, h^p)). We denote the volume of a set $\mathcal{A} \subset \mathcal{X}$ in (M, h^p) as $\mathrm{Vol}_p(\mathcal{A})$.

Lemma 7.3 (Density gets uniform and pd-distance becomes Euclidean) *Consider a compact smooth Riemannian manifold (M, h) and its uniformizing transformation (M, h^p).*

- *The length of a smooth curve $\gamma : [0, 1] \to M$ in (M, h^p) is the PD-length of γ in (M, h).*
- *The geodesic distance between two points in (M, h^p) corresponds to their PD-distance in (M, h).*
- *The uniformizing transformation induces a uniform density on M: all sets $\mathcal{A}_1, \mathcal{A}_2 \subset \mathcal{X}$ with $p(\mathcal{A}_1) = p(\mathcal{A}_2)$ also have the same volume in (M, h^p), that is $Vol_p(\mathcal{A}_1) = Vol_p(\mathcal{A}_2)$.*

Proof. *Part 1.* The length of a smooth curve γ in (M, h^p) is defined as $\int_0^1 \|\gamma'(t)\|_{h^p}dt$ where $\|\cdot\|_{h^p}$ is the norm induced by the inner product on the tangent space $T^pM(\gamma(t))$ at point $\gamma(t)$. Using $\|x\|_{h^p} = p(x)^{1/d}\|x\|_h$ and the definition of the PD-length leads to the result.

Part 2. The geodesic distance is the infimum over the lengths of paths connecting the two points. Exploiting Part 1 will give the result. So $\varrho_p(c, x) = D_{\text{PD}}(c, x)$.

Part 3. From properties of the conformal change of a metric (see Theorem 1.159 in Besse, 1987) we get $dw_p(x) = p(x)dw(x)$. Then

$$\frac{\mathrm{Vol}_p(\mathcal{A}_1)}{\mathrm{Vol}_p(\mathcal{A}_2)} = \frac{\int_{\mathcal{A}_1} dw_p(x)}{\int_{\mathcal{A}_2} dw_p(x)} = \frac{\int_{\mathcal{A}_1} p(x)dw(x)}{\int_{\mathcal{A}_2} p(x)dw(x)} = \frac{p(\mathcal{A}_1)}{p(\mathcal{A}_2)} = 1.$$

□

7.4.2 Quantization with the exact pd-distance

From here on, we restrict ourselves to Euclidean spaces for the sake of simplicity. However, all theorems can be generalized to smooth manifolds. We are now going to study the behavior of centroids with respect to the PD-distance, and later the behavior of the empirical centroids with respect to the shortest path distance. In the following we always assume that the set of optimal centers is unique. Our results can also be extended in a straightforward manner to the non-unique case by studying the set of unique centers, but this introduces even more heavy notation. So we decided to just state our results for the unique case.

The next theorem is a variation of Theorem 1 in Gruber (2004) and specifies the distribution of the centroids with respect to the PD-distance. The intuition behind the theorem is that the density of κ-means centroids matches the data points density when the latter is uniform. So we use a distance measure that induces a uniform density on the underlying space.

Theorem 7.4 (Quantization with exact pd-distance) *Let $\mathcal{X}$ be a connected and compact subset of $\mathbb{R}^d$ endowed with a Lipschitz continuous density p. Assume that for all $x \in \mathcal{X}$ we have $0 < p_{\min} \leq p(x) \leq p_{\max} < \infty$. Let $A^{*\kappa}$ be an optimal κ-means centroid with respect to the PD-distance, that attains the minimum*

$$A^{*\kappa} = \underset{A,|A|=\kappa}{\operatorname{argmin}} \left\{ \int_{\mathcal{X}} \{\min_{c \in A} D_{\text{PD}}(c,x)^2\} p(x) dx \right\}. \tag{7.1}$$

Then $A^{*\kappa}$ is distributed with density p in $\mathcal{X}$ as $\kappa \to \infty$.

Proof. Using Lemma 7.3

$$\int \{\min_{c \in A} \varrho_p(c,x)^2\} dw_p(x) = \int \{\min_{c \in A} D_{\text{PD}}(c,x)^2\} p(x) dw(x),$$

where $w(x)$ is the Lebesgue measure on $\mathcal{X}$. Now Part 2 from Theorem 1 in Gruber (2004) shows that the centers minimizing

$$\int \{\min_{c \in A} \varrho_p(c,x)^2\} dw_p(x)$$

have uniform density with respect to the area measure $w_p(x)$ as $n \to \infty$. This means they are distributed with density $p(x)$ with respect to the Lebesgue measure. □

Theorem 7.4 shows that the optimal κ-means centers with respect to the PD-distance follow the underlying density of our data. However, it is neither easy to compute the PD-distance from p, nor from a set of i.i.d. sample points from p.

7.4.3 Quantization with approximate pd-distance

We now provide a simple and effective approach to approximate κ-means centers with respect to the PD-distance. The procedure is the following: First build an unweighted kNN graph G_n based on samples $x_1, ..., x_n \in \mathcal{X}$ from the density p with a proper k. Then quantize with respect to the graph shortest path distance in G_n. In Theorem 7.5 we show that the centers constructed by this procedure converge to optimal κ-means centers with respect to the PD-distance.

Define the representation error functions

$$g_{\text{PD}}(x,y) = D_{\text{PD}}(x,y)^2 \quad , \quad g_{sp}(x,y) = D_{sp}(x,y)^2. \tag{7.2}$$

Although all results in this chapter hold for $g_f(x,y) = D_f(x,y)^t$ with arbitrary $t \geq 1$, we restrict ourselves to the more common case $t = 2$.

Set the constant $\beta = \eta_d p_{\min}^{d+1}/(4L)^d$ where η_d is the volume of a Euclidean unit ball in $\mathbb{R}^d$ and L is the Lipschitz constant of the density p.

Theorem 7.5 (Convergence of κ-medoids center to κ-means centers) *Consider the unweighted kNN graph G_n based on the i.i.d. sample $x_1, ..., x_n \in \mathcal{X}$ from the density p. We choose k such that for a fixed $0 < \alpha \ll 1$,*

$$24\log(n)^{1+\alpha} < k < \beta\frac{n}{(\log n)^{\alpha}}.$$

Fix κ. Let A_n denote the optimal empirical κ-medoids in graph G_n with respect to the shortest path distance. We assume that the optimal κ-means centers for $\mathcal{X}$ with respect to the PD*-distance is unique and denote it by A^* . As n tends to infinity, the empirical κ-medoids centers A_n converge almost surely to A^* with respect to the Hausdorff distance.*

To prove this theorem, we need the Theorem 2.1 for the convergence of the shortest path distance to PD-distance in probability. In Lemma 7.6 we present a simplified version of that theorem. Then in Lemma 7.7 we use the Borel-Cantelli lemma to show the almost sure convergence of $\Phi(g_{sp}, \mathcal{C}_A, P_n)$, the normalized quantization error with respect to the shortest path distance, to $\Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)$, the quantization error with respect to the PD-distance. The price we pay to reach almost sure convergence instead of convergence in probability is to have a slightly stronger condition on k, the connectivity parameter of the kNN graph. To keep our proofs readable, we ignore boundary effects. One can show that the boundary effects are tiny and converge toward zero in the limit.

Lemma 7.6 (Convergence of D_{sp} to $D_{\mathbf{pd}}$) *Consider a sequence of i.i.d. samples $V_n = \{x_1, ..., x_n\} \subset \mathcal{X}$ from the density p. For any n we build an unweighted kNN graph G_n based on $x_1, ..., x_n$ and denote the shortest path distance on this graph by D_{sp}^n. Set $c_n = (\frac{k}{n\eta_d})^{1/d}$. Assume for a small fixed α, $24\log(n)^{1+\alpha} < k < \beta\frac{n}{(\log n)^{\alpha}}$. Set $\lambda = (\log n)^{-\frac{\alpha}{4d}}$ and let $n > 3 \cdot 2^{d+1}$.*

*Then with probability at least $1 - \frac{1}{n^2}$, for **all pairs** $x, y \in V_n$*

$$\frac{1}{(1-\lambda)^{1/d}} D_{\text{PD}}(x,y) \leq c_n D_{sp}^n(x,y) \leq \frac{1}{\frac{1}{(1+\lambda)^{1/d}} - 2\frac{(1+\lambda)^{1/d}}{k^{\alpha^2}}} D_{\text{PD}}(x,y).$$

Proof. In Theorem 2.1, set $k' = k^{1-\alpha^2}$. Then with probability at least $1 - \frac{1}{n^2}$, for **all pairs** $x, y \in V_n$:

$$\frac{1}{(1-\lambda)^{1/d}} D_{\text{PD}}(x,y) \leq c_n D_{sp}^n(x,y) \leq \frac{1}{\frac{1}{(1+\lambda)^{1/d}} - 2\frac{(1+\lambda)^{1/d}}{k^{\alpha^2}}} D_{\text{PD}}(x,y).$$

□

Lemma 7.7 (Convergence of quantization errors) *Consider the setting in Lemma 7.6. Let $n \to \infty$ and fix κ. Then for every subset $A \subset \mathcal{X}$ with $|A| = \kappa$, and any $\mathcal{C}_A$ with support in $\mathcal{X}$,*

$$|c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) - \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)| \to 0 \ \ a.s.$$

Proof. By the definition of $\Phi(g_{sp}, \mathcal{C}_A, P_n)$ and Lemma 7.6, the following inequalities hold with probability at least $1 - n^{-2}$:

$$\begin{aligned} c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) &= \frac{c_n^2}{n} \sum_{v \in V_n} D_{sp}(\mathcal{C}_A(v), v)^2 \geq \frac{1}{n(1-\lambda)^{2/d}} \sum_{v \in V_n} D_{\text{PD}}(\mathcal{C}_A(v), v)^2 \\ &= \frac{1}{(1-\lambda)^{2/d}} \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n). \end{aligned}$$

For the upper bound, set $e = 1/(\frac{1}{(1+\lambda)^{1/d}} - 2\frac{(1+\lambda)^{1/d}}{k^{\alpha^2}})$. Then we have

$$c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) \leq e^2 \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n) + c_n^2 + 2c_n e \Phi(D_{\text{PD}}, \mathcal{C}_A, P_n). \tag{7.3}$$

Note that in Lemma 7.6, the property holds for **all pairs** of vertices simultaneously. It is easy to show that for a fixed κ, $\Phi(g_{\text{PD}}, \mathcal{C}_A, P)$ and consequently $\Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)$ and $\Phi(D_{\text{PD}}, \mathcal{C}_A, P_n)$ are bounded. Using these inequalities, the boundedness of $\Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)$ and $\Phi(D_{\text{PD}}, \mathcal{C}_A, P_n)$, the Borel-Cantelli lemma and the convergence of hyperharmonic series $\sum i^{-2}$ leads to the almost sure convergence of $|c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) - \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)|$ to zero. □

Proof. [Theorem 7.5] *Overview:* The standard technique is to bound the difference between representation errors with A_n and A^* and show that it converges almost surely to 0. This is done by using the triangle inequality to bring the empirical representation errors into the play. Then the uniform strong law of large numbers shows the convergence of empirical representation errors (see, e.g., Pollard, 1981). In the proof of this theorem, we deal with extra intermediate terms that need be bounded using properties of the shortest path distance (see below).

Define $\mathcal{E}_\kappa = \{A \subset \mathcal{X} \mid |A| = \kappa\}$, the set of all possible sets of κ-centers. The set $\mathcal{E}_\kappa$ is compact under the topology induced by the Hausdorff metric. When we work with Euclidean distance, the map $A \to \Phi(g_{\|.\|}, \mathcal{C}_{A,\|.\|}, P)$ is continuous on $\mathcal{E}_\kappa$ (Pollard, 1981). This result can be extended to the continuity of $A \to \Phi(g_{\text{PD}}, \mathcal{C}_{A,\text{PD}}, P)$ when we use the PD-distance as our metric. We assumed that A^* is unique. So the almost sure convergence of $\Phi(g_{\text{PD}}, \mathcal{C}_{A_n,\text{PD}}, P)$ to $\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)$ will confirm the almost sure convergence of A_n to the optimal κ-means centers A^* (with respect to the Hausdorff distance).

This means we only need to show that $\Phi(g_{\text{PD}}, \mathcal{C}_{A_n,\text{PD}}, P)$ converges to $\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)$ almost surely. We prove this by using several intermediate steps. In these steps we alternate between P and P_n, between g_{PD} and g_{sp}, and between $\mathcal{C}_{A,\text{PD}}$ and $\mathcal{C}_{A,sp}$. Also we need to reach A^* from A_n. This is done using an intermediate step that goes from vertices A_n to a set of vertices near to A^*. To this end we define $\tilde{A^*}$ as the nearest subset of samples to A^*:

$$\tilde{A^*} = \{\tilde{v}_j | \tilde{v}_j = \arg\min_{v \in V_n} \| v - A^*(j) \|; j = 1, ..., \kappa\}.$$

For $v_j \in A^*$, we also use the notation $\tilde{v}_j = \tilde{A^*}(v_j)$ to denote the corresponding vertex from V_n. Set $c_n = (k/\eta_d n)^{1/d}$ as before. We bound $\Delta\Phi = \Phi(g_{\text{PD}}, \mathcal{C}_{A_n,\text{PD}}, P) - \Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)$ as follows:

$$
\begin{aligned}
\Delta\Phi \leq\ & |\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P_n)| && \text{(7.4a)}\\
& + \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P_n) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,sp}, P_n) && \text{(7.4b)}\\
& + |\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,sp}, P_n) - c_n^2\Phi(g_{sp}, \mathcal{C}_{A_n,sp}, P_n)| && \text{(7.4c)}\\
& + c_n^2\big(\Phi(g_{sp}, \mathcal{C}_{A_n,sp}, P_n) - \Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},sp}, P_n)\big) && \text{(7.4d)}\\
& + c_n^2\big(\Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},sp}, P_n) - \Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n)\big) && \text{(7.4e)}\\
& + |c_n^2\Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n)| && \text{(7.4f)}\\
& + |\Phi(g_{\mathrm{PD}}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A^*,\mathrm{PD}}, P_n)| && \text{(7.4g)}\\
& + |\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A^*,\mathrm{PD}}, P_n) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A^*,\mathrm{PD}}, P)|. && \text{(7.4h)}
\end{aligned}
$$

Now we show that every term on the right hand side is either non-positive or almost surely converges to zero.
(7.4a) The proof of the uniform strong law of large numbers (SLLN) for $\Phi(g_{\|.\|}, \mathcal{C}_{\cdot,\|.\|}, P)$ (see, e.g., Pollard 1981) holds for all metric spaces with compact closed balls. This results in the uniform SLLN for $\Phi(g_{\mathrm{PD}}, \mathcal{C}_{\cdot,\mathrm{PD}}, P)$ (uniform over all set of κ-centers) and almost sure convergence of $\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P_n)$ to $\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P)$. So

$$|\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P_n)| \xrightarrow{a.s.} 0.$$

(7.4b) From the definition, $\mathcal{C}_{A,\mathrm{PD}}(x)$ is the centroid with shortest PD-distance to x, so

$$\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,\mathrm{PD}}, P_n) \leq \Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,sp}, P_n).$$

(7.4c) From Lemma 7.7,

$$\sup_{A_n} |\Phi(g_{\mathrm{PD}}, \mathcal{C}_{A_n,sp}, P_n) - c_n^2\Phi(g_{sp}, \mathcal{C}_{A_n,sp}, P_n)| \xrightarrow{a.s.} 0.$$

(7.4d) The centers A_n are optimal κ-medoids centers with respect to the shortest path distance, so

$$\Phi(g_{sp}, \mathcal{C}_{A_n,sp}, P_n) - \Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},sp}, P_n) \leq 0.$$

(7.4e) $\mathcal{C}_{A,sp}(x)$ is defined as the centroid with graph shortest path distance to x, so

$$\Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},sp}, P_n) \leq \Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n).$$

(7.4f) From Lemma 7.7,

$$\sup_{\tilde{A^*}} |c_n^2\Phi(g_{sp}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n) - \Phi(g_{\mathrm{PD}}, \mathcal{C}_{\tilde{A^*},\mathrm{PD}}, P_n)| \xrightarrow{a.s.} 0.$$

(7.4g) By the construction of $\tilde{A^*}$, the distance between $\tilde{A^*}$ and A^* decreases as the sample size n increases. Define $\delta = \max_{v\in A^*} D_{\mathrm{PD}}(\tilde{A^*}(v), v)$. It is easy to check that δ almost surely converges to zero as $n \to \infty$. Now we can use the squared triangle inequality to get

$$\begin{aligned}
\Phi(g_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n) &= \frac{1}{n}\sum D_{\text{PD}}(\mathcal{C}_{\tilde{A}^*,\text{PD}}(X_i), X_i)^2 \\
&\leq \frac{1}{n}\sum D_{\text{PD}}(\tilde{A}^*(\mathcal{C}_{A^*,\text{PD}}(X_i)), X_i)^2 \\
&\leq \frac{1}{n}\sum D_{\text{PD}}(\mathcal{C}_{A^*,\text{PD}}(X_i), X_i)^2 \\
&\quad + \frac{1}{n}\sum D_{\text{PD}}(\mathcal{C}_{A^*,\text{PD}}(X_i), \tilde{A}^*(\mathcal{C}_{A^*,\text{PD}}(X_i)))^2 \\
&\quad + \frac{2}{n}\sum D_{\text{PD}}(\mathcal{C}_{A^*,\text{PD}}(X_i), X_i) D_{\text{PD}}(\mathcal{C}_{A^*,\text{PD}}(X_i), \tilde{A}^*(\mathcal{C}_{A^*,\text{PD}}(X_i))) \\
&\leq \Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n) + \delta^2 + 2\delta\Phi(D_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n).
\end{aligned}$$

Similarly, we get

$$\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n) \leq \Phi(g_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n) + \delta^2 + 2\delta\Phi(D_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n).$$

Note that $\Phi(D_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)$ and $\Phi(D_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n)$ almost surely converge to $\Phi(D_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)$ and $\Phi(D_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P)$, which are bounded. Here,

$$|\Phi(g_f, \mathcal{C}_{\tilde{A}^*,f}, P_n) - \Phi(g_f, \mathcal{C}_{A^*,f}, P_n)| \xrightarrow{a.s.} 0.$$

(7.4h) By the strong law of large number for $\Phi(g_{\text{PD}}, \mathcal{C}_{\cdot,\text{PD}}, P)$ we have

$$|\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n) - \Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)| \xrightarrow{a.s.} 0.$$

So all in all we get $|\Phi(g_{\text{PD}}, \mathcal{C}_{A_n,\text{PD}}, P) - \Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P)| \xrightarrow{a.s.} 0$ which finishes the proof. □

Convergence for $\kappa \to \infty$

So far we only proved the convergence results for fixed κ. However,Theorem 7.4 holds for $\kappa \to \infty$. As we work with empirical centers, it is necessary that κ, k and n go to infinity together. The next theorem specifies the interplay between these parameters. It shows that if all parameters go to infinity at appropriate speed, the empirical κ-medoids centers will converge to optimal κ centers and will resemble the original density.

Theorem 7.8 (κ-medoids centers resemble the original density)
Consider the unweighted kNN graph G_n based on the i.i.d. sample $X_1, ..., X_n \in \mathcal{X}$ drawn from the density p. Assume that for all $x \in \mathcal{X}$ the density at x is bounded

$$0 < p_{\min} \leq p(x) \leq p_{\max} < \infty,$$

and let $\alpha > 0$ be a constant. We choose κ, k and n such that $k \geq \log(n)^{1+\alpha}$, $k\log(n)/n \to 0$ and $\kappa k/n \to 0$. Let A_n denote the optimal empirical κ-medoids in graph G_n with respect to the shortest path distance. Assume that the set of optimal κ-means centers with respect to the PD*-distance is unique. Then as κ, k and n tend to infinity, the empirical κ-medoids centers A_n follow the density p.*

The conditions on k and κ have intuitive interpretations. The condition on k is a bit stronger than the usual condition $k/\log(n) \to \infty$ that guarantees connectivity in random geometric graphs. The condition on κ also has an intuitive meaning. If we choose a large κ, say $\kappa \approx n/\log(n)$, each center would only have around $\log(n)$ points in its own cell. Thus each center is connected to almost all points inside the cell with a path of length 1. Therefore the shortest paths inside the cells are not good approximations for PD-distances. The condition $\kappa k/n \to 0$ ensures that for each center, many of the points inside the cell has shortest path distance $\omega(1)$. As a rule of thumb, we can set $k \approx \log(n)$ and choose κ smaller than $n/\log(n)^2$.

The proof of this theorem needs a more careful investigation of Equation (7.4). We have to show that each term on the right-hand side converges to zero if κ, k and n go to infinity at the specified speed. For terms (7.4a) and (7.4h), this can be done using standard results from the literature (Pollard, 1982, Bartlett et al., 1998). However, the terms (7.4c) and (7.4f) only appear in our algorithm and need a separate analysis. Details can be found in Section 7.7.1.

7.5 Downsampling geometric graphs

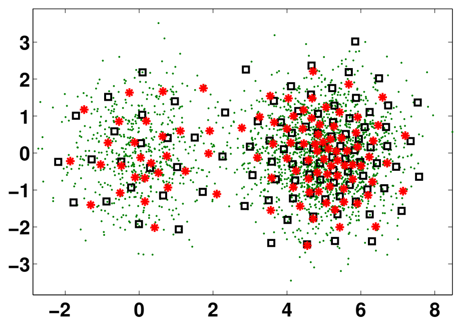
In this section, we use the proposed algorithm to downsample random geometric graphs. Assume that we are given a massive unweighted kNN graph G with n vertices. The vertices of the graph are sampled from a probability density p, but we neither have access to the point locations of vertices, nor to the underlying density. Our task is to "downsample" G to a much smaller graph G' with $n' \ll n$ vertices. The downsampled graph G' should "look like" a kNN graph built on n' samples from the density p.

A simple idea is to randomly select n' vertices. Then connect each vertex to its k' nearest neighbors, where distances are measured by the shortest path distances in the original graph G. If we choose k' in the right range ($k' \geq c \log(n')$ for a sufficiently large c), we end up with a connected kNN graph. Can we benefit from the large geometric graph G to find a "better" solution?

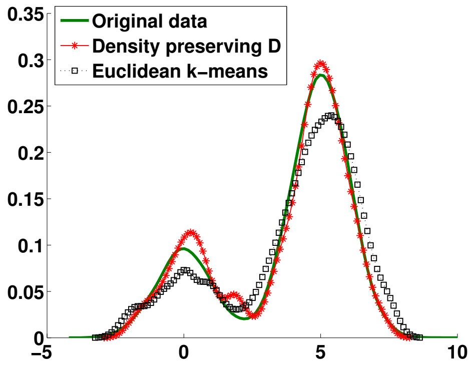
We motivate our "better" solution by an example. Let u denote the uniform density on the unit square. We want to represent u by a subset of m^2 points from $[0,1]^2$. Consider two solutions for this problem: select m^2 random samples from the density u or select m^2 points on a grid of width $\Theta(1/m)$. One can show that the grid solution is a better representation of the density u in the following sense: For the first solution, there exist a circle inside $[0,1]^2$ with radius $\Theta(\log(m)/m)$ such that it does not contain any sample point with high probability. For the grid solution, the largest radius of such a circle is $\Theta(1/m)$. Moreover, if we build a neighborhood graph on our grid points, the graph will be connected as soon as we connect every point to its 4 nearest neighbors. However in the random sample, the graph would be connected only if we connect every point to at least $\log(m)$ of its nearest neighbors.

We claim that our density-preserving downsampling algorithm results in samples that are more evenly spaced than a random sample. More specifically, if we connect each center to a constant number of its nearest neighbors, the graph would be connected. This constant depends on the geometry and the dimension of our underlying space, but not on the number of centers. The next theorem states this for optimal κ-means centers in the continuous case.

Theorem 7.9 (Sparse neighborhood graphs on optimal centers) *Let $\mathcal{X}$ be a connected and compact subset of $\mathbb{R}^d$ endowed with a Lipschitz continuous density p. Assume that for all $x \in \mathcal{X}$ we have $0 < p_{\min} \leq p(x) \leq p_{\max} < \infty$. Let $A^{*\kappa}$ be the set of optimal κ-means centroids with respect to the PD-distance that attains the minimum of $\Phi(g_{\text{PD}}, \mathcal{C}_{A,\text{PD}}, P)$. Then there exists a constant c such that the c nearest neighbor graph built on centroids is connected.*



(a)



(b)

Figure 7.1: a) Original data (green) and centroids based on the shortest path distance (red) and the Euclidean distance (black).
b) The marginal distribution of samples and quantization centroids in the direction of x-axis.

The constant c is related to the doubling constant of the underlying space $\mathcal{X}$, which depends on d but not on κ.

This theorem is in sharp contrast with the fact that for connectivity of a kNN graph, it is necessary that $k \geq C \log(n)$ (see, e.g., Penrose, 1999). The proof uses a geometric argument based on the *Delone property* of Voronoi cells corresponding to optimal κ-means centers. It also shows how we can find the constant c: Look at the Voronoi diagram of the optimal centers and set c as the maximum number of facets of a Voronoi cell in that diagram. Details of the proof are discussed in Section 7.7.2.

Now we can put Theorems 7.8 and 7.9 together to get our desired result. Namely, if we apply the density-preserving downsampling on unweighted kNN-graphs, the centers will follow the underlying density of the original graph. Moreover, we can build a sparse neighbourhood graph on these κ downsampled vertices. The total number of edges in this graph will depend linearly on κ. The original graph G can also be a ε-graph or a weighted kNN graph. The same downsampling algorithm is applicable on ε-graphs by finding a kNN subgraph of G.

7.6 Implementation and simulation

We now suggest a heuristic to simplify the implementation of the quantization procedure when our points are embedded in an Euclidean space. Then we discuss the case where we do not have access to the embedding of our sample points, but only to the adjacency matrix of the kNN graph. Finally, we illustrate an example of density preserving quantization with shortest path distances on a synthetic dataset as a proof of concept.

The classical implementation of κ-means is an EM type algorithm by Lloyd (1982). After initializing the centers, it iterates over two steps:

1. The assignment step: It is referred also as the expectation step. In this step, data points are assigned to centers.

2. The update or maximization step: Recalculate the centers using the result from Step 1.

It is easy to adapt the assignment step when we use the shortest path distance. The centers are not necessarily vertices from our graph, so we connect each center to its nearest sample point. Every point will be assigned to the center with smallest shortest path distance. For the update step, instead of finding the exact center of each Voronoi cell with respect to the shortest path distance, we use a simple heuristic. We assume that the density inside each cell is constant, in which case the update would be the same as for the standard Euclidean κ-means. If there are c_i points in the Voronoi cell i, this reduces the computational complexity of finding the center from $O(c_i^2)$ to $O(c_i)$. This heuristic is useful when c_i is relatively large (e.g. $c_i = O(\sqrt{n})$) and applicable when the density of points inside each cell is close to uniform.

As a proof of concept, we apply this algorithm on a synthetic dataset. We draw 12,000 points from an unbalanced mixture of two Gaussian distributions in $\mathbb{R}^2$ and build an unweighted kNN graph with $k = 15$. Then we quantize to $\kappa = 100$ centers using the shortest path distance and Euclidean distance. The results are depicted in Figure 7.1a. One can see that the Euclidean distance based quantization tends to have more centers in low density regions compared to the shortest path distances based centers. In the former, centers resemble the density $p(x)^{1/2}$ (Graf and Luschgy, 2000), hence low density regions are amplified. To compare the density of the centers of standard k-means and our algorithm, we plot their estimated marginal density in the direction of x-axis in Figure 7.1b. The density bias of quantization centers with respect to the Euclidean distance is again apparent in this figure.

Computing κ-medoids or κ-means with respect to the shortest path distance is used in the literature for clustering (Kim et al., 2007, Feil and Janos, 2007). Asgharbeygi and Maleki (2008) propose a relatively simple algorithm for finding centers when there is no access to the embedding of the data. They first show that for proper distance matrices, one can apply multidimensional scaling (MDS) to find a distance preserving embedding of the points in a higher dimensional Euclidean space and use the Lloyd algorithm in the embedded space. At the end, they show how to do all the computations in the original space without paying the cost of the embedding step.

7.7 Proofs

7.7.1 Proof of Theorem 7.8

The first step of the proof is similar to the proof of Theorem 7.5, up to Equation (7.4). We show that each term on the right-hand side of that equation converges to zero if k, κ and n go to infinity at specified rates.

(7.4a) Let $\mathcal{F}$ denote the class of functions

$$\mathcal{F}_\kappa = \{f_A(x) | f_A(x) = D_{\text{PD}}(x, \mathcal{C}_A(x))^2, A \subset \mathcal{X}, |A| = \kappa\},$$

and $\mathcal{N}(t, \mathcal{F}_\kappa)$ the covering number of $\mathcal{F}_\kappa$ with respect to the supremum norm. To cover $\mathcal{X}$ with PD-balls of radius t, we need $\Theta(t^{-d})$ of such balls (see Lemma 2.7 in Chapter 2). This means that $\mathcal{N}(t, \mathcal{F}_\kappa) = \Theta(t^{-\kappa d})$. We also have

$$P(\sup_A \left|\Phi(g_{\text{PD}}, \mathcal{C}_{A,\text{PD}}, P) - \Phi(g_{\text{PD}}, \mathcal{C}_{A,\text{PD}}, P_n)\right| > t) \le \mathcal{N}(t, \mathcal{F}_\kappa) e^{-nt^2/B},$$

where B is a constant depending on the diameter of $\mathcal{X}$. This shows that the difference converges to zero with high probability (probability converging to 1) if $\kappa \log(n)/n \to 0$.

(7.4c) We revisit Lemma 7.7 for $\kappa \to \infty$. We rewrite the proof of the lemma for a general $\mathcal{C}_A$:

$$\begin{aligned}\frac{1}{(1-\lambda)^{2/d}}\Phi(g_{\text{PD}}, \mathcal{C}_A, P_n) &\leq c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) \\ &\leq e^2 \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n) + c_n^2 + 2c_n e \Phi(D_{\text{PD}}, \mathcal{C}_A, P_n).\end{aligned}$$

Set $\mathcal{C}_A = \mathcal{C}_{A,sp}$. We need to show that as κ, k and n go to infinity,

$$\frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)}{c_n^2} \to \infty \quad \text{and} \quad \frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)}{c_n e \Phi(D_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)} \to \infty.$$

From Theorem 1 in Gruber (2004), we know that

$$\Phi(g_{\text{PD}}, \mathcal{C}_{A,sp}, P_n) = \Theta(\frac{1}{\kappa^{2/d}}) \quad \text{and} \quad \Phi(D_{\text{PD}}, \mathcal{C}_{A,sp}, P_n) = \Theta(frac1\kappa^{1/d}).$$

Recalling $c_n = (\frac{k}{n\eta_d})^{1/d}$, we get

$$\frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)}{c_n^2} == \Theta\big((\frac{\eta_d n}{\kappa k})^{2/d}\big) \quad \text{and} \quad \frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)}{c_n e \Phi(D_{\text{PD}}, \mathcal{C}_{A,sp}, P_n)} = \Theta\big((\frac{\eta_d n}{\kappa k})^{1/d}\big).$$

This shows that $|c_n^2 \Phi(g_{sp}, \mathcal{C}_A, P_n) - \Phi(g_{\text{PD}}, \mathcal{C}_A, P_n)|$ almost surely converges to zero if $\kappa k/n \to 0$.

(7.4f) Similar to Part (7.4c).

(7.4g) Consider the proof of the corresponding part in Theorem 7.5. We need to show that

$$\frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)}{\delta^2} \to \infty \quad \text{and} \quad \frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)}{\delta \Phi(D_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n)} \to \infty.$$

Instead of bounding δ, it is easier to bound $\delta' = \sup_{x \in \mathcal{X}, v \in V_n} \|x - v\|$. Using a standard sphere packing lemma, we know that $\delta' \leq (k/n)^{1/d}$ with probability converging to 1. For large enough n, we have

$$\delta = \max_{v \in A^*} D_{\text{PD}}(\tilde{A}^*(v), v) \leq 2p_{\max}^{1/d} \max_{v \in A^*} \|\tilde{A}^*(v) - v\| \leq 2p_{\max}^{1/d}\delta'.$$

This means that, with high probability, $\delta \leq 2p_{\max}^{1/d}(k/n)^{1/d}$. Similar to Part (7.4c), we have

$$\frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)}{\delta^2} = \Theta\big((\frac{n}{k\kappa})^{2/d}\big) \quad \text{and} \quad \frac{\Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)}{\delta \Phi(D_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n)} = \Theta\big((\frac{n}{k\kappa})^{1/d}\big).$$

This shows that $|\Phi(g_{\text{PD}}, \mathcal{C}_{\tilde{A}^*,\text{PD}}, P_n) - \Phi(g_{\text{PD}}, \mathcal{C}_{A^*,\text{PD}}, P_n)|$ almost surely converges to zero if $\kappa k/n \to 0$.

(7.4h) Similar to Equation (7.4a), this term converges to zero if $\kappa \log(n)/n \to 0$. □

7.7.2 Proof of Theorem 7.9

Consider the Voronoi diagram induced by the optimal centers. Connect each center to centers of all neighbor cells to attain a connected graph. Let c be the maximum degree in this graph.

We extend this graph into a nearest neighbor graph with neighborhood size c. Consider a vertex v_i with degree d_i. Extend the neighborhood of v_i by connecting it to its next $c - d_i$-nearest neighbors (with respect to the PD-distance). We show that c is a constant independent of κ, which proves the theorem.

The Voronoi cells corresponding to optimal centers $A^{*\kappa}$ cannot be very thin or very long (Gruber, 2001). This property and a sphere packing lemma are used to bound the number of neighbors of each Voronoi cell. We first mention the proposition from Gruber (2004) on the shape of optimal Voronoi cells.

Proposition 7.10 (Optimal Voronoi cells are delone) *Let $\mathcal{X}$ be a compact and smooth Riemannian d-manifold. Let $A^{*\kappa}$ be the optimal quantization centroids with respect to the Riemannian metric ϱ, that attains the minimum of $\Phi(g_\sigma, \mathcal{C}_A, P)$. Then there exist constants $a, b > 0$ such that $A^{*\kappa}$ is $(a\kappa^{1/d}, b\kappa^{1/d})$-Delone. This means that*

- *Every two distinct centers of $A^{*\kappa}$ have distance at least $a\kappa^{1/d}$.*
- *For each point of $\mathcal{X}$, there exists a center in $A^{*\kappa}$ at distance at most $b\kappa^{1/d}$.*

Constants a and b depend on d and the geometry of $\mathcal{X}$, but not on κ.

Note that this proposition is not asymptotic and holds for every n. Let $B_\varrho(x, r)$ denote the closed ϱ-ball with radius r and center x

$$B_\varrho(x, r) = \{y | \varrho(x, y) \leq r\}.$$

Also denote the Voronoi cell of an optimal center $a_i \in A^{*\kappa}$ by D_i. The next lemma provides the necessary tools for our proof.

Lemma 7.11 *Consider the setting in Proposition 7.10 and let D_i and D_j be two neighboring Voronoi cells.*

- *The neighbor centers are not far from each other: $\varrho(a_i, a_j) \leq 2b\kappa^{1/d}$.*
- *The cell D_j is inside the ϱ-ball around a_i with radius $3b\kappa^{1/d}$: $D_j \subset B_\varrho(a_j, 3b\kappa^{1/d})$.*
- *Voronoi cells are fat: The ϱ-ball with radius $0.5a\kappa^{1/d}$ around a_i is completely inside D_i:*

$$B_\varrho(a_i, 0.5a\kappa^{1/d}) \subset D_i.$$

Proof. *Part 1.* Consider the geodesic path between a_i and a_j. Let m be the mid point of this path: $\varrho(a_i, m) = \varrho(a_i, a_j)/2$. Point m is on the boundary of D_i and D_j, so $m \in D_i$. We show that $\varrho(a_i, m) \leq b\kappa^{1/d}$. If not, from Part 2 of Proposition 7.10, there exist a center a_k such that $\varrho(a_k, m) \leq b\kappa^{1/d}$. This means that $\varrho(a_k, m) < \varrho(a_i, m)$, which contradicts the fact that m is in D_i.

Part 2. Consider a point $x \in D_j$. Similar to Part 1, we can show that $\varrho(a_j, x) \leq b\kappa^{1/d}$. Using the triangle inequality, we have

$$\varrho(a_i, x) \leq \varrho(a_i, a_j) + \varrho(a_j, x) \leq 3b\kappa^{1/d}.$$

Part 3. If not, there exists a point x such that $\varrho(a_i, x) \leq 0.5a\kappa^{1/d}$ but $x \notin D_i$. Therefore, the point x is inside a cell D_l with center a_l such that $\varrho(a_l, x) < \varrho(a_i, x)$. This means that

$$\varrho(a_i, a_l) \leq \varrho(a_i, x) + \varrho(x, a_l) < a\kappa^{1/d},$$

which is in contradiction with Part 1 of Proposition 7.10. □

Consider an optimal center a_i and denote the set of centers of all neighboring cells by A_i. The PD-balls with radius $0.5a\kappa^{1/d}$ around centers in A_i are all disjoint. These balls are also completely inside $B_{\text{PD}}(a_i, 3b\kappa^{1/d})$, thus

$$p\big(B_{\text{PD}}(a_i, 3b\kappa^{1/d})\big) \geq \sum_{v \in A_i} p\big(B_{\text{PD}}(v, 0.5a\kappa^{1/d})\big) \geq |A_i| p_{\min} e_1 \kappa,$$

where e_1 is a constant (depending on d). Also

$$p\big(B_{\text{PD}}(a_i, 3b\kappa^{1/d})\big) \leq e_2 p_{\max} \kappa,$$

for a constant e_2. All in all, we have $|A_i| \leq \frac{p_{\max} e_2}{p_{\min} e_1}$ which is a constant independent of κ. □

Bibliography

L. A. Adamic and E. Adar. Friends and Neighbors on the Web. *Social Networks*, 25:211–230, 2003.

M. Alamgir and U. von Luxburg. Multi-agent random walks for local clustering. In *International Conference on Data Minig (ICDM)*, 2010.

M. Alamgir and U. von Luxburg. Phase transition in the familiy of p-resistances. In *Neural Information Processing Systems (NIPS)*, 2011.

M. Alamgir and U. von Luxburg. Shortest path distance in random k-nearest neighbor graphs. In *International Conference on Machine Learning (ICML)*, 2012.

N. Alon, C. Avin, M. Koucky, G. Kozma, Z. Lotker, and M. R. Tuttle. Many random walks are faster than one. In *Symposium on Parallelism in Algorithms and Architectures*, pages 119–128, 2008.

R. Andersen and Y. Peres. Finding sparse cuts locally using evolving sets. In *ACM Symposium on Theory of Computing (STOC)*, pages 235–244, 2009.

R. Andersen, F. R. K. Chung, and Kevin J. L. Local graph partitioning using pagerank vectors. In *Foundations of Computer Science (FOCS)*, pages 475–486, 2006.

N. Asgharbeygi and A. Maleki. Geodesic k-means clustering. In *International Conference on Pattern Recognition (ICPR)*, 2008.

L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *International Conference on Web Search and Data Mining (WSDM)*, 2011.

A. L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, pages 509–512, 1999.

P.L. Bartlett, T. Linder, and G. Lugosi. The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory*, 44(5), 1998.

F. Bavaud and G. Guex. Interpolating between random walks and shortest paths: A path functional approach. In *Proceedings of the 4th International Conference on Social Informatics*, pages 68–81, 2012.

M. Bazaraa, J. Jarvis, and H. Sherali. *Linear Programming and Network Flows.* Wiley-Interscience, 2010.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. *Journal of Machine Learning Research (JMLR)*, 7:2399–2434, 2006.

A. Benczúr and D. Karger. Approximating s-t Minimum Cuts in O(n^2) time. In *ACM Symposium on Theory of Computing (STOC)*, pages 47–55, 1996.

A.L. Besse. *Einstein Manifolds.* Classics in Mathematics. Springer, 1987.

A. Bijral, N. Ratliff, and N. Srebro. Semi-supervised Learning with Density Based Distances. In *Uncertainty in Artificial Intelligence (UAI)*, 2011.

B. Bollobas. *Modern Graph Theory.* Springer, 1998.

O Bousquet, O Chapelle, and M Hein. Measure Based Regularization. In *Neural Information Processing Systems (NIPS)*. 2004.

T. Bühler and M. Hein. Spectral clustering based on the graph p-Laplacian. In *International Conference on Machine Learning (ICML)*, pages 81–88, 2009.

P. Burai and Á. Száz. Relationships between homogeneity, subadditivity and convexity properties. *Publikacija Elektrotehničkog Fakulteta - Serija: Matematika*, (16):77 – 87, 2005.

A. Chandra, P. Raghavan, W. Ruzzo, R. Smolensky, and P. Tiwari. The Electrical Resistance of a Graph Captures its Commute and Cover Times. In *Symposium on Theory of Computing (STOC)*, pages 574–586, 1989.

P. Chebotarev. A class of graph-geodetic distances generalizing the shortets path and the resistance distances. *Discrete Applied Mathematics*, 159(295–302), 2011.

G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

V. de Silva and J. B. Tenenbaum. Global Versus Local Methods in Nonlinear Dimensionality Reduction. In *Neural Information Processing Systems (NIPS)*, 2002.

V. de Silva and J.B. Tenenbaum. Sparse multidimensional scaling using landmark points. Technical report, Stanford University, 2004.

E.J. Delp and O.R. Mitchell. Moment preserving quantization. *IEEE Transactions on Communications*, 39(11):1549–1558, 1991.

P. Diaconis and D. Stroock. Geometric bounds for eigenvalues of Markov chains. *The Annals of Applied Probability*, pages 36–61, 1991.

P. G. Doyle and J. L. Snell. Random walks and electric networks, 1984.

B. Feil and A. Janos. Geodesic distance based fuzzy clustering. *Lecture Notes in Computer Science, Soft Computing in Industrial Applications*, pages 50–59, 2007.

G. Fejes Tóth. A stability criterion to the moment theorem. *Studia Scientiarum Mathematicarum Hungarica*, 38(1):209–224, 05 2001.

C. Gallesco, S. Müller, and S. Popov. A note on spider walks. *ArXiv e-prints*, October 2009.

C. Gallesco, S. Muller, S. Popov, and M. Vachkovskaia. Spiders in random environment. *ArXiv e-prints*, January 2010.

T. W. Gamelin. *Complex Analysis.* Springer-Verlag, New York, Inc., 2007.

A. Gersho. Asymptotically optimal block quantization. *IEEE Transactions on Information Theory*, 25(4):373–380, 1979.

A. V. Goldberg and C. Harrelson. Computing the shortest path: A search meets graph theory. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005.

G. Golub and C. Van Loan. *Matrix computations.* Johns Hopkins University Press, Baltimore, 1996.

M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the Temporal Dynamics of Diffusion Networks. In *International Conference on Machine Learning*, (ICML), 2011.

L. Grady. Random Walks for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1768–1783, 2006.

S. Graf and H. Luschgy. *Foundations of quantization for probability distributions.* Lecture notes in mathematics. Springer-Verlag New York, Inc., 2000.

P. M. Gruber. Optimal configurations of finite sets in Riemannian 2-manifolds. *Geometriae Dedicata*, 84(1-3):271–320, 2001.

P. M. Gruber. Optimum quantization and its applications. *Advances in Mathematics*, 186(2): 456 – 497, 2004.

P. Heckbert. Color image quantization for frame buffer display. In *Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 1982.

A. Hegde, D. Erdogmus, T. Lehn-Schioler, Y. Rao, and J. Principe. Vector-Quantization by density matching in the minimum Kullback-Leibler divergence sense. In *IEEE International Conference on Neural Networks*, 2004.

M. Hein, J.-Y. Audibert, and U. von Luxburg. Graph Laplacians and their convergence on random neighborhood graphs. *Journal of Machine Learning Research (JMLR)*, 8:1325 – 1370, 2007.

M. Herbster and G. Lever. Predicting the Labelling of a Graph via Minimum p-Seminorm Interpolation. In *Conference on Learning Theory (COLT)*, 2009.

J. J Hull. A Database for Handwritten Text Recognition Research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

M.M. Van Hulle. Faithful representations with topographic maps. *Neural Networks*, 12(6): 803 – 823, 1999.

S. J. Hwang, S. B. Damelin, and A. O. Hero. Shortest Path through Random Points. January 2012. URL `http://arxiv.org/abs/1202.0045`.

G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1999.

L Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18:39–43, 1953.

J. Kim, K. Shim, and S. Choi. Soft geodesic kernel k-means. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 429–432, 2007.

I. Kivimäki, M. Shimbo, and M. Saerens. Developments in the theory of randomized shortest paths with a comparison of graph node distances. *Physica A: Statistical Mechanics and its Applications*, 393:600 – 616, 2014.

J. Kleinberg, A. Slivkins, and T. Wexler. Triangulation and embedding using small sets of beacons. In *Foundations of Computer Science (FOCS)*, 2004.

E. Kolaczyk. *Statistical Analysis of Network Data*. Springer, 2009.

R. I. Kondor and J. D. Lafferty. Diffusion Kernels on Graphs and Other Discrete Input Spaces. In *International Conference on Machine Learning*, (ICML), 2002.

V. Krishnamurthy, M/ Faloutsos, M. Chrobak, L. Lao, J H. Cui, and A. G. Percus. Reducing large internet topologies for faster simulations. In *International IFIP-TC6 Networking Conference*, 2005.

J. Lafferty and G. Lebanon. Diffusion Kernels on Statistical Manifolds. *Journal of Machine Learning Research (JMLR)*, 6:129–163, 2005.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

F. T. Leighton and A. Moitra. Extensions and limits to vertex sparsification. In *ACM Symposium on Theory of Computing (STOC)*, pages 47–56, 2010.

J. Leskovec and C. Faloutsos. Sampling from large graphs. In *Conference on Knowledge Discovery and Data Mining (KDD)*, 2006.

J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 2007.

J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Statistical properties of community structure in large social and information networks. In *International Conference on World Wide Web (WWW)*, pages 695–704, 2008.

T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43:29–44, 2001.

M. Li, J. Klejsa, and W. Bastiaan Kleijn. On distribution preserving quantization. *CoRR*, abs/1108.3728, 2011.

D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *International Conference on Information and Knowledge Management (CIKM)*, pages 556–559, 2003.

W. Liu, J. He, and S-F. Chang. Large Graph Construction for Scalable Semi-Supervised Learning. In *International Conference of Machine Learning (ICML)*, 2010.

S. Lloyd. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.

L. Lü, C. H. Jin, and T. Zhou. Similarity index based on local paths for link prediction of complex networks. *Physical Review E*, 80(4):46122, 2009.

M. McPherson, S. L. Lynn, and M. C. James. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

P. Meinicke and H. Ritter. Quantizing density estimators. In *Neural Information Processing Systems (NIPS)*, 2001.

A. K. Menon and Ch. Elkan. Link prediction via matrix factorization. In *European Conference on Machine learning and Knowledge Discovery in Databases(ECML PKDD)*, 2011.

K. Miller, T. Griffiths, and M. I. Jordan. Nonparametric latent feature models for link prediction. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

B. Nadler, N. Srebro, and X. Zhou. Semi-supervised learning with the graph Laplacian: The limit of infinite unlabelled data. In *Neural Information Processing Systems (NIPS)*, 2009.

J. Pan, H. Yang, C. Faloutsos, and P. Duygulu. Automatic multimedia cross-modal correlation discovery. In *Conference on Knowledge Discovery and Data mining (KDD)*, pages 653–658, 2004.

M. Penrose. A strong law for the longest edge of the minimal spanning tree. *The Annals of Probability*, 27:246–260, 1999.

P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:629–639, 1990.

D. Pollard. Strong consistency of k-means clustering. *Annals of Statistics*, 9(1):135–140, 1981.

D. Pollard. A central limit theorem for k-means clustering. *The Annals of Probability*, (4), 1982.

D. Rafiei and S. Curial. Effectively Visualizing Large Networks Through Sampling. In *IEEE Visualization Conference*, 2005.

N. Ruan and R. Jin. Distance preserving graph simplification. *International Conference on Data Mining (ICDM)*, 2011.

Sajama and A. Orlitsky. Estimating and computing density based distance metrics. In *International Conference on Machine Learning (ICML)*, 2005.

P. Sarkar, Ch. Deepayan, and M. Jordan. Nonparametric Link Prediction in Dynamic Networks. In *International Conference on Machine Learning*, (ICML), 2012.

V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *Conference on Knowledge Discovery and Data Mining (KDD)*, 2009.

J. Silva, J. S. Marques, and J. M. Lemos. Selecting Landmark Points for Sparse Manifold Learning. In *Neural Information Processing Systems (NIPS)*, 2005.

D. Spielman and N. Srivastava. Graph sparsification by effective resistances. In R Ladner and C Dwork, editors, *Proceedings of the 40th Annual Symposium on Theory of Computing (STOC)*, pages 563–568, 2008.

D. A. Spielman and S.-H. Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *CoRR*, abs/0809.3232, 2008.

J. Tenenbaum, V. de Silva, and J. Langford. Supplementary material to "A Global Geometric Framework for Nonlinear Dimensionality Reduction". *Science*, 2000.

V. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395 – 416, 2007.

U. von Luxburg, M. Hein, and A. Radl. Hitting times, commute distances and the spectral gap in large random geometric graphs. In *Preprint available at Arxiv*, March 2010. URL `http://arxiv.org/abs/1003.1266`.

D. Yan, L. Huang, and M. I. Jordan. Fast approximate spectral clustering. In *International Conference on Knowledge Discovery and Data Mining(SIGKDD)*, pages 907–916, 2009.

S. Yang, B. Long, A. Smola, N. Sadagopan, Zh. Zheng, and H. Zha. Like like alike: joint friendship and interest propagation in social networks. In *International conference on World Wide Web*, WWW, 2011.

L. Yen, M. Saerens, A. Mantrach, and M. Shimbo. A family of dissimilarity measures between nodes generalizing both the shortest-path and the commute-time distances. In *International Conference on Knowledge Discovery and Data Mining(SIGKDD)*, pages 785–793, 2008.

K. Zhang, J. T Kwok, and B. Parvin. Prototype vector machine for large scale semi-supervised learning. In *International Conference of Machine Learning (ICML)*, 2009.

D. Zhou and B. Schölkopf. Regularization on Discrete Spaces. In *DAGM-Symposium*, pages 361–368, 2005.

T. Zhou, L. Lu, and Y. Zhang. Predicting Missing Links via Local Information. *European Physical Journal B*, 71:623–630, 2009.

X. Zhou and M. Belkin. Semi-supervised Learning by Higher Order Regularization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

X. Zhu. *Semi-supervised learning with graphs.* PhD thesis, Pittsburgh, PA, USA, 2005.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *International Conference of Machine Learning (ICML)*, 2003.

Eidesstattliche Versicherung