UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS

ECOLE DOCTORALE STIC

SCIENCES ET TECHNOLOGIES DE L'INFORMATION
ET DE LA COMMUNICATION

THÈSE

pour obtenir le titre de

Docteur en Sciences

de l'Université de Nice - Sophia Antipolis
Mention : Informatique

Présenté et soutenue par

Jithin Kazhuthuveettil Sreedharan

Sampling and Inference in Complex Networks

(Échantillonnage et Inférence dans Réseaux Complexes)

Thèse dirigé par Konstantin Avrachenkov

et préparée au sein du laboratoire INRIA, équipe Maestro

soutenue le 2 décembre 2016

Jury

Directeur :	Konstantin Avrachenkov	-	Inria, Sophia Antipolis, France
Rapporteurs :	Nelly Litvak	-	University of Twente, Enschede, The Netherlands
	Don Towsley	-	University of Massachusetts, Amherst, USA
Examinateurs :	Philippe Jacquet	-	Nokia Bell Labs, Paris-Saclay, France
	Alain Jean-Marie	-	Inria, Sophia Antipolis, France

Abstract

The recent emergence of large evolving networks, mainly due to the rise of Online Social Networks (OSNs), brought out the difficulty to gather a complete picture of a network and it opened up the development of new distributed techniques. Due to the constraints imposed by large networks, it is realistic to assume that only local information is available at each node: the list of neighboring nodes that the node connects to. Random walk based and diffusion techniques are notably suitable in this frame work. However, despite many recent studies, several open issues remain to be addressed for such algorithms. This thesis proposes some novel distributed algorithms for sampling, estimation and inference of network functions, and for approximating the spectrum of graph matrices.

The thesis begins by tackling the problem of sampling in spectral domain: the classical problem of finding the dominant eigenvalues and their corresponding eigenvectors of symmetric graph matrices such as adjacency or Laplacian of undirected graphs. By relating the spectrum to a Schrödinger-type differential equation, we develop a scalable technique called "complex power iterations", a variant of power iterations, which gives a simple interpretation of spectrum in terms of peaks at the eigenvalue points. Distributed implementation is then formed with diffusion over the graph and with gossiping algorithms. The relation of quantum random walks with our formulation leads us to a simple algorithm based on quantum computing. Higher order approximations and symplectic numerical integrators are also proposed to solve the underlying differential equation.

Next, we consider sampling and estimation of network functions (aggregate and average) using random walks on graph. In order to avoid the burn-in time of Markov chain sampling, we use the idea of regeneration at the renewal epochs when the random walk revisits a fixed node. This help us to develop an estimator for the aggregate function, which is non-asymptotically unbiased and can be implemented in a massively distributed way. We introduce the idea of a "super-node" as the anchoring node for the renewals, to tackle disconnected or "weakly-knit" graphs. We derive an approximation to the Bayesian posterior of the estimate and it provides a real-time assessment of estimation accuracy. As a cross between the deterministic iteration and Markov sampling, an estimator based on reinforcement learning is also developed making use of the idea of regeneration.

The final part of the thesis deals with the use of extreme value theory to make inference from the stationary samples of the random walks. Extremal events like the first hitting time of a large degree node, order statistics and mean cluster size are well captured in the parameter "extremal index" from extreme value theory. We theoretically study and estimate the extremal indices of different random walk sampling techniques.

The techniques and tools developed in this thesis are tested on real-world networks and show promising results.

Résumé

L'émergence récente de grands réseaux, surtout les réseaux sociaux en ligne, a révélé la difficulté de crawler le réseau complet et a déclenché le développement de nouvelles techniques distribuées. Dans cette thèse, nous concevons et analysons des algorithmes basés sur les marches aléatoires et la diffusion pour l'échantillonnage, l'estimation et l'inférence des fonctions des réseaux.

La thèse commence par le problème classique de trouver les valeurs propres dominantes et leurs vecteurs propres de matrices de graphe symétriques, comme la matrice Laplacienne de graphes non orientés. En utilisant le fait que le spectre est associé à une équation différentiel de type Schrödinger, nous développons des techniques évolutives à l'aide de la diffusion sur le graphe. Ensuite, nous considérons l'échantillonnage des fonctions de réseau (comme somme et moyenne) en utilisant les marches aléatoires sur le graphe. Afin d'éviter le temps « burn-in » de marche aléatoire, avec l'idée de régénération à un nœud fixe, nous développons un estimateur de la fonction de somme qui est asymptotiquement non-biaisé et dérivons une approximation à la postérieure Bayésienne. La dernière partie de la thèse étudie l'application de la théorie des valeurs extrêmes pour faire une inférence sur les événements extrêmes à partir des échantillons stationnaires des différentes marches aléatoires pour l'échantillonnage de réseau.

Acknowledgments

"If I have seen further, it is by standing on the shoulders of giants."
– Issac Newton

It is with immense gratitude that I acknowledge the support and the help of my guide Dr. Konstantin Avrachenkov (Kostia). He has been a tireless source of fresh ideas and provided me necessary guidance to recover whenever my steps faltered. I am also grateful to him for creating chances to meet my collaborators and to initiate the discussions.

I thank Inria-Bell Labs joint lab for funding my PhD and all the travels related to conferences.

Many thanks to Dr. Philippe Jacquet for discussions on various aspects of this thesis. His ideas and insights have gone into my results in Chapter 3. Philippe has been a great supporter and recommended me for my internship at NSF center CSoI in Purdue. My sincere thanks to Prof. Wojciech Szpankowski and Prof. Ananth Grama for hosting me at Purdue and for considering me as one of their students.

I express my thanks to the jury of this thesis: my supervisor, Prof. Nelly Litvak, Prof. Don Towsley; Dr. Philippe Jacquet and Dr. Alain Jean-Marie for their valuable time in reviewing my thesis.

I would also like to thank Prof. Bruno Ribeiro for our collaboration and for spending time with me on improving my presentation skills.

My team in Inria - Maestro - has been with me during these three years in France. I thank them wholeheartedly, especially, Philippe Nain for hosting me in the team, Laurie for all the administrative and travel support, Alain for being a wonderful leader of the team, Giovanni for taking me along to Rio and Sara for all the advices. I also thank Arun and Hlib for helping me in many ways and I am indebted to them beyond words. I also appreciate all the help I have received from rest of Maestro members.

I am grateful to my master thesis advisor in IISc Bangalore, Prof. Vinod Sharma for introducing me into research.

I owe to many people during my stay in Inria, in particular to Ashwin Rao for all the discussions and Ratnesh Kumar for the support I had during my initial days in France.

My parents, Suseela and Sreedharan, have always inspired me and gave me full freedom to choose the path I love. I owe my deepest gratitude to their unflagging support and care. The same appreciation applies to my parent-in-laws, Suchithra and Santhosh as well.

Finally I wish to dedicate this thesis to my greatest discovery during my PhD days - my wife Nivi. I learned from her one of the most important lesson in research - perseverance!

Contents

Frequently used Notation

Symbol	Description
G	Graph
(V, E)	Node set and edge set
$\|V\|, \|E\|$	Number of nodes, Number of edges
d_j	Degree of node j without including self loop
Δ	$\max\{d_1, \ldots, d_n\}$
$\boldsymbol{A}$	Adjacency matrix $[a_{ij}]$
$\boldsymbol{D}$	Diagonal matrix formed from $d_1, \ldots, d_n$
$\boldsymbol{L}$	Laplacian matrix, $\boldsymbol{D} - \boldsymbol{A}$
$\boldsymbol{P}$	Transition probability matrix of random walk
$\lambda_1^M, \ldots, \lambda_{\|V\|}^M$	Real eigenvalues of symmetric matrix M in descending order. $^{(i)}$
$\boldsymbol{u}_{\mathbf{1}}^M, \ldots, \boldsymbol{u}_{\|\boldsymbol{V}\|}^M$	Eigenvectors corresponding to $\lambda_1^M, \ldots, \lambda_n^M$. $^{(i)}$
$\mathbf{1}$	Column vector of ones. Dimension implied from the context.
$\boldsymbol{I}$	Identity matrix. Dimension implied from the context.
$\mathcal{N}_j$	Neighbor list of node j without including self loop
$\boldsymbol{a}(k)$	kth component of a column vector $\boldsymbol{a}$
$\\|\boldsymbol{a}\\|$	Euclidean norm of vector $\boldsymbol{a}$
$\boldsymbol{x}_\ell$	Approximation of $\exp(i\varepsilon\ell\mathbf{A})$ multiplied by $\boldsymbol{b}_0$
$f(x) \sim g(x)$	$f(x)/g(x) \to 1$ as $x \to a$, $x \in S$. $^{(ii)}$
$f(x) = o(g(x))$	$\lim_{x \to a} f(x)/g(x) = 0$. $^{(ii)}$
$f(x) = \mathcal{O}(g(x))$	$\exists\ \delta > 0$ and $K > 0$ such that $\|f(x)\| \leqslant K\|g(x)\|$ for $\|x - a\| < \delta$. $^{(ii)}$
$\mu(G)$	$\sum_{(u,v) \in E} g(u, v)$
$\nu(G)$	$\sum_{u \in V} g(u)$
i.i.d.	Independent and identically distributed
$\mathbb{I}\{\mathcal{A}\}$	Indicator function: 1 if the event $\mathcal{A}$ is true, 0 otherwise.

$^{(i)}$ The dependency on M will de dropped if it is clear from the context.

$^{(ii)}$ Functions $f(x)$ and $g(x)$ are defined on some set S and a is a limit point of S.

Chapter 1
Introduction

The last few years have witnessed a boom in the science of massive data processing. Large datasets, often dubbed *Big data* in the industry, have triggered the development of many software and mathematical tools. This also helped a closely related inter-disciplinary field called “network science”, which studies networks in general and includes technological networks (Internet, phone networks, etc.), social networks (collaboration networks, Online Social Networks, etc.), information networks (World Wide Web, citation networks, etc.), and biological networks (protein interaction networks, metabolic and neural networks, etc.). In particular, a specific class called “complex networks” is popular in network science these days. It is used to describe, informally, networks with the following characteristics:

- Large size
- Sparse topology
- Small average distance (also called small-world phenomenon)
- Many triangles
- Heavy tail degree distribution (also called scale-free phenomenon)

Many of the above listed properties can be observed in real-world networks: six degrees of separation with Milgram’s small world experiments surprised the world with how connected we are and shows evidence for the small-world phenomenon [Travers & Milgram 1969]. Such close connectivities are observed in the Internet (Autonomous Systems) as well.

The study of large networks faces many issues: collecting data from the underlying networks takes time and huge resources; for instance, data collection in real-world networks is severely limited by the restrictions on the Application Programming Interface (API) queries imposed by the provider. Even if the whole graph is collected, centralized processing with many matrix algorithms has large memory requirements and incurs long delays to observe any fruitful output. Therefore, researchers resort to distributed and decentralized algorithms. Since many network algorithms require exponential time to finish, randomized and approximation algorithms combined with distributed implementation look promising in network science.

This thesis deals with efficient ways to collect representative samples from a large network using probabilistic techniques and makes statistical inference about the network properties with these samples. The emphasis is on distributed strategies.

The following sections provide a short introduction to the problems addressed in this thesis.

1.1 Preliminary Definitions

Let $G = (V, E)$ be an undirected labeled network, where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. Unlike the usual definition of E where each edge is an unordered pair and only present once, to simplify our notation we consider that if $(u, v) \in E$ then $(v, u) \in E$.[1] With a slight abuse of notation, the total number of edges $|E|$ is redefined as $|E|/2$. Both edges and nodes can have labels.

We define two typical matrices which appear frequently in network analysis. First one is the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{|V| \times |V|}$ in which the individual entries are given by

$$a_{uv} = \begin{cases} 1, & \text{if } u \text{ is a neighbour of } v, \\ 0, & \text{otherwise.} \end{cases}$$

The focus here is on undirected graphs and hence $\boldsymbol{A}^\intercal = \boldsymbol{A}$. The matrix $\boldsymbol{A}$ is also called the unweighted adjacency matrix and one can also define a weighted version in which the weight 1 for an edge is replaced by any nonnegative weight such that $a_{uv} = a_{vu}$.

Another matrix which is found very common in many graph theoretic problems is the Laplacian matrix $\boldsymbol{L}$ (also known as combinatorial Laplacian or unnormalized Laplacian in literature) defined as $\boldsymbol{L} = [\ell_{i,j}] := \boldsymbol{D} - \boldsymbol{A}$. Here the matrix $\boldsymbol{D}$ is diagonal with elements equal to degrees of the nodes $\{d_v\}, v \in V$.

For any general symmetric graph matrix $\boldsymbol{M}$, due to symmetry the eigenvalues are real and can be ranked in decreasing order as $\lambda_1^{\boldsymbol{M}} \geqslant \lambda_2^{\boldsymbol{M}} \geqslant \ldots \geqslant \lambda_{|V|}^{\boldsymbol{M}}$. Let $\boldsymbol{u}_1^{\boldsymbol{M}}, \boldsymbol{u}_2^{\boldsymbol{M}}, \ldots, \boldsymbol{u}_{|V|}^{\boldsymbol{M}}$ denote the eigenvectors corresponding to $\lambda_1^{\boldsymbol{M}}, \ldots, \lambda_{|V|}^{\boldsymbol{M}}$.[2]

1.2 Sampling and Estimation in Networks

Consider a large network that is impossible to observe completely, i.e., the network is assumed unknown in the sense that the graph matrices like adjacency or Laplacian are not completely known beforehand. How does one answer, at least approximately, questions about global properties of the networks? Examples include: what proportion among the population in a city supports a given political party? What is the average age of users in online social networks like Friendster, Myspace or Facebook? What is the fraction of male-female connections against that of female-female connections in a given Online Social Network (OSN)? Is the OSN assortative or disassortative?

[1] For convenience, some of the later chapters will redefine the graph notation.

[2] We will drop the dependence on the matrix in the eigenvalues and eigenvector notation whenever it is evident from the context.

One can provide approximate solutions to the above problems via sampling. There are several ways to collect representative samples in a network. One straightforward way is to collect independent samples via uniform node or edge sampling. However, uniform sampling is not efficient because the user ID space of the network under consideration need not contain consecutive numbers and in most cases it is sparsely allocated. Hence the sampler wastes many samples issuing invalid IDs resulting in an inefficient and costly data collection method. Moreover, almost all the OSNs impose rate limitations on the API queries, for e.g., Twitter with 313 million active users enforces 15 requests per 15 minutes time window for most of the APIs.[3] Taking into account these limitations, we resort to other feasible techniques.

1.2.1 Local information constraints

To collect information from an OSN, the sampler issues an API query for a particular user which returns its hop neighborhood and the contents published by the user. Though some third parties can obtain the complete database of OSNs (for instance Twitter) with additional expense, we focus here on the typical case where the third party can get information only about the neighbors of a particular user through API queries to it.

The problem of interest to us can be formulated as follows:

Problem Formulation for Network Sampling

- Estimate
$$\nu(G) = \sum_{u \in V} g(u), \quad \bar{\nu}(G) = \frac{1}{|V|} \sum_{u \in V} g(u), \tag{1.1}$$
where $g(u)$ is a bounded function of node u. Sometimes we also focus on the associated estimation problem on edges, with $g(u,v)$ denoting a function over the edge (u,v):
$$\mu(G) = \sum_{(u,v) \in E} g(u,v), \quad \bar{\mu}(G) = \frac{1}{|E|} \sum_{(u,v) \in E} g(u,v). \tag{1.2}$$
- Graph G is unknown.
- Only local information is known: we have information only about the seed nodes and their neighbor IDs, we can only query or visit a neighbor of a node; then, we also have information about the visited nodes and their neighbors.
We will slightly relax this criterion later in Section 2.5 allowing a small number of uniformly sampled queries.

[3] https://dev.twitter.com/rest/public/rate-limits

Note that functions of the form (A.2) are general enough to compute node statistics

$$\nu(G) = \sum_{(u,v)\in E} \frac{g(u)}{d_u}\,, \tag{1.3}$$

where d_u is the degree of node $u \in V$, $g(u)$ is any function of the node u, and statistics of triangles such as the local clustering coefficient of G

$$\begin{aligned} \mu_{\triangle}(G) = \frac{1}{|V|} \sum_{(u,v)\in E} \Bigg\{ & \frac{\mathbb{I}\{d_v > 2\}}{d_v} \\ & \frac{1}{\binom{d_v}{2}} \sum_{a\in\mathcal{N}(v)} \sum_{\substack{b\in\mathcal{N}(v)\\ b\neq a}} \mathbb{I}\Big\{(v,a)\in E\} \cap \{(v,b)\in E\} \cap \{(a,b)\in E\}\Big\}\Bigg\}, \end{aligned} \tag{1.4}$$

where the expression inside the sum is zero when $d_v < 2$ and $\mathcal{N}(v)$ are the neighbors of $v \in V$ in G.

Let $\widehat{\nu}^{(n)}_{\mathrm{X}}(G)$ denote the estimator for $\nu(G)$ from n samples using the technique X and $\widehat{\bar{\nu}}^{(n)}_{\mathrm{X}}(G)$ for $\bar{\nu}(G)$. Similar notation holds for $\mu(G)$ and $\bar{\mu}(G)$.

A possible solution, under the constraint that only local information is available at each node is **snowball sampling**: here after picking an initial node, the sampler probes all its neighbors, and then for each of the neighbor, the process of probing all of its neighbors is repeated. The process continues in this way. One main drawback of this procedure is that the sampled node size increases exponentially and will soon cover the entire network. Another issue is that in case of a very large network, this sampling is asymptotically biased towards the principal eigenvector of the adjacency matrix (called eigenvector centrality) [Newman 2010]: let $\boldsymbol{x}_t$ be a column vector over nodes denoting the number of times each node has been sampled till a time instant t. Then for the initial vector $\boldsymbol{x}_0$ with starting node i, $\boldsymbol{x}_0(i) = 1$ and $\boldsymbol{x}_0(k) = 0$, for all $k \neq i$. The snowball sampling gives the following iteration: $\boldsymbol{x}_t = \boldsymbol{A}\boldsymbol{x}_{t-1}$, which essentially is the power iteration. By representing $\boldsymbol{x}_0$ as $\sum_{i=1}^{|V|} c_i \boldsymbol{u}_i^{\boldsymbol{A}}$ for appropriate c_i's, we now have for large t,

$$\boldsymbol{x}_t \approx c_1 (\lambda_1^{\boldsymbol{A}})^t \boldsymbol{u}_1^{\boldsymbol{A}}.$$

Such a bias is difficult to compensate for since knowledge of the eigenvector centrality requires the entire network or special tools as discussed later in Chapter 3 which will further delay the whole process. In short, this procedure might be more suitable for sampling from a hidden population in the network, which has a relatively small size compared to the actual network size. Popular algorithms like breadth-first search and depth-first search from graph theory can also fit under snow-ball sampling and face similar issues.

A **simple random walk** on a graph under the same local constraint provides a viable solution. In a simple random walk, after picking an initial node randomly or deterministically, the random walk chooses one of the neighbors of the present node uniformly at random and moves to the selected node, and this process continues.

Asymptotically, the choice of the random walk sampler is biased towards large degrees and since such a bias includes only local information, it can be easily reversed. In general, a random walk need not sample the neighbors uniformly and can take any transition probability compliant to the underlying graph. We will discuss more about random walk sampling in Chapter 2.

The burn-in period (or mixing time) of the random walks is the time period after which the random walk will be approximately independent of the initial distribution and produces almost stationary samples. In most of the random walk based estimation procedures, the samples until the burn-in period are discarded in order to provide theoretical guarantees. This poses serious limitations, especially with stringent constraints on the number of samples imposed by API query rates. Furthermore, if we limit the allowed number of samples, it is clearly not known how accurate the estimated value is. A real time assessment of the estimator with Bayesian posterior distribution will be useful in such a situation. We address these problems in sampling in this thesis.

1.3 Spectral Decomposition: Sampling in "Spectral Domain"

Spectral properties of a graph or a network are of interest to diverse fields due to their strong influence in many practical algorithms. As explained later in this section, many properties of networks are concisely abstracted in a few dominant eigenvalues of the matrices associated with networks. However the computational complexity associated with the estimation of eigenvalue spectrum and eigenvectors has been a demanding problem for a long time. Thus, in the context of network science the design of *distributed* spectral decomposition methods is particularly important.

Problem

We study efficient algorithms for spectral decomposition in order to find k dominant eigenvalues (it can be smallest or largest k) and its eigenvectors with high resolution. The algorithms must allow a distributed implementation in the sense that each node does the processing on its own with the data from its one hop neighborhood (see Section 1.2.1). In particular, we restrict our attention to matrices which are graph compliant, i.e., matrix entry at (i, j) is 0 when there is no corresponding edge (i, j) in the graph. This makes development of distributed techniques easier.

To motivate the computation of eigen-spectrum of graph matrices, we now provide a few of its applications.[4]

[4] Eigen-spectrum and spectrum meant the eigenvalues of the graph matrix in the context.

1.3.1 Relevance of spectral decomposition

The existing literature explores many applications of the spectrum of graph matrices. Here we give a limited list of uses of the spectrum of the adjacency and Laplacian matrices. In particular, the eigenvalues and eigenvectors can be used globally or locally. Global applications require a central unit to collect eigenvalues and eigenvector components from all the nodes and then pass this global information to the algorithms behind the application. But in case of local applications, the underlying algorithms run separately at each node by making use of the respective component in the dominant eigenvectors, along with the knowledge of eigenvalues.

The eigen-spectrum of the adjacency matrix is connected with many useful quantities, for instance:

- $|E| = 1/2\sum_{i=1}^{|V|}(\lambda_i^{\boldsymbol{A}})^2$. This follows from the observation that $(\lambda_i^{\boldsymbol{A}})^2$ is an eigenvalue of $\boldsymbol{A}^2$ and the sum over them is the trace of $\boldsymbol{A}^2$. Each of the diagonal elements of $\boldsymbol{A}^2$ is actually the number of walks of length two and the sum of them is twice the number of edges.

- The total number of triangles in G is given by

$$T = \frac{1}{6}\sum_{i=1}^{|V|}(\lambda_i^{\boldsymbol{A}})^3. \tag{1.5}$$

  The sum on the right-hand side is the trace of $\boldsymbol{A}^3$ and the diagonal entries of $\boldsymbol{A}^3$ corresponds to cycles of length three. Each cycle is counted six times; thrice due to each of the vertex in the cycle and twice from each vertex due to the two different directions it has.

- Let $\mathbf{1} = \sum_{i=1}^{|V|} a_i \boldsymbol{u_i^A}$ for some a_i's. Then the total number of walks of length k is

$$\begin{aligned}
\mathbf{1}^\intercal \boldsymbol{A}^k \mathbf{1} &= \left(\sum_{i=1}^{|V|} a_i (\boldsymbol{u_i^A})^\intercal\right) \boldsymbol{A}^k \left(\sum_{i=1}^{|V|} a_i \boldsymbol{u_i^A}\right)\\
&= \left(\sum_{i=1}^{|V|} a_i (\boldsymbol{u_i^A})^\intercal\right) \left(\sum_{i=1}^{|V|} a_i (\lambda_i^{\boldsymbol{A}})^k \boldsymbol{u_i^A}\right)\\
&= \sum_{i=1}^{|V|} a_i^2 (\lambda_i^{\boldsymbol{A}})^k.
\end{aligned}$$

  This can be approximated by $a_1^2(\lambda_1^{\boldsymbol{A}})^k$, when G is connected and non-bipartite (to make $\lambda_i^{\boldsymbol{A}} < \lambda_1^{\boldsymbol{A}}$ for all $i \neq 1$ according to Perron-Frobenius theorem [Horn & Johnson 2012, Section 8.5]).

Following are some more applications of the adjacency and Laplacian matrices related to graphs and networks.

Number of local triangles

The spectral information of the adjacency matrix can be used to obtain information about the global as well as local knowledge of the number of triangles (in other words, about global and local clustering). The total number of triangles in a graph is given in (A.3). Now the number of triangles that a node m participated in is $1/2\sum_{i=1}^{|V|}\lambda_i^3(\boldsymbol{A})\,(\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{A}}(m))^2$. This is because, using the spectral theorem, the number of triangles the node m is involved in is given by

$$\begin{aligned}\frac{1}{2}\boldsymbol{A}^3(mm) &= \frac{1}{2}\sum_{i=1}^{|V|}(\lambda_i^{\boldsymbol{A}})^3\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{A}}(\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{A}})^\intercal(mm)\\ &= \frac{1}{2}\sum_{i=1}^{|V|}(\lambda_i^{\boldsymbol{A}})^3(\boldsymbol{u}_{\boldsymbol{i}}^{\boldsymbol{A}}(m))^2.\end{aligned}$$

Hence if we calculate top-k eigenvalues and eigenvector components locally at node m, we can approximate with good accuracy how much connected its neighbors are.

Dimensionality reduction, link prediction and Weak and strong ties

After the computation of top-k eigenvectors, each node is mapped to a point in $\mathbb{R}^k$ space with the eigenvector components and closeness in the new space implies affinity in terms of the position in the network. Therefore new links can be suggested among unconnected nodes when the distance between them in $\mathbb{R}^k$ space is small [Kempe & McSherry 2008].

Weak ties occur when the endpoints of an edge are part of well connected nodes, but with very few common friends between the endpoints, while strong ties happen in the opposite sense [Easley & Kleinberg 2010, Chapter 3]. The k-dimensional vector associated with the endpoints can be used to find out weak or strong ties.

Finding near-cliques

Typical graph clustering works on the whole graph and will often assign isolated nodes to some clusters, and subsequently would fail to detect communities with good internal coherence. Therefore it is practically relevant to find communities which are like cliques and extract it from the main graph. The work in [Prakash et al. 2010] shows that the spectrum of adjacency matrix is useful for this. They propose the idea of EigenSpokes which is a phenomenon whereby straight lines are observed in the eigenvector-eigenvector scatter plots of the adjacency matrix. It is then shown that nodes that are placed close to one another on the EigenSpokes have the same set of neighbors and hence can be used to detect clique-like structures.

Spectral clustering

The problem of finding clusters in a network (especially in social networks) is an old one with many developments in recent years. Among the studied techniques, spectral clustering is a prominent solution [Von Luxburg 2007]. In its basic statement, the spectral clustering algorithm takes the first k normalized eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ of the adjacency or the Laplacian matrix. Let $\boldsymbol{\phi}(h)$ be a vector made of components of node h in $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$. The k-means clustering algorithm is then applied to different $\boldsymbol{\phi}(h)$'s, and this partitions the nodes in the network. In fact, the matrix to be considered is dependent on the objective function to optimize (*average association* in case of adjacency matrix instead of *normalized cut* in Laplacian) [Shi & Malik 2000]. The main bottleneck in spectral clustering is the computation of dominant eigenvectors, which we try to accomplish here with much less complexity.

Number of spanning trees

A subgraph of G, also a tree, which is incident on all the vertices is called a spanning tree. The number of spanning trees of G is given by a well known expression containing the eigenvalues of Laplacian matrix,

$$\frac{\lambda_1^L \lambda_2^L \ldots \lambda_{|V|-1}^L}{|V|}.$$

More details about the ways to compute eigenvalues and eigenvectors are explained in Chapter 3.

1.4 Extreme Value Theory and its Applications

After collecting random samples from the network, a question arises on how to make more inferences with them (other than estimation). To that end, note that many social network entities are correlated. For instance, if we take a co-authorship network where the nodes are the authors and a link is established when two authors write a research paper together, the highly cited or high profile researchers tend to make connections more to each other. How can we extract any information about the correlation structure in the network with a few random samples? In such scenarios, mathematical tools from different fields like extreme value theory (EVT) appear to be very helpful. In this thesis, we also investigate the relationship between extreme value theory and network sampling. Assuming that a stationary sequence of random samples from a network is available, we study about extremal properties like the first time to hit a large degree node, the clusters explored during the sampling process, etc.

Extreme value theory, in general, is the study of rare events. In particular, it deals with the convergence of $M_n := \max\{X_1, \ldots, X_n\}$ of the random samples $\{X_i\}_{i \leqslant n}$. In the following subsection, we give a very short introduction to EVT.

Independent and identically distributed samples

In case of i.i.d. samples with $F_X(x)$ as the cumulative distribution function, it is clear that $M_n \to x_F$ a.s., where $x_F := \sup\{x \in \mathbb{R} : F_X(x) < 1\}$. The EVT studies the order of magnitude of maximum M_n. The following theorem lays the foundation of the EVT of i.i.d. samples.

Theorem 1.1 ([Embrechts et al. 2011, Fisher-Tippet theorem]). *Let $\{X_n\}_{n\geqslant 1}$ be a sequence of i.i.d. random variables. If there exist normalizing constants $c_n > 0$, and $d_n \in \mathbb{R}$ and some non-degenerate random variable Y such that*

$$\frac{M_n - d_n}{c_n} \to Y,$$

then Y has a Fréchet, Weibull or Gumbel distribution:

$$\begin{aligned}
&\textit{Fréchet}: && \begin{cases} 0, & x \leqslant 0 \\ \exp\{-x^{-\alpha}\}, & x > 0 \end{cases} \quad \alpha > 0. \\
&\textit{Weibull}: && \begin{cases} \exp\{-(-x)^{\alpha}\}, & x \leqslant 0 \\ 1, & x > 0 \end{cases} \quad \alpha > 0. \\
&\textit{Gumbel}: && \exp\{-e^{-x}\}, x \in \mathbb{R}.
\end{aligned}$$

The condition $1 - F_X(x) = L_X(x)x^{-\alpha}$ with the function $L_X(x)$ slowly varying is called *regularly varying* with exponent $-\alpha$. Here a function $L(x)$ is called slowly varying if for every $c > 0$,

$$\lim_{x\to\infty} \frac{L(cx)}{L(x)} = 1.$$

Note that the concept of *regularly varying* function is weaker than the typical power-law definition $1 - F_X(x) \propto x^{-\alpha}$ or $\mathbb{P}(X = k) \propto k^{-\alpha-1}$.

Interestingly, when the distribution of the random variables $\{X_n\}$ is regularly varying, only one of the extremal distributions, Fréchet, is the possible distribution:

Theorem 1.2 ([Embrechts et al. 2011, Theorem 3.3.7]). *Let $\{X_n\}_{n\geqslant 1}$ be a sequence of i.i.d. unbounded random variables having regularly varying distribution with exponent $1 - \tau$, $\tau > 1$, then*

$$\frac{M_n}{u_n} \to Y,$$

where Y has a Fréchet distribution with parameter $\alpha = \tau - 1$ and

$$u_n := \sup\left\{x : 1 - F_X(x) \geqslant \frac{1}{n}\right\}.$$

Stationary samples

Since we are interested in random walks in this thesis, the focus is on extreme value theory for stationary sequences $\{X_i\}_{i\leqslant n}$. Here the classical techniques study the maximum M_n of $\{X_i\}_{i\leqslant n}$ using the maximum $\widetilde{M_n}$ of the associated i.i.d. samples $\{\widetilde{X_i}\}_{i\leqslant n}$. In particular, under proper mixing conditions, $c_n^{-1}(\widetilde{M_n} - d_n) \to H$ and $c_n^{-1}(M_n - d_n) \to G$ are related as $G = H^{\theta}$, where θ is called extremal index. It turns out that extremal index is related to several interesting extremal events in sampling. We will explore extremal index and its applications in detail in Chapter 6.

1.5 Contribution of the Thesis

We make the following contributions on random walk based network estimation and on distributed spectral decomposition.

1.5.1 Distributed ways to sample in spectral domain

In this work, we address the problem of finding top-k dominant eigenvalues (smallest or largest) and corresponding eigenvectors of symmetric graph matrices in networks in a distributed way. We propose a novel idea called complex power iterations to decompose the eigenvalues and eigenvectors at node level, analogous to time-frequency analysis in signal processing. At each node, eigenvalues will correspond to the frequencies of spectral peaks and respective eigenvector components are the amplitudes at those points. Based on complex power iterations and motivated from fluid diffusion processes in networks, we devise distributed algorithms with different orders of approximation. We also introduce a Monte Carlo technique with gossiping which substantially reduces the computational overhead. An equivalent parallel random walk algorithm is also presented. We validate the algorithms with simulations on real-world networks. Our formulation of the spectral decomposition can be easily adapted to a simple algorithm based on quantum random walks. With the advent of quantum computing, the proposed quantum algorithm will be extremely useful.

We then extend the aforementioned distributed techniques to detect, with higher resolution, closely situated eigenvalues and corresponding eigenvectors of symmetric graph matrices. We model the system of graph spectral computation as physical systems with Lagrangian and Hamiltonian dynamics. The spectrum of Laplacian matrix, in particular, is framed as a classical spring-mass system with Lagrangian dynamics. The spectrum of any general symmetric graph matrix turns out to have a simple connection with quantum systems and it can be thus formulated as a solution to a Schrödinger-type differential equation. Taking into account the higher resolution requirement in the spectrum computation and the related stability issues in the numerical solution of the underlying differential equation, we propose the application of symplectic integrators to the computation of eigenspectrum. The effectiveness of the proposed techniques is demonstrated with numerical simulations on real-world networks of different sizes and complexities.

1.5.2 Network sampling with random walk techniques

Non-asymptotically unbiased sampling and Bayesian inference

Are OSN-A users more likely to form friendships with those with similar attributes? Do users in OSN-B score a content X more favorably than another content Y? Such questions frequently arise in the context of Social Network Analysis (SNA) but often crawling an OSN network via its API is the only way to gather data for a third party. To date, the majority of public datasets are formed from partial API crawls and thus they lack statistical guarantees, severely limiting SNA research progress. Using regenerative properties of the random walks, we propose estimation techniques based on short crawls that have proven statistical guarantees: non-asymptotic unbiasedness and evasion of burn-in time. Moreover, our short crawls can be implemented in massively distributed algorithms. We also provide an adaptive crawler that makes our method parameter-free, significantly improving our statistical guarantees. We then derive an approximation to the Bayesian posterior of the estimates. In addition, we obtain an estimator for the expected value of node and edge statistics in an equivalent configuration model or Chung-Lu random graph model of the given network (where nodes are connected randomly) and use it as a basis for testing null hypotheses. The theoretical results are supported with simulations on a variety of real-world networks.

Reinforcement learning based sampling

Reinforcement learning (RL) provides a way to approach the estimation of average of network functions. The RL based approach used in this thesis is also based on the regenerative properties of the random walks and hence avoids the burn-in time barrier. The RL-technique is essentially a stochastic approximation formed from the Poisson equation of an associated semi-Markov process. The performance of this technique depends on the stepsizes associated with the learning algorithm. This technique can be placed as an intermediate technique between pure Markov chain Monte Carlo (MCMC) iteration (stochastic) and relative value iteration (deterministic). The stepsizes control the stability of the RL-technique and its trajectories are much more stable than that of the standard random walk based estimation procedures. Its performance is also comparable to respondent driven sampling which has small asymptotic variance than many other estimators.

1.5.3 Extreme value theory and network sampling processes

This work explores the dependence structure in the sampled sequence of an unknown network. We consider randomized algorithms to sample the nodes and study extremal properties in any associated stationary sequence of characteristics of interest like node degrees, number of followers or income of the nodes in Online Social Networks, etc., which satisfy two mixing conditions. Several useful extremes of the sampled sequence like kth largest value, clusters of exceedances over a threshold, first hitting time of a large value, etc., are investigated. We abstract the dependence and the statistics of extremes into a single parameter that appears in extreme value theory, called extremal index (EI). In this work, we derive this parameter analytically and also estimate it empirically. We propose the use of EI as a parameter to compare different sampling procedures. As a specific example, degree correlations between neighboring nodes are studied in detail with three prominent random walks as sampling techniques.

1.6 Organization of the Thesis

This thesis is organized as follows. In Chapter 2, we review some background in random walks on graphs and present classical results.

In Chapter 3 we present diffusion based algorithms for eigen-spectrum computation of symmetric graph matrices. Next, an analysis of the complexity and rate of convergence is provided. After explaining the choice of parameters and connection with quantum computing, algorithms which decompose the eigenvalues with higher resolution, are explained.

Chapter 4 focuses on the estimation of sum function $\mu(G)$ of edges. The idea of "super-node" is first developed and it provides a way to tackle disconnected graphs. We then derive a non-asymptotically unbiased estimator. Some of the extensions provided in this chapter are the posterior distribution of the estimator and testing with a null hypothesis that the function values are generated from a random graph model.

Chapter 5 deals with an estimator for average function $\bar{\nu}(G)$ of nodes and is loosely based on the idea of regeneration of Markov chains. Algorithms based on stochastic approximation and reinforcement learning are proposed and compared with classical random walk estimators using simulations on real-world networks.

In Chapter 6, we explain the use of the parameter extremal index from EVT to find many extremal properties of the network sampling process. Theoretical study of the parameter is first introduced and it is followed by simulations on real data.

Finally Chapter A.3 concludes the thesis and presents future directions to explore.

1.7 Publications based on the Thesis

- [Avrachenkov et al. 2016c] Konstantin Avrachenkov, Bruno Ribeiro and Jithin K. Sreedharan. *Inference in OSNs via Lightweight Partial Crawls.* ACM SIGMETRICS Performance Evaluation Review, vol. 44, no. 1, pages 165 - 177, June 2016.

- [Avrachenkov et al. 2016b] Konstantin Avrachenkov, Philippe Jacquet and Jithin K. Sreedharan. *Distributed Spectral Decomposition in Networks by Complex Diffusion and Quantum Random Walk.* In Proc. IEEE International Conference on Computer Communications (INFOCOM), April 2016.

- [Avrachenkov et al. 2016a] Konstantin Avrachenkov, Vivek S. Borkar, Arun Kadavankandy and and Jithin K. Sreedharan. *Comparison of Random Walk based Techniques for Estimating Network Averages."* International Conference on Computational Social Networks (CSoNet), August 2016.

- Konstantin Avrachenkov, Natalia M. Markovich and Jithin K. Sreedharan. *Distribution and Dependence of Extremes in Network Sampling Processes*

- [Avrachenkov et al. 2015b] Computational Social Networks, Springer, 2015.
- Third International IEEE Workshop on Complex Networks and their Applications, November 2014.

Chapter 2

Review of Random Walk Sampling in Networks

From the problem formulation given in Section 1.2 and the explanations there, the random walk (RW) appears as a viable solution for sampling and estimating network functions.

A random walk on a graph is essentially a Markov chain with state space V and transition probability matrix $\boldsymbol{P}$. Before stating some facts about random walk on graphs, we now make the simple observation that an ergodic random walk on the state space of node set V with transition probability matrix $\boldsymbol{P} = [p_{ij}]$ and stationary probability distribution $\boldsymbol{\pi}$ to estimate node functions of the form $\nu(G)$ or $\bar{\nu}(G)$ (see (A.1)) can be extended to estimate the edge functions $\mu(G)$ and $\bar{\mu}(G)$ (see (A.2)) as follows: If $\{X_n\}_{n \geqslant 1}$ is the random walk, then $\{X_n, X_{n+1}\}_{n \geqslant 1}$ is also a Markov chain. Since $\{X_n\}$ is ergodic, $\{X_n, X_{n+1}\}$ is also ergodic and the stationary distribution of the chain $\{X_n, X_{n+1}\}$ is $\boldsymbol{\pi}'(i,j) = \boldsymbol{\pi}(i)p_{ij}$ for an edge (i,j). This two dimensional chain with the ergodic theorem (Theorem 2.1 below) gives estimates for the edge functions $\mu(G)$ and $\bar{\mu}(G)$. Thus for simplicity, we concentrate below on the random walk over V and all the theory can be extended to two dimensional random walk over edges.

Organization

Section 2.1 introduces the idea of mixing times and many classical results. Section 2.2 describes some examples of the random walks on graphs. In Section 2.3, we list some techniques to compare different random walks on graph. Section 2.4 explains the ways to achieve non-asymptotic unbiasedness with random walk estimators. Section 2.5 deals with the case when uniform node sampling is allowed in addition to random walk samples. Section 2.6 concludes the chapter.

2.1 Some Results on Convergence and Mixing Times in Markov Chains

Many results in Markov chains do not necessarily need the conditions such as aperiodicity and reversibility. We will state it in the context. We also limit our studies to finite graphs. The notation $\mathbb{E}_\xi$ or $\mathbb{P}_\xi$ indicates expectation or probability with respect to the initial distribution ξ.[1]

The main result we use throughout this thesis is the ergodic theorem (a.k.a., strong law of large numbers for the Markov chain):

Theorem 2.1. *Let f be a real-valued function $f : V \mapsto \mathbb{R}$. If $\{X_n\}$ is an irreducible finite Markov chain with stationary distribution π, then for any starting distribution ξ,*

$$\mathbb{P}_\xi \left\{ \lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} f(X_k) = \mathbb{E}_\pi[f(X_0)] \right\} = 1.$$

Later Theorem 2.5 provides a concentration bound in the context of ergodic theorem. Now the central limit theorem for Markov chains is given below. Let $Y_n \xrightarrow{d} Y$ denote convergence in distribution of random variables Y_n to Y.

Theorem 2.2 ([Roberts & Rosenthal 2004])**.** *Let f be a real-valued function $f : V \mapsto \mathbb{R}$ with $\mathbb{E}_\pi[f^2(X_0)] < \infty$. For a finite irreducible Markov chain $\{X_n\}$ with stationary distribution π,*

$$\sqrt{n}\left(\frac{1}{n} \sum_{k=0}^{n-1} f(X_k) - \mathbb{E}_\pi[f(X_0)] \right) \xrightarrow{d} \text{Normal}(0, \sigma_f^2),$$

irrespective of the initial distribution.[2] *It then follows that*

$$\begin{aligned} \sigma_f^2 &= \lim_{n\to\infty} n \times \mathbb{E}\left[\left\{ \frac{1}{n} \sum_{k=0}^{n-1} f(X_k) - \mathbb{E}_\pi[f(X_0)] \right\}^2 \right] \\ &:= \lim_{n\to\infty} \frac{1}{n} \text{Var}\left[\sum_{k=0}^{n-1} f(X_k) \right]. \end{aligned} \tag{2.1}$$

Note that both the above theorems hold for finite irreducible periodic chains also (with a unique solution to $\boldsymbol{\pi}^\intercal \boldsymbol{P} = \boldsymbol{\pi}^\intercal$).

Let us define the fundamental matrix of a Markov chain as $\boldsymbol{Z} := (\boldsymbol{I} - \boldsymbol{P} + \mathbf{1}\boldsymbol{\pi}^\intercal)^{-1}$. For two functions $f, g : V \to \mathbb{R}$, we define $\sigma_{ff}^2 := 2\langle \boldsymbol{f}, \boldsymbol{Z}\boldsymbol{f}\rangle_\pi - \langle f, f\rangle_\pi - \langle f, \mathbf{1}\boldsymbol{\pi}^\intercal f\rangle_\pi$, and $\sigma_{fg}^2 := \langle \boldsymbol{f}, \boldsymbol{Z}\boldsymbol{g}\rangle_\pi + \langle \boldsymbol{g}, \boldsymbol{Z}\boldsymbol{f}\rangle_\pi - \langle f, g\rangle_\pi - \langle f, \mathbf{1}\boldsymbol{\pi}^\intercal g\rangle_\pi$, where $\langle \boldsymbol{x}, \boldsymbol{y}\rangle_\pi := \sum_i x_i y_i \pi_i$, for any two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^{|V|\times 1}$, π being the stationary distribution of the Markov chain. Now [Brémaud 1999, Theorem 6.5] states that σ_f^2 in Theorem 2.2 is same as σ_{ff}^2 defined above.

We will also need the following theorem from [Nummelin 2002].

Theorem 2.3. *If f, g are two functions defined on the states of a random walk, define the vector sequence $\boldsymbol{Z}_k = \begin{bmatrix} f(X_k) \\ g(X_k) \end{bmatrix}$ the following central limit theorem holds*

$$\sqrt{n}\left(\frac{1}{n} \sum_{k=1}^{n} \boldsymbol{Z}_k - \mathbb{E}_\pi(\boldsymbol{Z}_k) \right) \xrightarrow{d} \text{Normal}(0, \boldsymbol{\Sigma}),$$

[1] The probability distribution ξ on a finite alphabet is also denoted by a vector bold notation $\boldsymbol{\xi}$. We drop the subscript in $\mathbb{E}$ and $\mathbb{P}$ if it is implied from the context.

[2] Normal(a, b) is Gaussian random variable with mean a and variance b.

where $\mathbf{\Sigma}$ *is* 2×2 *matrix such that* $\mathbf{\Sigma}_{11} = \sigma^2_{ff}, \mathbf{\Sigma}_{22} = \sigma^2_{gg}$ *and* $\mathbf{\Sigma}_{12} = \mathbf{\Sigma}_{21} = \sigma^2_{fg}$.

The time required by a random walk or Markov chain to reach stationarity is measured by a parameter called *mixing time* defined as

$$t_{\text{mix}}(\varepsilon) := \min\{t : \max_{u \in V} \|\boldsymbol{P}^t(x, \cdot) - \boldsymbol{\pi}\|_{\text{TV}} \leqslant \varepsilon\},$$

where $\|\xi_1 - \xi_2\|_{\text{TV}} := \max_{A \subset V} |\xi_1(A) - \xi_2(A)|$ is the total variational distance between the probability distributions ξ_1 and ξ_2.

Spectral gap and bounds of mixing time

Mixing time is difficult to accurately calculate or estimate, and there are many ways to find lower and upper bounds of mixing time. The eigenvalues of the transition probability matrix provide an useful tool.

Let $-1 \leqslant \lambda^{\boldsymbol{P}}_{|V|} \leqslant \ldots \lambda^{\boldsymbol{P}}_2 < \lambda^{\boldsymbol{P}}_1 = 1$ be the eigenvalues of a reversible ergodic Markov chain. Let us define the *spectral gap* as $\delta := 1 - \lambda^{\boldsymbol{P}}_2$, the *absolute spectral gap* as $\delta_* := 1 - \max\left\{|\lambda^{\boldsymbol{P}}_2|, \ldots, |\lambda^{\boldsymbol{P}}_{|V|}|\right\}$, and the *relaxation time* as $t_{\text{rel}} := 1/\delta_*$.[3]

Theorem 2.4 ([Aldous & Fill 2002, Chapter 3])**.** *Fix* $0 < \varepsilon < 1$ *arbitrary. Assume* $\boldsymbol{P}$ *irreducibile, aperiodic and reversible with stationary distribution* $\boldsymbol{\pi}$. *Then*

$$(t_{\text{rel}} - 1) \log\left(\frac{1}{2\varepsilon}\right) \leqslant t_{\text{mix}}(\varepsilon) \leqslant \log\left(\frac{1}{\varepsilon \min_i \boldsymbol{\pi}(i)}\right) t_{\text{rel}}.$$

Effect of bottlenecks of the graph in mixing time

The effect of graph topology on mixing time is usually illustrated by a dumbbell graph: two cliques joined by a line graph. Heuristically one can see that in this case the random walk will take much longer time to mix. This is rigorously stated by the notion of *bottleneck ratio.*

Let

$$Q(x, y) = \boldsymbol{\pi}(x)\boldsymbol{P}(x, y), \quad Q(C, D) = \sum_{x \in C, y \in D} Q(x, y),$$

for $x, y \in V$. Now the bottleneck ratio or conductance or Cheeger constant of the set $A \subset V$ and that for the entire chain are defined as,

$$\Phi(A) = \frac{Q(A, A^c)}{\boldsymbol{\pi}(A)}, \quad \Phi_* = \min_{A : \boldsymbol{\pi}(A) \leqslant 1/2} \Phi(A).$$

The normalization is introduced here to give weightage to relevant sets. It is related to mixing time as ([Levin et al. 2008, Theorem 7.3])

$$t_{\text{mix}}(1/4) \geqslant \frac{1}{4\Phi_*},$$

[3] For a lazy Markov chain with t.p.m. as $\frac{I + \boldsymbol{P}}{2}$, $\delta_* = \delta$.

or via Cheeger inequality ([Jerrum & Sinclair 1989])

$$\Phi_*^2/2 \leqslant \delta \leqslant 2\Phi_*, \tag{2.2}$$

if the Markov chain is reversible.

In order to show how the bottleneck captures the graph structure, consider a simple random walk on d-regular graph. Then $\Phi(A) = d^{-1}|E(A,A^c)|/|A|$ with $|E(A,A^c)|$ indicating the number of edges going outside the set A. Suppose the graph contains a large component with very few edges going outside the component, then $\Phi(A)$ will be very low for the component and this makes the mixing time large.

Concentration bound for ergodic theorem

The following theorem gives more informative results about the ergodic theorem and clearly illustrates the use of mixing time and the spectral gap of $\boldsymbol{P}$ in providing confidence interval for the ergodic estimate.

Theorem 2.5 ([Levin et al. 2008, Theorem 12.19])**.** *Let $\{X_n\}$ be a reversible Markov chain. If $r \geqslant t_{\text{mix}}(\varepsilon/2)$ and $n \geqslant [4\,\text{Var}_\pi[f]/(\eta^2\varepsilon)]\,t_{\text{rel}}$, then for any starting point $u \in V$,*

$$\mathbb{P}_u\left\{\left|\frac{1}{n}\sum_{k=0}^{n-1} f(X_{r+k}) - \mathbb{E}_\pi[f(X_0)]\right| \geqslant \eta\right\} \leqslant \varepsilon.$$

How the random walks are useful in complex networks?

First, an expander graph G is defined as follows: let $\boldsymbol{P}$ be the transition probability matrix of simple random walk. Then for the simple random walk on the expander, there exists an $\alpha > 0$ such that the Cheeger constant of the chain $\Phi_* \geqslant \alpha$. This implies with the Cheeger inequality (2.2) and Theorem 2.4 that the mixing time is $\mathcal{O}(\log n)$. In [Mihail et al. 2003], it is shown that the preferential attachment random graph is an expander. Note that Internet and several small world networks are modeled using preferential attachment.

2.2 Description of the Techniques

In light of the ergodic theorem and the concentration result Theorem 2.5, a Markov chain on the state space V can be used to estimate various graph functions. Such Markov chains are generally called random walks on graph (RW) and this section reviews some of the popular random walk based sampling techniques.

We do not consider many of the variations of random walks such as multidimensional, non-backtracking, etc., in this thesis and thus in this review, since our focus is mainly on providing general techniques such as non-asymptotic unbiasedness and real-time computation of the posterior distribution.

2.2.1 Simple random walk (SRW)

A simple random walk (SRW) is a time-homogeneous first-order Markov chain whose state space is V and the transition probabilities are given as

$$p_{ij} = \mathbb{P}\left(X_{t+1} = j | X_t = i\right) = \frac{1}{d_i},$$

if there is a link between i and j, i.e., $(i, j) \in E, d_i$ being the degree of node i. One can think of the simple random walker as a process that traverses the links of the graph in a purely random fashion. We can define $\boldsymbol{P}$ the transition probability matrix (t.p.m) of the Random walk as an $|V| \times |V|$ matrix, such that $\boldsymbol{P}_{ij} = p_{ij}$. Since we consider undirected networks, our random walk is time reversible. When the graph is connected the transition probability matrix $\boldsymbol{P}$ is irreducible and by the Perron-Frobenius theorem there always exists a unique stationary probability vector $\boldsymbol{\pi} \in \mathbb{R}^{|V| \times 1}$ which solves $\boldsymbol{\pi}^\intercal \boldsymbol{P} = \boldsymbol{\pi}^\intercal$, which is in fact $\boldsymbol{\pi}(i) = \frac{d_i}{2|E|}$. As the state space is finite, the Markov chain is also positive recurrent and the quantities such as hitting times, and cover times are finite and well-defined. In short, in graph theoretic terminology for an undirected graph, irreducibility means connectedness and aperiodicity implies that the graph is non-bipartite. Note that with the local information constraints (see Section 1.2.1) the random walk based techniques can be easily implemented via APIs of OSNs and can also be made distributed.

Stationary average with uniform distribution

The SRW is biased towards higher degree nodes and from the ergodic theorem the sample averages converge to the stationary average. Hence if the aim is to estimate an average function like

$$\bar{\nu}(G) = \frac{1}{|V|} \sum_{u \in V} g(u),$$

the RW needs to have uniform stationary distribution. Alternatively the RW should be able to unbias it locally. In the case of SRW by modifying the function g to $g'(u) = g(u)/\boldsymbol{\pi}(u)$ this can be accomplished. But since $\boldsymbol{\pi}(u)$ contains $|E|$ and the knowledge of $|E|$ is not available to us initially, it also needs to be estimated. To overcome this problem, we consider the following variations of SRW.

2.2.2 Metropolis-Hastings random walk

We present here the Metropolis Hastings MCMC (MH-MCMC) algorithm. When the chain is in state i, it chooses the next state j according to transition probability p_{ij}. It then jumps to this state with probability q_{ij} or remains in the current state i with probability $1 - q_{ij}$, where q_{ij} is given as below

$$q_{ij} = \begin{cases} \min\left(\frac{p_{ji}}{p_{ij}}, 1\right) & \text{if } p_{ij} > 0, \\ 1 & \text{if } p_{ij} = 0. \end{cases} \tag{2.3}$$

Therefore the effective jump probability from state i to state j is $q_{ij}p_{ij}$, when $i \neq j$. It follows then that such a process represents a Markov chain with the following transition matrix $\boldsymbol{P}^{\mathrm{MH}}$

$$\boldsymbol{P}_{ij}^{\mathrm{MH}} = \begin{cases} \frac{1}{\max(d_i,d_j)} & \text{if } (i,j) \in E \\ 1 - \sum_{k \neq i} \frac{1}{\max(d_i,d_k)} & \text{if } i = j \\ 0 & \text{if } (i,j) \notin E, i \neq j. \end{cases}$$

This chain is reversible with stationary the distribution $\boldsymbol{\pi}(i) = 1/|V| \; \forall i \in V$. Therefore the following estimate for $\bar{\nu}(G)$ using MH-MCMC, $\{X_n\}$ being MH-MCMC samples, is asymptotically consistent.

$$\widehat{\bar{\nu}}_{\mathrm{MH}}^{(n)}(G) = \frac{1}{n} \sum_{k=1}^{n} g(X_k).$$

By using the CLT for Markov chains Theorem 2.2, we can show the following central limit theorem for MH-MCMC.

Proposition 2.6 (Central Limit Theorem for MH-MCMC)**.** *For MH-MCMC with uniform target stationary distribution it holds that*

$$\sqrt{n}\left(\widehat{\bar{\nu}}_{\mathrm{MH}}^{(n)}(G) - \bar{\nu}(G)\right) \xrightarrow{d} \mathrm{Normal}(0, \sigma_{\mathrm{MH}}^2),$$

as $n \to \infty$, *where* $\sigma_{\mathrm{MH}}^2 = \sigma_{gg}^2 = \frac{2}{|V|}\boldsymbol{g}^\intercal \boldsymbol{Z}\boldsymbol{g} - \frac{1}{|V|}\boldsymbol{g}^\intercal\boldsymbol{g} - \left(\frac{1}{|V|}\boldsymbol{g}^\intercal\mathbf{1}\right)^2$.

2.2.3 Respondent driven sampling technique (RDS-technique)

The estimator with respondent driven sampling uses the SRW on graphs but applies a correction to the estimator to compensate for the non-uniform stationary distribution.

$$\widehat{\bar{\nu}}_{\mathrm{RDS}}^{(n)}(G) = \frac{\sum_{k=1}^{n} g(X_k)/d_{X_k}}{\sum_{k=1}^{n} 1/d_{X_k}} \tag{2.4}$$

We define $h^{'}(X_k) := g(X_k)/d_{X_k}, h(X_k) := 1/d_{X_k}$.

The asymptotic unbiasedness derives from the following result on the ratio form of the law of large numbers for Markov chain samples [Meyn & Tweedie 2012, Theorem 17.2.1]:

Theorem 2.7. *Let f and g be real-valued functions on V. If $\{X_n\}$ is an irreducible finite Markov chain with stationary distribution π, then for any starting distribution ξ,*

$$\mathbb{P}_{\xi}\left\{\lim_{n\to\infty} \frac{\sum_{k=0}^{n-1} f(X_k)}{\sum_{k=0}^{n-1} g(X_k)} = \frac{\mathbb{E}_{\pi}[f(X_0)]}{\mathbb{E}_{\pi}[g(X_0)]}\right\} = 1$$

Now for the central limit theorem, we have:

Proposition 2.8 (Asymptotic Distribution of RDS Estimate)**.** *The RDS estimate $\widehat{\bar\nu}^{(n)}_{\mathrm{RDS}}(G)$ satisfies a central limit theorem given below*

$$\sqrt{n}\left(\widehat{\bar\nu}^{(n)}_{\mathrm{RDS}}(G) - \bar\nu(G)\right) \xrightarrow{d} \mathrm{Normal}(0, \sigma^2_{\mathrm{RDS}}),$$

where σ^2_{RDS} is given by

$$\sigma^2_{\mathrm{RDS}} = d^2_{\mathrm{av}}\left(\sigma_1^2 + \sigma_2^2\mu^2(G) - 2\mu(G)\sigma^2_{12}\right),$$

where $\sigma_1^2 = \frac{1}{|E|}\boldsymbol{g}^\intercal \boldsymbol{Z}\boldsymbol{h}^{'} - \frac{1}{2|E|}\sum_x \frac{g(x)^2}{d_x} - \left(\frac{1}{2|E|}\boldsymbol{g}^\intercal\boldsymbol{1}\right)^2$, $\sigma_2^2 = \sigma^2_{hh} = \frac{1}{|E|}\boldsymbol{1}^\intercal\boldsymbol{Z}\boldsymbol{h} - \frac{1}{2|E|}\boldsymbol{h}^\intercal\boldsymbol{1} - (\frac{1}{d_{\mathrm{av}}})^2$ and $\sigma^2_{12} = \frac{1}{2|E|}\boldsymbol{g}^\intercal\boldsymbol{Z}\boldsymbol{h} + \frac{1}{2|E|}\boldsymbol{1}^\intercal\boldsymbol{Z}\boldsymbol{h}^{'} - \frac{1}{2|E|}\boldsymbol{g}^\intercal\boldsymbol{h} - \frac{1}{d_{\mathrm{av}}}\frac{1}{2|E|}\boldsymbol{1}^\intercal\boldsymbol{g}$.

Proof. Let $h^{'}(x) := \frac{g(x)}{d_x}$ and $h(x) := \frac{1}{d_x}$. Define the vector $\boldsymbol{Y}_k = \begin{bmatrix} h^{'}(X_k) \\ h(X_k)\end{bmatrix}$, and let $\boldsymbol{Z}_n = \sqrt{n}\,(\frac{1}{n}\sum_{k=1}^n \boldsymbol{Y}_k - \mathbb{E}_\pi(\boldsymbol{Y}_1))$. Then by Theorem 2.3, $\boldsymbol{Z}_n \xrightarrow{\mathcal{D}} \mathrm{Normal}(0, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is defined as in the theorem. On the other hand, by Skorohod's representation theorem [Billingsley 2008] in a space $(\Omega, \mathcal{F}, \mathbb{P})$, $\Omega \subset \mathbb{R}^2$, there is an embedding of $\boldsymbol{Z}_n$ such that $\boldsymbol{W}_n \to \boldsymbol{W}$ a.s. such that $\boldsymbol{W}_n \overset{\mathcal{D}}{=} \boldsymbol{Z}_n$ and $\boldsymbol{W} \sim \mathrm{Normal}(0, \boldsymbol{\Sigma})$. Let

$$\boldsymbol{Z}_n := \begin{bmatrix} Z_n^1 \\ Z_n^2 \end{bmatrix}, \boldsymbol{W}_n := \begin{bmatrix} W_n^1 \\ W_n^2 \end{bmatrix}, \boldsymbol{W} := \begin{bmatrix} W^1 \\ W^2 \end{bmatrix}, \mu_{h'} := \sum_{u\in V} h'(u), \text{ and } \mu_h := \sum_{u \in V} h(u).$$

Now

$$\begin{aligned}\frac{\sum_{t=1}^n h^{'}(X_t)}{\sum_{t=1}^n h(X_t)} &\overset{\mathcal{D}}{=} \frac{\frac{1}{\sqrt{n}}W_n^1 + \mu_{h'}}{\frac{1}{\sqrt{n}}W_n^2 + \mu_h} = \frac{W_n^1 + \sqrt{n}\mu_{h'}}{W_n^2 + \sqrt{n}\mu_h} = \frac{W_n^1 + \sqrt{n}\mu_{h'}}{\sqrt{n}\mu_h(1 + \frac{W_n^2}{\sqrt{n}\mu_h})} \\ &= \frac{1}{\sqrt{n}\mu_h}\left(W_n^1 - \frac{W_n^1 W_n^2}{\sqrt{n}\mu_h} + \sqrt{n}\mu_{h'} - \frac{W_n^2\mu_{h'}}{\mu_h} + \mathcal{O}(\frac{1}{\sqrt{n}})\right).\end{aligned}$$

This gives

$$\sqrt{n}\left(\frac{\sum_{t=1}^n h^{'}(X_t)}{\sum_{t=1}^n h(X_t)} - \frac{\mu_{h'}}{\mu_h}\right) \xrightarrow{\mathcal{D}} \frac{1}{\mu_h}\left(z_1 - z_2\frac{\mu_{h'}}{\mu_h}\right),$$

since the term $\mathcal{O}(\frac{1}{\sqrt{n}})$ tends to zero in probability, and using Slutsky's lemma [Billingsley 2008]. The result then follows from the fact that $\boldsymbol{z} \sim \mathrm{Normal}(0, \boldsymbol{\Sigma})$. □

2.3 Comparing Random Walk Techniques

Random walks can be compared in many ways. Two prominent techniques are in terms of the mixing time t_{mix} and the asymptotic variance σ_f^2 (2.1). Mixing time is relevant in the situations where the speed at which the RW approach the stationary distribution matter. However, many MCMC algorithms leave some initial samples to get rid of the independence on the initial distribution (called burn-in period) and this accounts for mixing time. After the burn-in period, the number of samples needed for achieving a certain estimation accuracy can be determined from the Gaussian approximation given by the central limit theorem (see Theorem 2.2). Hence another measure for comparison of the random walks is the asymptotic variance in the Gaussian approximation. The lower the asymptotic variance, the lesser number of samples needed for a certain estimation accuracy. Many authors consider asymptotic variance as the prime parameter to compare RWs. For instance, the authors in [Lee et al. 2012] proved the better performance of non-back tracking random walks compared to the SRW and MH-MCMC with the asymptotic variance. The asymptotic variance is related to the eigenvalues of $\boldsymbol{P}$ as follows,

$$\sigma_f^2 = \sum_{i=2}^{|V|} \frac{1+\lambda_i^{\boldsymbol{P}}}{1-\lambda_i^{\boldsymbol{P}}} \, |\langle f, \boldsymbol{u}_i^{\boldsymbol{P}} \rangle_\pi|^2,$$

where $\langle \boldsymbol{x}, \boldsymbol{y} \rangle_\pi = \sum_{i \in V} \boldsymbol{x}(i)\boldsymbol{y}(i)\boldsymbol{\pi}(i)$ [Brémaud 1999, Chapter 6]. When the interest is in the speed of convergence to equilibrium, then only the second-largest eigenvalue modulus matters. However, if the aim is to compute $\mathbb{E}_\pi[f(X_0)]$ as the ergodic mean $\lim_{n\to\infty} \frac{1}{n}\sum_{k=1}^n f(X_k)$, then all the eigenvalues become significant and this is captured when the quality of the ergodic estimator is measured by the asymptotic variance.

We have briefly mentioned a parameter called extremal index (EI) from extreme value theory at the end of Section A.1.3. EIs of different RWs are studied in detail in Chapter 6 and will be useful for determining some extremal events like the first time to hit a largest degree node. If the purpose of the RWs is to estimate such extremal events, then EI is also a good candidate for comparing RWs.

2.4 Non-asymptotic Unbiasedness of Random Walk Estimators

The ergodic theorem for RWs on connected graphs explains the asymptotic unbiasedness of the estimators. This is definitely useful since it is simple to implement and easy to analyze also. Still, there is a scope for improvement. This follows from the observation that revisits of the RW to the same node act as renewal epochs and the underlying Markov chains regenerates at such instants. The random walk samples between visits to a fixed node are independent conditioned on the returning time instants. This helps us to form a simple estimator as follows. Let

$$\xi_1 := \min\{n \geqslant 1 : X_n = i\}.$$

Now using renewal reward theorem, we have

$$
\begin{aligned}
\mathbb{E}_i\left[\sum_{n=1}^{\xi_1} g(X_n)\right] &= \mathbb{E}[\xi_1]\mathbb{E}_\pi[g] \\
&\overset{(a)}{=} \frac{1}{\boldsymbol{\pi}(i)} \sum_{u\in V} \boldsymbol{\pi}(u) g(u) \\
&= \frac{1}{d_i} \sum_{u\in V} g'(u), \qquad (2.5)
\end{aligned}
$$

where the function $g'(u) := g(u)/d_u$ and in (a) we have used the result that for a finite state ergodic Markov chain $\mathbb{E}[\xi_1] = 1/\boldsymbol{\pi}(i)$. The left-hand side of the above expression can be calculated as the empirical mean from independent runs of revisits to the same node i. This empirical mean multiplied by degree of the anchoring node forms an estimator for $\nu(G) = \sum_{u\in V} g(u)$ and is *non-asymptotically unbiased.*

Overcoming the burn-in time barrier

It is clear from the previous sections that the burn-in period (or mixing time) of random walks discards many initial samples, yet it is necessary for accurate estimation with performance guarantees. The technique mentioned in the previous section with estimator from random walk runs between the renewal epochs does not need to wait until the burn-in period, since one does not need to make the RW independent of the initial distribution unlike in SRW. In fact starting from a fixed node i, the samples are collected and this stops at the time RW returns to the same node i. This technique does not seem to be readily extending to average functions like $\bar{\nu} = |V|^{-1} \sum_{u\in V} g(u)$. Another technique based on reinforcement learning proposed in Chapter 5 addresses this problem, and also evades the burn-in time barrier.

2.5 Mixture of Random Walk and Uniform Node Sampling

The simple random walk has the advantage that it traverses only through the edges of the graph. But many times the designer will have the liberty to collect a few uniform samples from the network. Heuristically, in cases like dumbbell graph random jumps at certain instants improves the mixing time. There exists such variations of the RWs in the literature where the transitions are not restricted to the neighbors of a node. One example is the PageRank algorithm in which the transition probability matrix is modified as, for a constant $0 < c < 1$

$$\widetilde{\boldsymbol{P}} = c\boldsymbol{P} + (1-c)\frac{1}{n}\mathbf{1}\mathbf{1}^\intercal.$$

Here the spectral gap is $\delta = 1 - c$. A modification of the PageRank is rather than fixing c make it depends on the present node degree as,

$$c = \frac{d_{X_n}}{d_{X_n} + \alpha},$$

with a parameter $\alpha \geqslant 0$ [Avrachenkov et al. 2010]. This makes the random walk reversible (unlike PageRank) and it is proven that for a regular graph such a change improves the mixing time.

2.6 Conclusions

This chapter first reviewed some terminology like mixing time, spectral gap, relaxation time, etc., and key theorems associated with random walks on graph. We found how mixing and relaxation times are associated with the ergodic theorem, which is the main idea behind the development and analysis of the estimators of graph functions. The relation of mixing time to the graph structure and the eigenvalues of the transition probability matrix is also detailed later. Some fundamental random walks are explained and a few measures for comparing the random walks are listed. Non-asymptotically unbiased random walk estimator and random walks with a mix of uniform node sampling are explained later.

Chapter 3

Distributed Spectral Decomposition in Networks

This chapter considers the problem of spectral decomposition in networks. We focus on finding the eigenvalues and eigenvectors of symmetric graph matrices. The main interest here is on the distributed implementation and the algorithms are developed based on the assumption that the graph is not completely known. Each node in the network will know only its neighbors and can communicate with only them. In particular, for the distributed implementation we focus on symmetric graph matrices following G (i.e., having 0 at position (i,j) if $(i,j) \notin E$).

For instance, the matrix can be the adjacency matrix $\boldsymbol{A} = [a_{ij}], 1 \leqslant i,j \leqslant |V|$ in which a_{ij} denotes the weight of the edge between the nodes i and j. Due to symmetry the eigenvalues are real and can be ranked in decreasing order as $\lambda_1^{\boldsymbol{A}} \geqslant \lambda_2^{\boldsymbol{A}} \geqslant \ldots \geqslant \lambda_{|V|}^{\boldsymbol{A}}$. We study the largest k eigenvalues $\lambda_1^{\boldsymbol{A}}, \ldots, \lambda_k^{\boldsymbol{A}}$ and the corresponding eigenvectors $\boldsymbol{u}_1^{\boldsymbol{A}}, \ldots, \boldsymbol{u}_k^{\boldsymbol{A}}$.[1] As defined in Chapter 1, we also focus on another well known graph matrix called Laplacian matrix $\boldsymbol{L}$ which is defined as $\boldsymbol{D} - \boldsymbol{A}$. We will use the adjacency matrix to illustrate the ideas in the first part of the chapter and Laplacian for the second part of the chapter.

Organization

In Section 3.2 we review some classical challenges in computing the spectrum of graphs. Section 3.3 presents the central idea of the chapter, complex power iterations, and explains various orders of approximations. In Section 3.4 we present in detail two distributed approaches for network spectral decomposition: complex diffusion when each node can communicate with all its neighbors; complex gossiping when each node receives data from only one neighbor. We also derive the complexity of complex diffusion in the same section. Then, in Section 3.5 we explain that our approach can be efficiently realized using a quantum rank walk based algorithm. In Section 3.6 we analyse the error terms and rate of convergence, and provide recommendations for the choice of parameters. Numerical results presented in Section 3.7 demonstrate the scalability of the approach.

Sections 3.9-3.13 extends the developed algorithms for higher resolution. First in Section 3.9 we detail some of the issues in computation. Section 3.10 explains a mass-spring analogy specific to Laplacian matrix and derive a method to identify

[1] In order to simplify the notation, we drop dependence on the matrix whenever it is evident from the context.

the spectrum. Section 3.11 develops techniques for spectral decomposition based on solving the Schrödinger-type equation efficiently. Section 3.12 details a distributed implementation of these modified algorithms. Section 3.13 contains numerical simulations on networks of different sizes. Section 3.14 concludes the chapter.

3.1 Related Work

We provide a straightforward interpretation of eigenvalue spectrum and eigenvectors in terms of peaks in the frequency domain of complex exponential of $\boldsymbol{A}$ and exploit it for developing distributed algorithms. To the best of our knowledge, the first distributed network spectral decomposition algorithm was proposed in [Kempe & McSherry 2008]. The most challenging part of the algorithm in [Kempe & McSherry 2008] is the distributed orthonormalization at each step of the algorithm. This a difficult operation, which the authors solve by communicating information via random walks. Clearly if the graph has a small conductance (a typical case for many large graphs), this operation will take extremely long time at each step of the algorithm. Our first distributed algorithm based on the diffusion of complex fluid across the network, an implementation of *complex power iterations*, do not require orthonormalization. In [Sahai et al. 2012, Franceschelli et al. 2013] the authors use techniques from signal processing which are in the same spirit of our approach. However their approach needs either to use two time steps or two hop-neighbourhoods for each iteration, while our algorithms work with one time step and one-hop neighbourhood. The approach of [Sahai et al. 2012, Franceschelli et al. 2013] deforms the values of eigenvectors and eigenvalues and the correction needed is not evident. Moreover, since the methods in [Sahai et al. 2012, Franceschelli et al. 2013] are based on classical Fourier transforms, the eigenvalues might not get detected because of spurious peaks in the spectrum. Our approach overcomes this problem by using Gaussian smoothing. Our algorithms can also be immediately implemented via light weight gossiping and random walks with complex rewards. A gossip algorithm based on reinforcement learning was recently introduced in [Borkar et al. 2014], but it computes only the principal eigenvector. From the analysis of our diffusion technique, we observe that our algorithms are scalable in the order of maximum degree. Finally our method has a very interesting relation to the quantum random walks, which with the advancement of quantum computing can make our approaches very efficient.

3.2 Challenges in Classical Techniques

Here we illustrate two problems (among many) faced by existing techniques with the help of two classical algorithms.

The first one, power iteration, consists of computing the iterative power $\boldsymbol{b}_\ell = \boldsymbol{A}^\ell \boldsymbol{b}_0$ for the increasing integer ℓ with $\boldsymbol{b}_0$ as an initial vector. Using the spectral decomposition of $\boldsymbol{A}$, we have

$$\boldsymbol{A}^{\ell} = \sum_j \lambda_j^{\ell} \boldsymbol{u}_j \boldsymbol{u}_j^{\intercal} \ .$$

We adopt the convention $\|\boldsymbol{u}_j\| = 1$. Depending of λ_1 being greater or smaller than one, the iteration $\boldsymbol{b}_{\ell} = \boldsymbol{A}^{\ell}\boldsymbol{b}_0$ for $\ell \geqslant 1$ will exponentially decrease or increase without proper step normalization. The normalization introduced is

$$\boldsymbol{b}_{\ell+1} = \frac{1}{\|\boldsymbol{b}_{\ell}\|} \boldsymbol{A}\boldsymbol{b}_{\ell} \ . \tag{3.1}$$

Notice that $\boldsymbol{b}_{\ell}$ converges to $\lambda_1 \mathbf{u}_1$ when $\ell \to \infty$. The problem is that this method cannot be readily applied to the search of the other eigenvalues because the first eigenvalue screens the exponentially decreasing secondary eigenvalues: $\boldsymbol{b}_{\ell} = \lambda_1 \mathbf{u}_1 + \mathcal{O}\left((\frac{\lambda_2}{\lambda_1})^{\ell}\right)$.

In order to compute the other eigenvalues one can use the inverse iteration methods based on the formula

$$\boldsymbol{b}_{\ell+1} = \frac{1}{\|\boldsymbol{b}_{\ell}\|} (\boldsymbol{A} - \mu \boldsymbol{I})^{-1} \boldsymbol{b}_{\ell}, \tag{3.2}$$

for an arbitrary real number $\mu < \lambda_1$. The iteration will converge to $\frac{1}{\lambda_j - \mu}\boldsymbol{u}_j$ where j is the index of the eigenvalue closest to μ. The search consists of approaching the eigenvalue by tuning the parameter μ. The difficulty of the method is in computing the inverse matrix (or solution of the linear system) for each selected values μ which is computationally costly. Furthermore, the use of normalization at each iterative step, in (3.1) as well as in (3.2), will make it difficult an adaptation to the distributed context, i.e., the normalization requires frequent pausing of the iteration in order to compute and disseminate the normalization factor.

In the next section we propose a new method called complex power iterations. From an initial vector $\boldsymbol{b}_0$ and given integer k, the method will return the k first eigenvalues $\lambda_1, \ldots, \lambda_k$ and the vectors $(\boldsymbol{b}_0^{\intercal}\boldsymbol{u}_j)\boldsymbol{u}_j$ for $j = 1, \ldots, k$. Notice that the $\boldsymbol{u}_j$'s can be retrieved via normalization, but it will turn out that this is an unnecessary step.

3.3 Complex Power Iterations

Now we introduce the main idea in the chapter, the complex power iterations, which we use to compute the spectrum of the matrix of a network in an efficient way. We consider an undirected graph. The important notation used in this chapter are listed in Frequently used Notation (p. ix).

We derive a technique that is analogous to the frequency spectrum in the time-frequency analysis and apply it to graph spectrum analysis. In this perspective, the domain of eigenvalues corresponds to the frequency domain.

From the eigendecomposition of the symmetric matrix $\boldsymbol{A}$, $\boldsymbol{A} = \sum_j \lambda_j \boldsymbol{u}_j \boldsymbol{u}_j^{\intercal}$, we have $e^{i\boldsymbol{A}t} = \sum_j e^{it\lambda_j} \boldsymbol{u}_j \boldsymbol{u}_j^{\intercal}$. Notice that the function $e^{i\boldsymbol{A}t}$ is pseudo periodic and

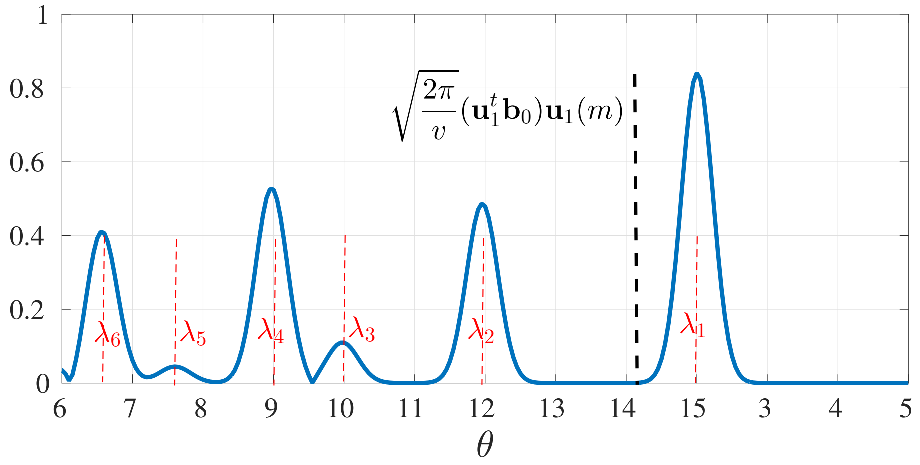


Figure 3.1: Spectral plot at node m.

its harmonics correspond exactly to the spectrum of the matrix. The advantage of complex exponential is that the whole spectrum can be recovered via the classical Fourier transform, contrary to the expression $e^{\boldsymbol{A}t}$ (dropping the imaginary unit i) where the effect of the other eigenvalues λ_j for $j > 1$ decays exponentially when $t \to \infty$. Indeed, we formally have

$$\frac{1}{2\pi}\int_{-\infty}^{+\infty} e^{i\boldsymbol{A}t}e^{-it\theta}dt = \sum_{j=1}^{|V|}\delta_{\lambda_j}(\theta)\boldsymbol{u}_j\boldsymbol{u}_j^\intercal \ , \tag{3.3}$$

with δ_{λ_j} being the Dirac function translated of the quantity λ_j. However this expression is not easy to handle numerically because any truncated or discretization version of the integral will generate too large fake harmonic oscillations that will hide the Dirac peaks (we show an example of this in Section 3.7). To overcome this problem we use the spectral smoothing via convolution with a Gaussian function of variance $v > 0$:

$$\frac{1}{2\pi}\int_{-\infty}^{+\infty} e^{i\boldsymbol{A}t}e^{-t^2v/2}e^{-it\theta}dt = \sum_{j=1}^{|V|}\frac{1}{\sqrt{2\pi v}}\exp(-\frac{(\lambda_j-\theta)^2}{2v})\boldsymbol{u}_j\boldsymbol{u}_j^\intercal \ , \tag{3.4}$$

Notice that the above expression converges to (3.3) when $v \to 0$. In order to ease visualization, a scalar product is taken with an initial vector $\boldsymbol{b}_0$. Figure 3.1 shows a sample plot produced from (3.4) at some node m, by varying θ. The detection of the eigenvalues corresponds to locating the peaks and the quantities $\sqrt{\frac{2\pi}{v}}(\boldsymbol{b}_0^\intercal\boldsymbol{u}_j)\boldsymbol{u}_j(m)$ corresponds to the values at these peaks as we will see later.

The key for the computation of (3.4) is the determination of the factor $e^{i\boldsymbol{A}t}$ which does not come naturally from the expression of matrix $\boldsymbol{A}$. We fix a number $\varepsilon > 0$ and we use the discretization $e^{i\boldsymbol{A}\ell\varepsilon} = (\boldsymbol{I} + i\varepsilon\boldsymbol{A})^\ell(1 + \mathcal{O}(\varepsilon^2\ell))$, where I is the identity matrix, for the calculation of left-hand side in (3.4),

$$\begin{aligned}&\int_{-\infty}^{+\infty} e^{i\boldsymbol{A}t}e^{-t^2v/2}e^{-it\theta}dt\\ &= \varepsilon\Re\left(\boldsymbol{I}+2\sum_{\ell=1}^{\ell_{\max}}(\boldsymbol{I}+i\varepsilon\boldsymbol{A})^{\ell}e^{-i\ell\varepsilon\theta}e^{-\ell^2\varepsilon^2v/2}\right)+\mathcal{O}(\varepsilon^2\ell_{\max})\end{aligned} \tag{3.5}$$

Here $\Re(x)$ denote real part of complex number x and $\ell_{\max}$ is the maximum number of steps used to approximate the left-hand side. The quantity $\ell\varepsilon$ in the sum plays the role of variable t in the integral. Applying the expression to an initial vector $\boldsymbol{b}_0$, we define,

$$\boldsymbol{f}_\theta = \varepsilon\Re\Big(\boldsymbol{b}_0+2\sum_{\ell=1}^{\ell_{\max}} e^{-i\ell\varepsilon\theta}e^{-\ell^2\varepsilon^2v/2}\boldsymbol{x}_\ell\Big), \tag{3.6}$$

where $\boldsymbol{x}_\ell$ is used to approximate $e^{i\varepsilon\ell\boldsymbol{A}}\boldsymbol{b}_0$. For instance, in (3.5), $(I+i\varepsilon\boldsymbol{A})^\ell$ is taken as an estimate of $e^{i\varepsilon\ell\boldsymbol{A}}$.

We notice that the expression of $\boldsymbol{f}_\theta$ does not use any discretisation over the θ variable or on the v variable and turns out to be analytical functions of these variables. Therefore the search of peaks corresponding to the eigenvalues turns out to be finding the zeroes of the derivative of $\boldsymbol{f}_\theta$ under the condition that $\boldsymbol{f}_\theta$ is above a certain threshold.

This process and the way to tune the best values of ε, v and $\ell_{\max}$ will be discussed in Section 3.6.3. In the next section we refine some higher order approximation of $e^{i\varepsilon\ell\boldsymbol{A}}$ which can be used to obtain more accurate expressions.

3.3.1 Higher order approximations

The approximation $\boldsymbol{x}_\ell$ of $e^{i\ell\varepsilon\boldsymbol{A}}\boldsymbol{b}_0$ in (3.6) can be made more accurate by using Runge-Kutta (RK) methods. Indeed, we have $\boldsymbol{x}_{\ell+1}=(\boldsymbol{I}+i\varepsilon\boldsymbol{A})\boldsymbol{x}_\ell$. We also notice that $e^{i\mathbf{A}t}\boldsymbol{b}_0$ is the solution of the differential equation $\dot{\boldsymbol{x}}=(i\mathbf{A})\boldsymbol{x}$ with initial condition $\boldsymbol{x}(0)=\boldsymbol{b}_0$. We use Runge-Kutta (RK) methods [Tenenbaum & Pollard 1985] to solve this differential equation numerically. With $\boldsymbol{x}(\ell\varepsilon)$ approximated by $\boldsymbol{x}_\ell$, the iteration in such a method can be defined as follows,

$$\boldsymbol{x}_{\ell+1}=\boldsymbol{x}_\ell+\frac{1}{6}k_1+\frac{1}{3}k_2+\frac{1}{3}k_3+\frac{1}{6}k_4, \tag{3.7}$$

where $\boldsymbol{x}_0=\boldsymbol{b}_0$, $k_1=\varepsilon(i\mathbf{A}\boldsymbol{x}_\ell)$, $k_2=\varepsilon i\mathbf{A}(\boldsymbol{x}_\ell+1/2\,k_1)$, $k_3=\varepsilon i\mathbf{A}(\boldsymbol{x}_\ell+1/2\,k_2)$ and $k_4=\varepsilon i\mathbf{A}(\boldsymbol{x}_\ell+k_3)$. It is observed that (3.7) is equivalent to $\boldsymbol{x}_\ell=\Big(\mathbf{I}+(i\varepsilon\boldsymbol{A})+\ldots+\frac{(i\varepsilon\boldsymbol{A})^4}{4!}\Big)^\ell\boldsymbol{b}_0$. This is the RK method of order-4. This equivalence can be easily generalized to any order r RK methods as,

$$\boldsymbol{x}_\ell=\left(\sum_{j=0}^{r}\frac{(i\varepsilon\boldsymbol{A})^j}{j!}\right)^\ell\boldsymbol{b}_0. \tag{3.8}$$

In this work, we use order $r=1,2$ and 4.

3.4 Complex Diffusion

The previous analysis is based on a centralised approach where we can manage to make matrix multiplications. The real challenge in the networking context is to make the computations distributed. Thus, in order to compute (3.6), we propose and compare the following three techniques:

(i) **Centralized approach:** Here we assume that the adjacency matrix $\boldsymbol{A}$ is completely known to a centralized unit. We use order-1 (3.5), order-2 or order-4 (3.7) technique to compute the approximation.

(ii) **Complex diffusion:** It is a distributed and asynchronous approach in which only local information is available at each node, which is the list of its neighbors. Here, each node communicates with all its neighbors at each time epoch.

(iii) **Monte Carlo techniques:** This is also a distributed technique with only local information, but with much reduced complexity than Complex diffusion as each node communicates with only one neighbor. Monte Carlo techniques can be implemented either using Monte Carlo Gossiping or using parallel random walks.

The matrix $\boldsymbol{X}$ (size $|V| \times (\ell_{\max}+1)$) approximates $e^{i\ell\varepsilon \boldsymbol{A}}\boldsymbol{b}_0$ for $0 \leqslant \ell \leqslant \ell_{\max}$, in the following sense.

$$\begin{aligned} \boldsymbol{X} &= \begin{bmatrix} \boldsymbol{x}_0 & \boldsymbol{x}_1 & \ldots & \boldsymbol{x}_{\ell_{\max}} \end{bmatrix} \\ &\approx \begin{bmatrix} \boldsymbol{b}_0 & e^{i\varepsilon \boldsymbol{A}}\boldsymbol{b}_0 & \ldots & e^{i\ell_{\max}\varepsilon \boldsymbol{A}}\boldsymbol{b}_0 \end{bmatrix}. \end{aligned}$$

The above three methods employ different techniques to compute $\boldsymbol{X}$. At any node m, once the corresponding row in $\boldsymbol{X}$ is computed, then $\boldsymbol{f}_\theta(m)$ (cf. (3.6)) can be calculated independent of other nodes, and thus spectral plot and the dominant eigenvalues and eigenvectors are obtained.

3.4.1 Complex diffusion

The key of the computation in (3.6) is the calculation of the sequence $\boldsymbol{x}_\ell$ or the associated generating polynomial $\boldsymbol{x}(z) = \sum_{\ell=0}^{\ell_{\max}} z^\ell \boldsymbol{x}_\ell$. Complex Diffusion uses the idea of fluid diffusion in networks to compute the coefficients of this polynomial in z. The algorithms proposed in this section are distributed in the sense that each node needs only to know and to communicate with its neighbors. They can be made asynchronous as well since there is no central clock to control the fusion-diffusion process.

We first consider the complex diffusion based on the order-1 approximation, i.e., $\boldsymbol{x}_\ell = (\boldsymbol{I} + i\varepsilon \boldsymbol{A})^\ell \boldsymbol{b}_0$. For order-1 calculations, the node m will start with an initial fluid $\boldsymbol{b}_0(m)$ and a copy of this fluid is diffused to all of its neighbors with weight $i\varepsilon a_{m,h}, h \in \mathcal{N}(m)$. A copy is also diffused to itself with weight $1 + i\varepsilon a_{mm}$. The technique is detailed in Algorithm 3.4.1. At each node, we compute the polynomial

$x(z)$ which corresponds to the equivalent row in $\boldsymbol{X}$. Then vector $\boldsymbol{x}(z)$ is made of the polynomials $x(z)$ computed on each node m. In the algorithm, the procedure Send(h, f) transmits fluid f to node h and Receive(h) collects fluid from h.

Algorithm 3.4.1: ComplexDiffusion$(m, \boldsymbol{b}_0)$

```
$C(z) \leftarrow \boldsymbol{b}_0(m)$
$\ell \leftarrow 0$
while ($\ell \leqslant \ell_{\max}$)
do
    $x(z) \leftarrow x(z) + C(z)$
    for each $h \in \mathcal{N}(j)$
        do {Send$(h, a_{m,h}\varepsilon i C)$
    $C(z) \leftarrow (1 + i\varepsilon a_{j,j})C(z)z$
    for each $h \in \mathcal{N}(j)$
        do {$C(z) \leftarrow C(z) + z$Receive$(h)$
    $\ell \leftarrow \ell + 1$
return ($x(z)$)
```

The total number of fluid diffusion-fusion cycles at each node should be $\ell_{\max}$ in case the diffusions are synchronised. For asynchronous diffusions the diffusion will stop when all quantities C have only monomials of degree larger than $\ell_{\max}$, thus equal to zero after truncation. This will occur when all paths of length $\ell_{\max}$ have been processed in sequence. This can be easily detected if we assume a maximum time $t_{\max}$ for a diffusion-fusion cycle on each node, the process should stop after any laps of duration $t_{\max}\ell_{\max}$ with no diffusion-fusion.

At first sight, the quantities $x(z)$'s would need to be collected only at the end of the parallel computations. In fact, even this is not needed since the computation of $\boldsymbol{f}_\theta(m)$ and the peak detection can be made locally as long as the initial vector $\boldsymbol{b}_0$ is not orthogonal to the $\boldsymbol{u}_j$, and the quantity $(\boldsymbol{b}_0^\intercal \boldsymbol{u}_j)\boldsymbol{u}_j(m)$ be returned.

The polynomial technique explained in Algorithm 3.4.1 can also be implemented in vector form. We can extend the technique to higher order approximations. The pseudo-code in Algorithm 3.4.2 implements the order-2 complex diffusion. The use of parameter $C_2(z)$ is the artefact for the diffusion of matrix $(i\varepsilon\boldsymbol{A})^2$. Indeed, the fluid must be retransmitted towards a relay before being added to $x(z)$. This is the reason why the number of iterations must be raised to $2\ell_{\max}$. The generalisation to a diffusion of order-r, is straightforward since it consists of the diffusion of matrix $(i\varepsilon\boldsymbol{A})^r$. To this end we add additional parameters $C_2(z)$ by a vector $C_r(z)$ with $r-1$ components $C_r^2, \dots C_r^r$ and the procedure Send transmits $i\varepsilon a_{m,h}(C(z) + C_r^r(z))$ and Send-r consists of transmitting the vector $[C(z), C_r^2(z), \dots, C_r^{r-1}(z)]$.

In parallel each node can compute their function $\boldsymbol{f}(\theta)$ (cf. (3.6)) and detect the peaks. When a node has detected the values λ_ℓ for those peaks, with $\ell = 1, \dots, k$

Algorithm 3.4.2: DIFFUSIONORDER2$(m, \boldsymbol{b}_0)$

```
$C(z) \leftarrow \boldsymbol{b}_0(m)$
$C_2(z) \leftarrow 0$
$\ell \leftarrow 0$
while $(\ell \leqslant 2\ell_{\max})$
  do
    $x(z) \leftarrow x(z) + C(z)$
    for each $h \in \mathcal{N}(m)$
      do
        SEND$(h, a_{m,h}\varepsilon i(C(z) + \frac{1}{2}C_2(z)))$
        SEND-2$(h, a_{m,h}\varepsilon i C(z))$
    $C_2(z) \leftarrow i\varepsilon a_{m,m} C(z)$
    $C(z) \leftarrow (1 + i\varepsilon a_{m,m}) z C(z)$
    for each $h \in \mathcal{N}(m)$
      do
        $C(z) \leftarrow C(z) + z$RECEIVE$(h)$
        $C_2(z) \leftarrow C_2(z) +$ RECEIVE-2$(h)$
    $\ell \leftarrow \ell + 1$
return $(x(z))$
```

and the corresponding values $\boldsymbol{f}(\lambda_\ell)$ are broad-casted in a packet with a specific sequence number. The packet will be repeated all along the network. At the end of the process, all the nodes in the network can reconstitute the vectors $\boldsymbol{f}(\lambda_\ell)$ and perform any algorithm based on eigenvectors. It is very possible that the values of the λ_ℓ will not perfectly match from one node to another, but this is not crucial since, in most cases, the structure of the vectors $\boldsymbol{f}(\lambda_\ell)$ can accept some inaccuracy.

3.4.2 Complexity of the algorithm

In the following, we provide complexity calculations for the inverse iterations (see Section 3.2) and for the complex diffusion of order-1. The power iterations are not considered here as it allows only the computation of the principal eigenvalue and its eigenvector.

The inverse iterations

This scheme is difficult to operate in a distributed manner, since the matrix $(\mathbf{A} - \mu\mathbf{I})^{-1}$ is not supported by the graph itself (it will have coefficients over non existing edges). The cost of inverting the matrix is $|V|^3$, times the number of different μ's needed, and the computation is done at a central node. Then the coefficients of the inverse matrix must be spread over the whole network which will lead to $|E|$ repetitions of the same packets if done by pure diffusion, or $|V|$ repetitions if a spanning tree is used (but this requires some knowledge of the topology).

Each iteration proceeds in two steps: (1) every node diffuses the coefficients of $\boldsymbol{b}_{k+1} = (\mathbf{A} - \mu\mathbf{I})^{-1}\boldsymbol{b}_k$ to all the other nodes, thus requires $|V||E|$ packets by pure diffusion or $|V|^2$ packets via a spanning tree. (2) a central node collects and computes $\|\boldsymbol{b}_{k+1}\|$, and forwards the results to all the other nodes. Such an operation costs $|E|$ packets (or $|V|$ via spanning tree). This marks a synchronization point which will trigger the next iteration.

Assuming $\ell_{\max}$ iterations, we obtain the net computational cost as $|E||V|^2 + (|V||E| + |E|)\ell_{\max}$ (in terms of packet exchanged). If we consider the delay, many packets may fly in parallel, and a diffusion will have a delay proportional to $\operatorname{diam}(G)$, the diameter of the graph. Therefore the net delay will be of the order of $\operatorname{diam}(G) + 2\operatorname{diam}(G)\ell_{\max}$.

The complex diffusion of order-1

The iterations in order-1 complex diffusion do not need any synchronization. An iteration consists of a packet on each edge, and thus a total of $|E|$ packets sent in parallel. This leads to the delay proportional to unit time instant, since transmission only happens to one hop neighborhood.

After $\ell_{\max}$ iterations, the results are sent to a central collector node, and it requires $|V||E|$ packets via pure diffusion or $|V|^2$ via a spanning tree. If all packets fly in parallel, then the delay will be $\operatorname{diam}(G)$.

Thus the total number of packets exchanged is $|E|\ell_{\max} + |V||E|$ and the total delay is $\ell_{\max} + \operatorname{diam}(G)$. The gains in the complex diffusion compared to the inverse iterations is at least, per iteration, $|V|$ in the net computational cost and $\operatorname{diam}(G)$ in the net delay. Note that the net cost and delay expressions in the complex iterations are derived for only one μ, and at least $|V|$ μ's might be required to distinguish between different eigenvalues. This is not taken into account in the calculation and will further worsen the performance of the inverse iterations.

3.4.3 Complex gossiping

In the order-1 computation (3.5), the terms $(I + i\varepsilon\boldsymbol{A})^\ell\boldsymbol{b}_0$ for $0 \leqslant \ell \leqslant \ell_{\max}$ can also be estimated by the following Monte Carlo approach. We have

$$\boldsymbol{x}_{k+1} = (\boldsymbol{I} + i\varepsilon\boldsymbol{A})\boldsymbol{x}_k, \quad \boldsymbol{x}_0 = \boldsymbol{b}_0.$$

Then

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + i\varepsilon\boldsymbol{D}\boldsymbol{P}\boldsymbol{x}_k,$$

where $\boldsymbol{D}$ is the diagonal matrix with entries as the degrees of the nodes $(d_1, \ldots, d_n)$, and $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$ is the transition probability matrix of a random walk on graph defined by matrix $\boldsymbol{A}$. For m and h nodes in the graph, we denote p_{mh} the probability transition from m to h, namely equal to $\frac{a_{m,h}}{d_m}$. We have the identity on the mth component of $\boldsymbol{x}_{k+1}$,

$$\boldsymbol{x}_{k+1}(m) = \boldsymbol{x}_k(m) + i\varepsilon d_m\mathbb{E}[\boldsymbol{x}_k(\xi_m)], \tag{3.9}$$

with ξ_m as a randomly picked neighbour of node-m. Similar local updating rule applies for other nodes as well. The expectation in (3.9) can be calculated using Monte Carlo approach. Interestingly we observe in simulation results that a small number of iterations provides a good accuracy for the Monte Carlo simulation. The algorithm is iterated several times and then averaged in order to smooth the variations due to the random selection of neighbors. The algorithm is as follows:

Algorithm 3.4.3: ComplexGossiping$(m, \boldsymbol{b}_0)$

```
$\ell \leftarrow 1$
$x(z) \leftarrow \boldsymbol{b}_0(m)$
while $(\ell \leqslant \ell_{\max})$
  do
    for each $j \in \mathcal{N}(m)$
      do
        if RequestWaitingFrom$(j)$
          then
            $\ell' \leftarrow$ RequestLevel$(j)$
            if $(\ell' < \ell)$ then
              Send$(j, i\varepsilon$ Coeff$(x(z), z^{\ell'}))$
    $\xi_m \leftarrow$ RandomNeighbor$(m)$
    SendRequest$(\xi_m, \ell - 1)$
    $x(z) \leftarrow x(z)+$
        $z^{\ell}$ Coeff$(x(z), z^{\ell-1}) + D_m z^{\ell}$Receive$(\xi_m)$
    $\ell \leftarrow \ell + 1$
return $(x(z))$
```

The procedure Coeff$(x(z), z^{\ell})$ returns coefficient of the term z^{ℓ} in $x(z)$. Requestwaitingfrom(j) is a boolean valued procedure indicating if a request is waiting from node j, Requestlevel(j) is the degree of the coefficient required by node j and SendRequest(j, ℓ) is the procedure which sends to node j the request to fetch the coefficient of degree ℓ. Notice that with the procedure SendRequest, the local process will wait until the neighbor ξ_m will respond. Though this will introduce delays, it is limited to waiting time of a single neighbor to deliver. But such a scheme will avoid the use of a synchronous clock in the system.

The gossiping algorithm introduced here is a variant of the diffusion algorithms mentioned in the previous section, i.e., instead of taking fluid from all the neighbors, the gossiping technique collects the fluid from only one random neighbor. The algorithm can also be extended to order-2 and order-4.

3.5 Quantum Random Walk Techniques

We have described in the previous section methods to solve a discretisation of equation

$$\frac{\partial}{\partial t}\boldsymbol{b}_t = i\boldsymbol{A}\boldsymbol{b}_t.$$

This equation is very similar to the classical Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\boldsymbol{\Psi}_t = \mathcal{H}\boldsymbol{\Psi}_t,$$

where $\boldsymbol{\Psi}_t$ is the wave function represented by a vector of complex numbers, $\hbar$ is the Planck constant and $\mathcal{H}$ is the Hamiltonian of the system (which is a Hermitian matrix). If $\mathcal{H}$ is formed from graph matrices like $\boldsymbol{A}$, we have the wave function of a particle wandering on a network, which is called quantum random walk. In this section we will first show that the evolution of the wave function of a quantum random walk can be simulated via several parallel classical random walks with complex reward and use it in order to extract the spectrum information. Then we propose a pure quantum random walk algorithm.

3.5.1 Quantum random walk algorithm via classical random walks

This is a distributed technique in which the nodes require only local information, but with reduced complexity compared to Complex gossiping as only one node communicates with another neighbor node at each cycle, for each random walks.

Assume we consider order-1 approximation. At each iteration, we run the following algorithm:

1. Each node-m has a vector of length $\ell_{\max} + 1$ (mth row of $\boldsymbol{X}$) with $\boldsymbol{x}_\ell(m)$ representing mth entry of $(\boldsymbol{I} + i\varepsilon\boldsymbol{A})^\ell\boldsymbol{b}_0$.

2. A set of classical random walks start from each node initially. All the random walks associated with node-m have the initial fluid $\boldsymbol{b}_0(m)$.

3. All the random walks start moving to one of their neighbors, which is selected uniformly at random.

4. At time step $k > 2$, at each node m, only the first arrived random walk wins and the fluid carried by this random walk (F) will be used for updating $\boldsymbol{x}_k(m)$. The fluid update will happen at node-m when it also receives the fluid from its own previous level $\boldsymbol{x}_{k-1}(m)$. Then

$$\boldsymbol{x}_k(m) = \boldsymbol{x}_{k-1}(m) + i\varepsilon d_m F.$$

   After the update of $\boldsymbol{x}_k(m)$, all the random walks at node-m (the one which won and the ones which lost) will update their fluid as $\boldsymbol{x}_k(m)$. In case no random walks arrive at a particular node at level k after a certain time, it will take the fluid from its previous level $k - 1$ only.

5. The random walks keep moving and all of them stop when the time step k reaches $\ell_{\max}$.

Like the gossiping technique, the above algorithm is also iterated several times to get the desired convergence. The pseudocode of a more generalized technique is given in Algorithm 3.5.1.

Algorithm 3.5.1: ParallelRandomWalk($m, \boldsymbol{b}_0$)

```
$\ell \leftarrow 0$
$x(z) \leftarrow \boldsymbol{b}_0(m)$
while $(\ell \leqslant \ell_{\max})$
  do
    for each $M \in$ MoveWaiting()
      do
        $\ell' \leftarrow$ MoveLevel($M$)
        if $(\ell' > \ell)$ then
          for $\ell'' \leftarrow \ell + 1$ to $\ell'$
            do {$x(z) \leftarrow x(z) + z^{\ell''}x(z)$
          $\ell \leftarrow \ell'$
          $x(z) \leftarrow x(z) + z^{\ell} d_m$MoveValue($M$)
        $\xi_m \leftarrow$ RandomNeighbor($m$)
        SendMove($\xi_m, \ell + 1, \varepsilon i$Coeff($x(z), z^{\ell}$))
return $(x(z))$
```

The procedure MoveWaiting() is the set of the notification of random walk moves to node m, MoveLevel(M) is the level of the move notification M, and SendMove(j, d, c) is the process of moving random walk to the level d at node j carrying the value c, and MoveValue(M) is the value carried the random walk notification M.

3.5.2 Pure quantum random walk algorithm

In this section, we elaborate the connection of our technique to quantum computing. We make use of quantum random walks (QRW) on graphs to massively distribute our spectrum computations. Compared to the classical random walks, in which the walker can exist in only one state at a time, a QRW moves simultaneously over all the states by exploiting the idea of superposition in quantum mechanical systems [Venegas-Andraca 2008]. The quantum mechanical system we assume to perform the spectral decomposition is described as follows.

We focus on continuous time QRW on a graph in which the position of the walker depends on the Hamiltonian $\mathcal{H}$ which is taken as $\mathcal{H} = \boldsymbol{A} + \Delta\boldsymbol{I}$ where Δ is the maximum degree. The translation by Δ is necessary in order to make the energy values positive and will only cost a translation of the spectrum.

The walker is represented by a qubit made out of a chain of Γ atoms where each of them is spin oriented, either up (1) or down (0). Therefore the qubit has a capacity of Γ bits and can describe all the 2^Γ binary codewords of length Γ, corresponding to integers between 0 and $\ell_{\max} - 1$, with $\ell_{\max} = 2^\Gamma$. For $0 \leqslant k \leqslant \ell_{\max} - 1$, we denote $|k\rangle$ the state of the qubit corresponding to the integer k. At initialization, state of the qubit is uniform on all codewords: $(1/\sqrt{\ell_{\max}}) \sum_{k=0}^{\ell_{\max}-1} |k\rangle$.

We consider a *splitting chain* made of Γ polarized gates such that the state $|k\rangle$ is delayed by $k\varepsilon$ time units. To achieve this, for $0 \leqslant r \leqslant \Gamma$ the rth gate let the qubit with spin 0 pass or delay the spin 1 by $2^r\varepsilon$ via a delay line. At the end of *splitting chain*, the qubit is *injected* in the graph on each node ℓ. The wave function $\boldsymbol{\Psi}_t^{\ell_{\max}}$ of the walker at time t, a complex valued vector on the vertices of the graph, formed from such a process satisfies

$$\boldsymbol{\Psi}_t^{\ell_{\max}} = \frac{1}{\sqrt{\ell_{\max}}} \sum_{k=0}^{\ell_{\max}-1} e^{i(t-k\varepsilon)\mathcal{H}} \boldsymbol{\Psi}_0 |k\rangle. \tag{3.10}$$

Here $\boldsymbol{\Psi}_0$ is the wave function of the walker when it is inserted in the graph, for instance when the walker is introduced on node ℓ: $\boldsymbol{\Psi}_0(m) = \delta_\ell(m)$ for any node m. At time $t \geqslant \varepsilon\ell_{\max}$, we take the qubits on any node m, $\boldsymbol{\Psi}_t^{\ell_{\max}}(m)$. We apply on it the quantum Fourier transform (QFT) as described in the Shor's algorithm [Shor 1997]. Essentially this implements a discrete Fourier transform (DFT) approximation of the continuous Fourier transform in (3.3). The QFT outputs the DFT coefficients $\{y_k\}$ as $\sum_{k=0}^{\ell_{\max}-1} y_k|k\rangle$. During measurement of the QRW, kth index is obtained with probability $|y_k|^2$, and this will be an eigenvalue point shifted by Δ (along with appropriate scaling of frequencies in discrete domain to continuous domain). Thus multiple run-measurement process of the QRW produces the different eigenvalues. The empirical probability of $\lambda_j + \Delta$ can be calculated via measurements and will be proportional to $|\boldsymbol{u}_j(m)|^2$.

The *splitting chain* technique with *injection* into the original graph can be further modified in order to introduce the Gaussian smoothing (3.4) which improves the accuracy of the spectral decomposition, but is not described here. Moreover, $\boldsymbol{\Psi}_0$ could be $(1/\sqrt{2})(\delta_{\ell'}(m) + \delta_\ell(m))$ so that $\operatorname{sign}(\boldsymbol{u}_j(\ell)\boldsymbol{u}_j(\ell'))$ can be revealed.

3.6 Parameter Analysis and Tuning

3.6.1 Rate of Convergence

There are three factors governing the convergence rate:

1. Riemann integral approximation to left-hand side of the integral (3.4):

$$\begin{aligned} &\varepsilon\Re\left(\boldsymbol{I} + 2\sum_{\ell=1}^{\ell_{\max}} e^{i\ell\varepsilon\boldsymbol{A}}\boldsymbol{b}_0 e^{-i\ell\varepsilon\theta} e^{-\ell^2\varepsilon^2 v/2}\right) \\ &= \int_{-T}^{+T} e^{i\boldsymbol{A}t}\boldsymbol{b}_0 e^{-t^2 v/2} e^{-it\theta} dt + \mathcal{O}\left(\lambda_1\varepsilon^2\ell_{\max}\|\boldsymbol{b}_0\|\right), \end{aligned} \tag{3.11}$$

where $T = \varepsilon\ell_{\max}$. The factor $\lambda_1\|\boldsymbol{b}_0\|$ is an upperbound of the derivative of $e^{it\boldsymbol{A}}\boldsymbol{b}_0$. Notice that λ_1 can in turn be upper bounded by Δ the maximum weighted degree of the graph.

2. Approximating $e^{i\ell\varepsilon\boldsymbol{A}}$ by r-order Runge-Kutta method (with the equivalent expression (3.8)), we get

$$\begin{aligned} &\varepsilon\Re\left(I + 2\sum_{\ell=1}^{\ell_{\max}} e^{-i\ell\varepsilon\theta}e^{-\ell^2\varepsilon^2 v/2}\boldsymbol{x}_l\boldsymbol{b}_0\right) \\ = \quad &\varepsilon\Re\left(\boldsymbol{I} + 2\sum_{\ell=1}^{\ell_{\max}} e^{i\ell\varepsilon\boldsymbol{A}}\boldsymbol{b}_0 e^{-i\ell\varepsilon\theta}e^{-\ell^2\varepsilon^2 v/2}\right) + \mathcal{O}\left(\lambda_1\varepsilon^{r+2}(\ell_{\max})^2\|\boldsymbol{b}_0\|\right). \end{aligned}$$

3. Error in truncated integral:

$$\begin{aligned} &\int_{-T}^{T} e^{i\boldsymbol{A}t}\boldsymbol{b}_0 e^{-t^2 v/2}e^{-it\theta}dt \\ &= \int_{-\infty}^{\infty} e^{i\boldsymbol{A}t}\boldsymbol{b}_0 e^{-t^2 v/2}e^{-it\theta}dt + \mathcal{O}\left(\sqrt{\frac{2\pi}{v}}\mathrm{erf}\left(\sqrt{\frac{v}{2}}\varepsilon\ell_{\max}\right)\|\boldsymbol{b}_0\|\right) \end{aligned}$$

where $\mathrm{erf}(x)$ indicates the Gaussian error function.

It can be seen that convergence rate in (3.11) dominates. In addition, we should have $\varepsilon\ell_{\max}$ large while $\varepsilon^2\ell_{\max}$ is small.

3.6.2 Choice of initial vector and algebraic multiplicity

In order to compute the approximation (3.6) in a distributed manner in the network itself, each node selects its own component of the initial vector $\boldsymbol{b}_0$. The components could be all equal to 1, $\boldsymbol{b}_0 = \mathbf{1}$, but in this case it may be orthogonal to eigenvectors. Indeed if the graph is regular then $\mathbf{1}$ is colinear to the main eigenvector, and therefore orthogonal to the other eigenvectors. To circumvent this problem, each node can randomly select a component so that the probability it results into an orthogonal vector be negligible.

Another interesting option is to select $\boldsymbol{b}_0$ as a vector of i.i.d. Gaussian random variables with zero mean and variance w. In this case

$$\mathbb{E}[\boldsymbol{b}_0^\intercal\boldsymbol{f}_\theta] = w\sum_{j=1}^{|V|}\sqrt{\frac{2\pi}{v}}\exp(-\frac{(\lambda_j-\theta)^2}{2v}).$$

The left-hand side of the above expression can be calculated by Monte Carlo techniques and this will give equal peaks for all the eigenvalues. Hence, the eigenvalues and subsequently the eigenvector components can be deduced with very good accuracy. We call this technique *trace-technique*, since this method indeed is like taking trace of the original approximation matrix, right-hand side of (3.4).

In case of algebraic multiplicity k of a particular eigenvalue λ, *trace-technique* will give the peaks approximately as $kw\sqrt{2\pi/v}$ and $\mathbb{E}[\boldsymbol{b}_0\boldsymbol{f}_{\theta=\lambda}^\intercal]$ will be the projection matrix on eigenspace of λ, i.e., $\mathbb{E}[\boldsymbol{b}_0\boldsymbol{f}_{\theta=\lambda}^\intercal] = \sqrt{\frac{2\pi}{v}}w\sum_{\ell=1}^{k}\boldsymbol{u}_\ell\boldsymbol{u}_\ell^\intercal$.

3.6.3 Choice of parameters

Parameter v

The selection of v is governed by the fact that we need to discern distinct eigenvalues by locating the peaks in the spectral plot. When we need to locate top-k eigenvalues, with 99.7% of the Gaussian areas not overlapping, $6v < \min_{1 \leqslant i \leqslant k-1} |\lambda_i - \lambda_{i+1}|$. In general for a large graph, a sufficiently lower value of v will be enough to distinguish between the peaks. For larger $\varepsilon \ell_{\max}$, variation in v will not affect the plot apart from the resolution of the peaks, but for lower $\varepsilon \ell_{\max}$, lower value of v creates spurious ripples across the plot.

Parameter ε

From Fourier analysis literature, it is known that the discretization of the integral in (3.4) with ε leads to multiple copies of the spectrum. According to the sampling theorem, in order to avoid aliasing among them, the choice of ε is controlled by the net bandwidth B of the spectrum as $\varepsilon < 1/(2B)$. Here $B \approx |\lambda_1 - \lambda_{|V|}| + 6v$, including 99.7% of the Gaussian variations associated with λ_1 and $\lambda_{|V|}$. We have, $|\lambda_1 - \lambda_{|V|}| < 2\lambda_1 < 2\Delta$, with Δ being the maximum degree of the graph and a proof of the last inequality can be found in [Lovász 2007]. Hence choosing $\varepsilon < 1/(4\Delta + 12v)$ will ensure that sampling theorem is satisfied.

Parameter $\ell_{\max}$

From Section 3.3.1, T should → ∞ and $T\varepsilon \to 0$. This implies as $\ell_{\max} \to \infty$, ε should be chosen as $1/\ell_{\max} < \varepsilon < 1/\sqrt{\ell_{\max}}$, asymptotically.

Scalability

Combining the argument behind the selection of ε and $\ell_{\max}$ (with $\ell_{\max}$ chosen accordingly by fixing ε), we can say that $\ell_{\max}$ depends on the maximum degree Δ, not on the number of nodes $|V|$. Thus, we expect that our approach is highly scalable.

3.7 Numerical Results

We demonstrate the algorithms described above with numerical studies on real-world networks. First, in order to compare and show the effectiveness of the different techniques, we consider Les Misérables network, graph of characters of the novel Les Misérables. Later, we examine Enron email network, the email communication network among Enron employees and DBLP network which is a co-authorship network from the DBLP computer science bibilography. We have chosen these datasets so that their sizes differ in orders of magnitude. The datasets are taken from [Leskovec & Krevl 2014] where several parameters of the datasets can be found.

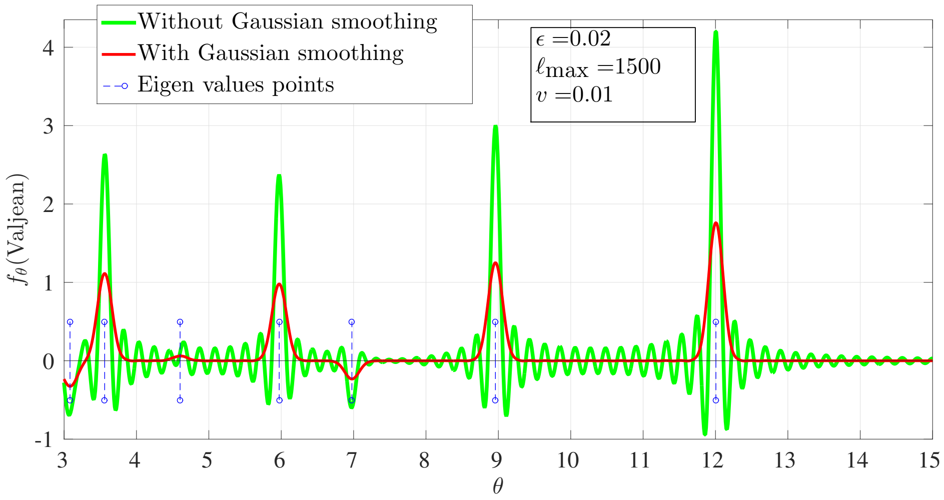


Figure 3.2: Les Misérables graph: with and without Gaussian smoothing

In the following simulation results, in order to show the decomposition at the node level, we consider one particular node in each of the networks examined. We select such a node as one of the top 2% highest degree nodes in the network. In the figures, we have also shown the actual eigenvalue points, which are cross-checked with *eigs* function in Matlab and *adjacency_spectrum* function in the Python module Networkx.

Les Misérables network

In Les Misérables network, nodes are the characters and edges are formed if two characters appear in the same chapter. The number of nodes is 77 and number of edges is 254. We look at the spectral plot in a specific node called Valjean, a character in the associated novel.

We first show in Figure 3.2 the smoothing effect the Gaussian term brings in the finite sum approximation (3.6). Indeed, the Gaussian smoothing technique eliminates spurious picks.

Different centralized algorithms are shown in Figure 3.3. As shown in the figure, the order-1 algorithm takes ten times more $\ell_{\max}$ than order-2 and order-4. This is mainly because lower ε is needed for order-1 due to slower convergence and this in turn leads to higher $\ell_{\max}$. We also observe that order-4 matches nearly perfectly with the theoretical values.

The numerical results for the Monte Carlo gossiping technique explained in Section 3.4.3 is shown in Figure 3.4. Interestingly, even one iteration of Monte Carlo averaging provides sufficient information about the eigenvalues and it can be observed that smaller number of iterations are needed for practical convergence of this algorithm.

Figure 3.5 presents the random walk implementation of the Monte Carlo gossiping. Here we used only one Monte Carlo averaging and four random walks are

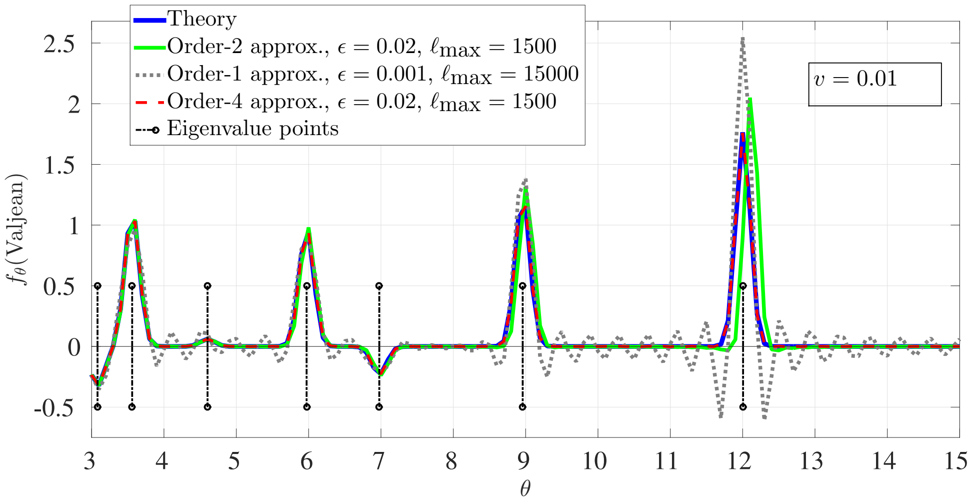


Figure 3.3: Les Misérables graph: Centralized algorithms

started from each node. We notice again that just with one iteration, it shows very good performance with respect to the order-1 centralized algorithm.

Enron email network

The nodes in this network are the email ID's of the employees in Enron and the edges are formed when two employees communicated through email. Since the graph is not connected, we take the largest connected component with $33,696$ nodes and $180,811$ edges. The node considered is the highest degree node in that component.

Figure 3.6 shows the complex diffusion with order-4 calculation and we find it is exactly matching with the theory. In Figure 3.7, the gossiping approach is shown which also performs very well.

DBLP network

We provide one more example, DBLP computer science network, which is ten times larger than Enron email network. It is made out of co-authorship network where the nodes are the authors and the edges between two authors are formed if they have written at least one paper together. The number of nodes is $317,080$ and number of edges is $1,049,866$. We consider the node with ID 6737, which has degree in top 2% of the network.

In order to show the effectiveness of the scalability argument provided in Section 3.6.3 for higher order approximations, the diffusion algorithm with order-4 is shown in Figure 3.8. The algorithm is matching very well with the theory. The ℓ_{max} for order-4 diffusion in Enron-email and DBLP networks for a good match with theory, is around 5000. This is in tune with our finding in Section 3.6.3 that ℓ_{max} mainly depends on $|\lambda_1 - \lambda_{|V|}|$. In case of Enron-email $|\lambda_1 - \lambda_{|V|}| = 159.72$ and in DBLP network $|\lambda_1 - \lambda_{|V|}| = 132.42$. They are at least in the same order.

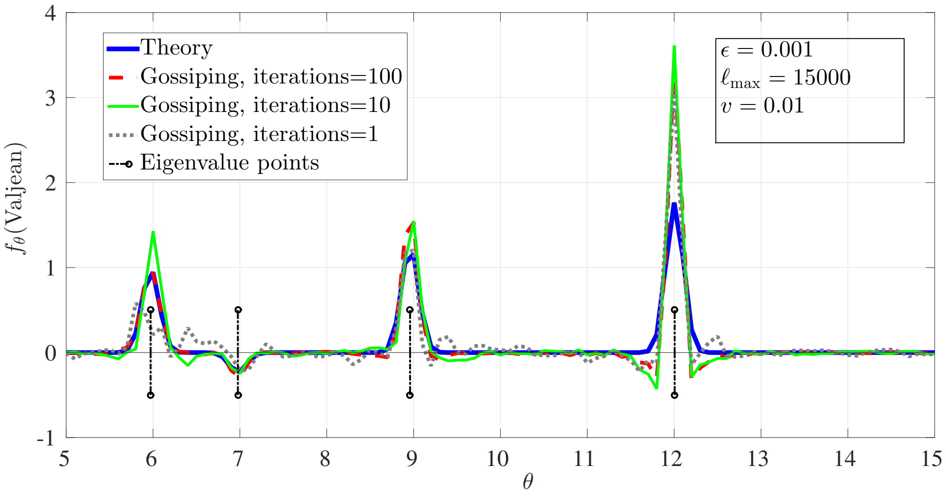


Figure 3.4: Les Misérables graph: Monte Carlo gossiping

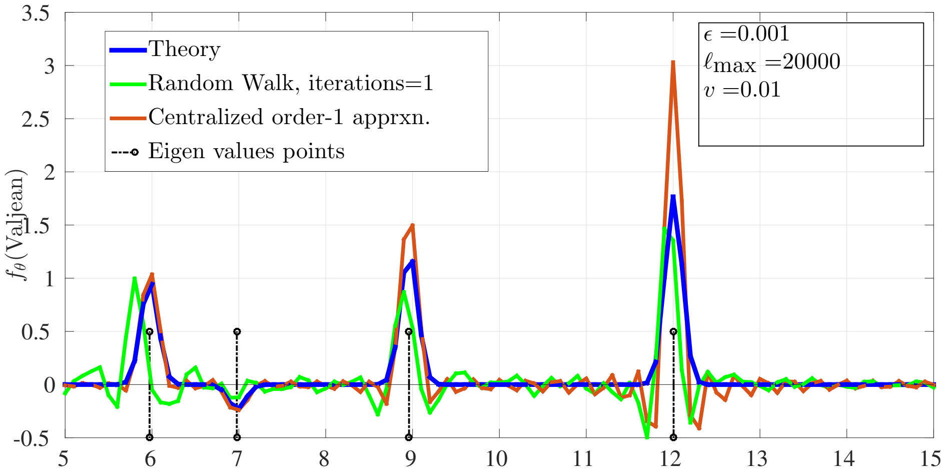


Figure 3.5: Les Misérables graph: random walk

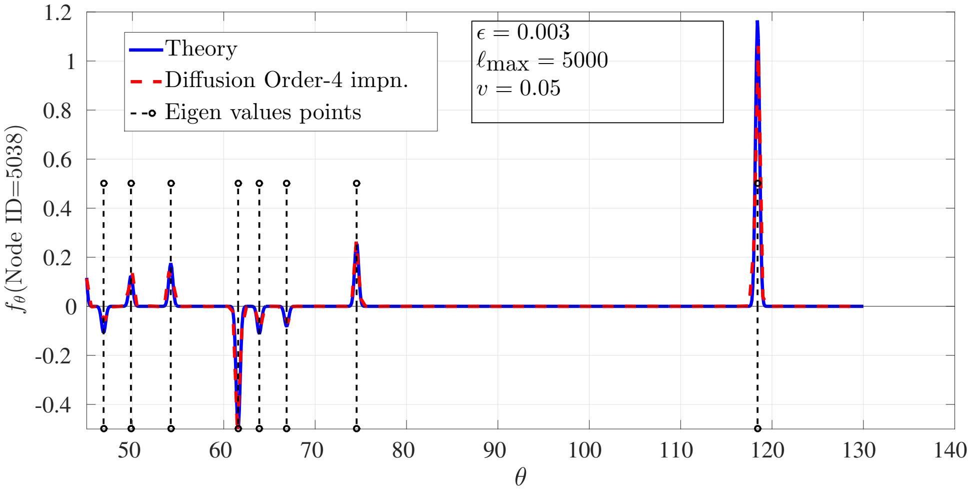


Figure 3.6: Enron email network: Diffusion order-4

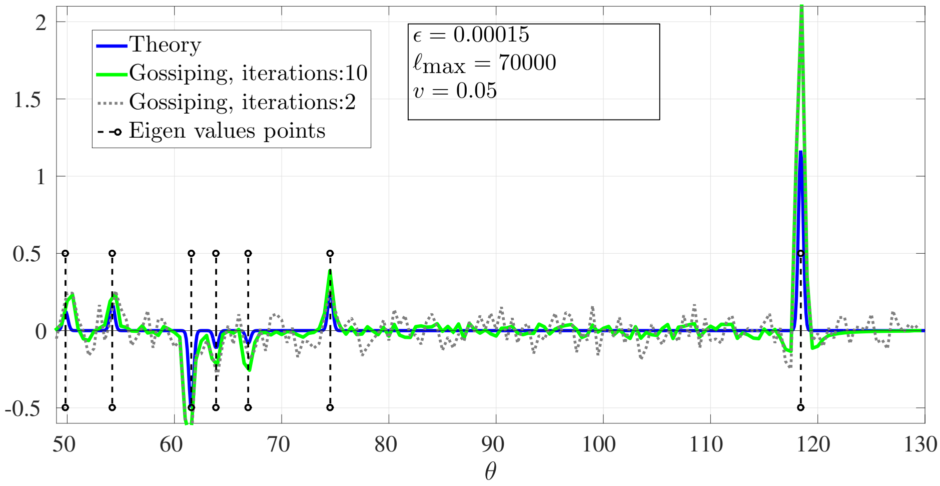


Figure 3.7: Enron email network: Gossiping

3.8 Vector Description of Complex Diffusion

The Algorithm 3.4.1 can be implemented in the following way. This technique is based on asynchronous updates. Each node m maintains three storage units. We use “level" to indicate index in the buffer, since index in the first two buffer represents the level of the diffusion.

- `Fld_Buff`: It is the main buffer which stores the fluid and is of size $\ell_{\max}+1$. Fluid fusion and diffusion happen in this buffer. Note that while diffusing, the buffer diffuses copies of the fluid in the buffer without disturbing the actual fluid.

- `Fld_Buff_count`: It is also of size $\ell_{\max}+1$. This buffer counts the number of fusions happened from neighbors at each level ℓ.

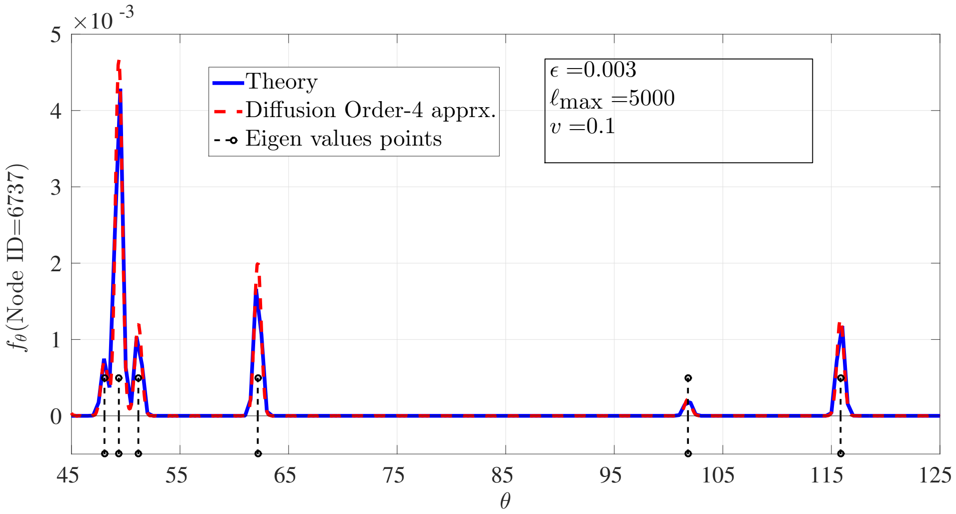


Figure 3.8: DBLP network: Diffusion order-4

- `Buff_to_diff`: It is a pointer to the level `Fld_Buff`.

Each fluid fusion request to node m from a neighbor h contains a unit of weighted fluid and the level l to which the fluid to be fused.

1. Initial value:

$$\begin{aligned}\texttt{Fld\_Buff}(0) &= \boldsymbol{b}_0(m)\\ \texttt{Fld\_Buff\_count}(0) &= d_m + 1\end{aligned}$$

2. `Fld_Buff_count`(ℓ) increments by one when `Fld_Buff`(ℓ) fuses one unit of fluid from a neighbor at level l.

3. At any point of time, if at some level l `Fld_Buff_count`(ℓ) $= d_m + 1$, then it triggers `Fld_Buff`(ℓ) to diffuse a copy of the fluid contained in it to the level $\ell + 1$ of all its neighbors and of itself.

4. `Buff_to_diff` contains the level in `Fld_Buff` where the latest diffusion happens. This protects the system from multiple diffusion from each level.

5. When the `Buff_to_diff` $= \ell_{\max} + 1$, the fusion-diffusion process stops.

6. Finally `Fld_Buff` at node m gives the mth row of $\boldsymbol{X}$.

3.9 Issues in the Computation with High Resolution

While doing numerical experiments, we have observed that the approaches explained in the previous sections work well for larger eigenvalues of the adjacency matrix of a graph but they do not perform that well when one needs to distinguish between the eigenvalues which are very close to each other. One of the main techniques proposed there to solve

$$\frac{\partial}{\partial t}\boldsymbol{\Psi}(t) = i\boldsymbol{A}\,\boldsymbol{\Psi}(t), \tag{3.12}$$

are via rth order Runge-Kutta method and its implementation as a diffusion process in the network. The r-th order Runge-Kutta method has the convergence rate of $\mathcal{O}(\varepsilon^r)$. However note that in practice we fix ε and increase the time interval of computation T (which is $s \times \varepsilon$ with s as the number of samples), and the error can increase over time. We have observed that this is the case while checking the trajectory of the associated differential equation solution; the solution diverges, and it happens when a large number of iterations s is required (see Section 3.13).

A larger value for s is anticipated from our approximation in (3.6) due to the following facts. From the theory of Fourier transform and Nyquist sampling, the following conditions must be satisfied:

$$\varepsilon \leqslant \frac{\pi}{\lambda_1} \text{ and } s \geqslant \frac{2\pi}{\varepsilon\lambda_{\text{diff}}}, \tag{3.13}$$

where λ_{diff} is the maximum resolution we require in the frequency (eigenvalue) domain, which is ideally $\min_i |\lambda_i - \lambda_{i+1}|$. This explains that when dealing with graph matrices with larger λ_n and require higher resolution, s will take higher orders. For instance in case of the Laplacian matrix, where the maximum eigenvalue is bounded as $\frac{|V|}{|V|-1}\Delta(G) \leqslant \lambda_1 \leqslant 2\Delta(G)$, with $\Delta(G)$ as the maximum degree of the graph and the lower eigenvalues are very close to each other, s turns out to be a large value.

In the rest of the chapter, we focus on the Laplacian matrix $\boldsymbol{L}$. We design algorithms based on Lagrangian as well as Hamiltonian mechanics, to compute the smallest k eigenvalues and the respective eigenvectors of the Laplacian matrix efficiently. For simplicity, we do not consider Gaussian smoothing (3.4) in the following sections, but the algorithms can be readily extended.

Modifications in the algorithms

1. We observe from our previous studies that the stability in trajectory of the differential equation solver is of significant influence in the eigenvalue-eigenvector technique. Thus we resort to geometric integrators to ensure the stability. In particular, by modeling as a Hamiltonian system, we use symplectic integrators (SI) which protect the volume preservation of Hamiltonian dynamics, thus preserve stability and improves accuracy.

2. We propose algorithms that are easy to design without involving many parameters with interdependence, compared to the algorithms proposed in the previous sections.

3.10 Mechanical Spring Analogy with Lagrangian Dynamics

Consider a hypothetical mechanical system representation of the graph G in which unit masses are placed on the vertices and the edges are replaced with mechanical springs of unit stiffness. Using either Lagrangian or Netwonian mechanics, the dynamics of this system is described by the following system of differential equations

$$\ddot{\boldsymbol{x}}(t) + \boldsymbol{L}\boldsymbol{x}(t) = \boldsymbol{0}. \tag{3.14}$$

The system has the Hamiltonian function as $\mathcal{H} = \frac{1}{2}\dot{\boldsymbol{x}}^\intercal \boldsymbol{I}\dot{\boldsymbol{x}} + \frac{1}{2}\boldsymbol{x}^\intercal \boldsymbol{L}\boldsymbol{x}$.

We note that once we obtain by some identification method the frequencies ω_k of the above oscillatory system, the eigenvalues of the Laplacian $\boldsymbol{L}$ can be immediately retrieved by the simple formula $\lambda_k = |\omega_k^2|$. This will be made clearer later in this section.

Starting with a random initial vector $\boldsymbol{x}(0)$, we can simulate the motion of this spring system. For the numerical integration, the Leapfrog or Verlet method [Verlet 1967] technique can be applied. The Verlet method has several remarkable properties. It has the same computational complexity as the Euler method but it is a second order method (Euler's method is employed in the complex diffusion of order-1) as the first order distributed diffusion). In addition, the Verlet method is stable for oscillatory motion and conserves the errors in energy and computations [Leimkuhler & Reich 2004, Chapter 4]. It has the following two forms. Let $\boldsymbol{p}(t) := \dot{\boldsymbol{x}}(t)$ and $\boldsymbol{x}_i$ be the approximation of $\boldsymbol{x}(i\varepsilon)$, similarly $\boldsymbol{p}_i$ for $\boldsymbol{p}(i\varepsilon)$. Here ε is the step size for integration. First, define

$$\boldsymbol{p}_{1/2} = \boldsymbol{p}_0 + \varepsilon/2(-\boldsymbol{L}\boldsymbol{x}_0).$$

Then, perform the following iterations

$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{x}_{i-1} + \varepsilon \boldsymbol{p}_{i-1/2} \\ \boldsymbol{p}_{i+1/2} &= \boldsymbol{p}_{i-1/2} + \varepsilon(-\boldsymbol{L}\boldsymbol{x}_i). \end{aligned}$$

Equivalently, one can do the updates as

$$\begin{aligned} \boldsymbol{x}_{i+1} &= \boldsymbol{x}_i + \varepsilon p_i + \varepsilon^2/2(-\boldsymbol{L}\boldsymbol{x}_i) \\ \boldsymbol{p}_{i+1} &= \boldsymbol{p}_i + \varepsilon[(-\boldsymbol{L}\boldsymbol{x}_i) + (-\boldsymbol{L}\boldsymbol{x}_{i+1})]. \end{aligned}$$

We name the above algorithm as Order-2 Leapfrog.

Solution of the differential equation (3.14) subject to the boundary values $\boldsymbol{x}(0) = \boldsymbol{a}_0$ and $\boldsymbol{p}(0) = \boldsymbol{b}_0$ is

$$\boldsymbol{x}(t) = \left(\frac{1}{2}\boldsymbol{a}_0 - i\frac{\boldsymbol{b}_0}{\sqrt{\Lambda}}\right)e^{it\sqrt{\boldsymbol{L}}} + \left(\frac{1}{2}\boldsymbol{a}_0 + i\frac{\boldsymbol{b}_0}{\sqrt{\Lambda}}\right)e^{-it\sqrt{\boldsymbol{L}}},$$

where we assume the decomposition of $\boldsymbol{L}$ based on spectral theorem, i.e., $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\intercal$ with $\boldsymbol{U}$ as the orthogonal matrix with columns as eigenvectors and $\boldsymbol{\Lambda}$ as the diagonal matrix formed from the eigenvalues. Further simplification of the above expression along with the fact that $f(\boldsymbol{L}) = \boldsymbol{U} f(\boldsymbol{\Lambda})\boldsymbol{U}^\intercal$, for any function f which can be expressed in terms of power series, gives

$$\boldsymbol{x}(t) = \cos(t\sqrt{\boldsymbol{L}})\boldsymbol{a}_0 + (\sqrt{\boldsymbol{L}})^{-1}\sin(t\sqrt{\boldsymbol{L}})\boldsymbol{b}_0.$$

or kth component of $\boldsymbol{x}(t)$ is

$$\boldsymbol{a}_0[k]\cos(t\sqrt{\lambda_k}) + \frac{\boldsymbol{b}_0[k]}{\sqrt{\lambda_k}}\sin(t\sqrt{\lambda_k}).$$

Now we have

$$\begin{aligned}
\int_{-\infty}^{+\infty} \boldsymbol{x}(t)e^{-it\theta}dt &= \int_{-\infty}^{+\infty}\sum_{k=1}^{|V|}\cos(t\sqrt{\lambda_k})\boldsymbol{u}_k(\boldsymbol{u}_k^\intercal\boldsymbol{a}_0)e^{-it\theta}dt \\
&\quad + \int_{-\infty}^{+\infty}(\sqrt{\boldsymbol{L}})^{-1}\sum_{k=1}^{|V|}\sin(t\sqrt{\lambda_k})\boldsymbol{u}_k(\boldsymbol{u}_k^\intercal\boldsymbol{b}_0)e^{-it\theta}dt \\
&= \sum_{k=1}^{|V|}\boldsymbol{u}_k(\boldsymbol{u}_k^\intercal\boldsymbol{a}_0)\left(\pi[\delta(\theta-\sqrt{\lambda_k})+\delta(\theta+\sqrt{\lambda_k})]\right) \\
&\quad +(\sqrt{\boldsymbol{L}})^{-1}\boldsymbol{u}_k(\boldsymbol{u}_k^\intercal\boldsymbol{b}_0)\left(-\pi i[\delta(\theta-\sqrt{\lambda_k})-\delta(\theta+\sqrt{\lambda_k})]\right).
\end{aligned}$$

Taking the real and positive spectrum will give $\pi\sum_{k=1}^{|V|}\boldsymbol{u}_k(\boldsymbol{u}_k^\intercal\boldsymbol{a}_0)\delta(\theta-\sqrt{\lambda_k})$. The whole operation can be approximated by applying an s-point FFT on $\{\boldsymbol{x}_i, 0 \leqslant i < s\}$, and taking real values. (To be exact, there is a phase factor to be multiplied to the kth point in FFT approximation, and is given by $(\sqrt{2\pi})^{-1}\varepsilon\exp(-it_0k\lambda_{\text{diff}})$, where we considered the time interval $[t_0, t_0 + s\varepsilon]$).

Note that (3.14) is different from the original differential equation in Section 3.3 where it is $\dot{\boldsymbol{x}}(t) = i\boldsymbol{L}\boldsymbol{x}(t)$ containing complex coefficients.

3.11 Hamiltonian Dynamics and Relation with Quantum Random Walk

In the previous sections we have studied the Schrödinger type equation of the form (3.12) with $\boldsymbol{\psi}$ as the wave function. Now let us consider a similar equation with respect to the graph Laplacian

$$\dot{\boldsymbol{\psi}}(t) = i\boldsymbol{L}\boldsymbol{\psi}(t). \tag{3.15}$$

The solution of this dynamics is closely related to the evolution of continuous time quantum random walk and algorithms are developed in Section 3.5 based on this observation.

Now since the matrix $\boldsymbol{L}$ is real and symmetric, it is sufficient to use the real-imaginary representation of the wave function $\boldsymbol{\psi}(t) = \boldsymbol{x}(t) + i\boldsymbol{y}(t)$, $\boldsymbol{x}(t), \boldsymbol{y}(t) \in \mathbb{R}$.

Substituting this representation into equation (3.15) and taking real and imaginary parts, we obtain the following system of equations

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= -\boldsymbol{L}\boldsymbol{y}(t)\\ \dot{\boldsymbol{y}}(t) &= \boldsymbol{L}\boldsymbol{x}(t),\end{aligned}$$

or equivalently in the matrix form

$$\frac{d}{dt}\begin{bmatrix}\boldsymbol{x}(t)\\ \boldsymbol{y}(t)\end{bmatrix} = \begin{bmatrix}0 & -\boldsymbol{L}\\ \boldsymbol{L} & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{x}(t)\\ \boldsymbol{y}(t)\end{bmatrix}. \tag{3.16}$$

Such a system has the following Hamiltonian function

$$\mathcal{H} = \frac{1}{2}\boldsymbol{x}^\intercal\boldsymbol{L}\boldsymbol{x} + \frac{1}{2}\boldsymbol{y}^\intercal\boldsymbol{L}\boldsymbol{y}. \tag{3.17}$$

The next, very helpful, decomposition

$$\begin{bmatrix}0 & -\boldsymbol{L}\\ \boldsymbol{L} & 0\end{bmatrix} = \begin{bmatrix}0 & 0\\ \boldsymbol{L} & 0\end{bmatrix} + \begin{bmatrix}0 & -\boldsymbol{L}\\ 0 & 0\end{bmatrix},$$

together with the observation that

$$\exp\left(\begin{bmatrix}0 & 0\\ \boldsymbol{L} & 0\end{bmatrix}\right) = \begin{bmatrix}I & 0\\ \boldsymbol{L} & I\end{bmatrix},$$

leads us to another modification of the leapfrog method known as symplectic split operator algorithm [Blanes et al. 2006]:

Initialize with

$$\boldsymbol{\delta y} = -\boldsymbol{L}\boldsymbol{x}_0,$$

then perform the iterations

$$\begin{aligned}\boldsymbol{y}_{i-1/2} &= \boldsymbol{y}_{i-1} - \frac{\varepsilon}{2}\boldsymbol{\delta y}\\ \boldsymbol{x}_i &= \boldsymbol{x}_{i-1} - \varepsilon\boldsymbol{L}\boldsymbol{y}_{i-1/2}.\end{aligned} \tag{3.18}$$

Update

$$\begin{aligned}\boldsymbol{\delta y} &= -\boldsymbol{L}\boldsymbol{x}_i\\ \boldsymbol{y}_i &= \boldsymbol{y}_{i-1/2} - \frac{\varepsilon}{2}\boldsymbol{\delta y}.\end{aligned} \tag{3.19}$$

The above modified leapfrog method belongs to the class of symplectic integrator (SI) methods [Blanes et al. 2008, Leimkuhler & Reich 2004, Iserles 2008]. We name the above algorithm as *order-2 SI*.

The Hamiltonian approach can be implemented in two ways:

1. Form the complex vector $\boldsymbol{x}_k + i\boldsymbol{y}_k$ at each of the ε intervals. Then $\{\boldsymbol{x}_k + i\boldsymbol{y}_k, 0 \leqslant k < s\}$ with $\boldsymbol{x}_0 = \boldsymbol{a}_0$ and $\boldsymbol{y}_0 = \boldsymbol{b}_0$ approximates $\exp(iLt)(\boldsymbol{a}_0 + i\boldsymbol{b}_0)$ at $t = 0, \varepsilon, \ldots, (s-1)\varepsilon$ intervals. A direct application of s point FFT with appropriate scaling will give the spectral decomposition as in (3.3).

2. Note that the formulation in (3.16) is equivalent to the following differential equations
$$\ddot{\boldsymbol{y}}(t) + \boldsymbol{L}^2\boldsymbol{y}(t) = 0, \quad \ddot{\boldsymbol{x}}(t) + \boldsymbol{L}^2\boldsymbol{x}(t) = 0, \tag{3.20}$$
which are similar to the one in (3.14) except the term $\boldsymbol{L}^2$. Now on the same lines of analysis in the previous section, taking the real and positive spectrum of just $\boldsymbol{y}$ component will give $\pi \sum_{k=1}^{|V|} \boldsymbol{u}_k(\boldsymbol{u}_k^\intercal \boldsymbol{a}_0)\delta(\theta - \lambda_k)$.

3.11.1 Fourth order integrator

The Hamiltonian $\mathcal{H}$ in (3.17) associated with the Schrödinger-type equation has a special characteristic that it is seperable into two quadratic forms, which help to develop higher order integrators. The r stage integrator has the following form. Between t and $t+\varepsilon$ intervals, we run for $j = 1, \ldots, r$,

$$\begin{aligned} \boldsymbol{y}_j &= \boldsymbol{y}_{j-1} + p_j \varepsilon \boldsymbol{L}\boldsymbol{x}_{j-1} \\ \boldsymbol{x}_j &= \boldsymbol{x}_{j-1} - q_j \varepsilon \boldsymbol{L}\boldsymbol{y}_j. \end{aligned}$$

In order to make qth order integrator $r \leqslant q$. For our numerical studies we take the optimized coefficients for order-4 derived in [Gray & Manolopoulos 1996]. We call the above algorithm as *order-4 SI*.

3.12 Distributed Implementation

Here we explain a diffusion algorithm of the Hamiltonian system based algorithm given in the previous section, in which each node needs to communicate only to its neighbors.

Note that the algorithm with the updates (3.18)-(3.19) can be further simplified as, with $\boldsymbol{x}(0) = \boldsymbol{a}_0$ and $\boldsymbol{y}(0) = \boldsymbol{b}_0$,

$$\begin{aligned} \boldsymbol{y}_i &= \boldsymbol{y}_{i-1} + \frac{\varepsilon}{2}\boldsymbol{L}(\boldsymbol{x}_{i-1} + \boldsymbol{x}_i) \\ \boldsymbol{x}_i &= \left(I - \frac{\varepsilon^2 \boldsymbol{L}^2}{2}\right)\boldsymbol{x}_{i-1} - \varepsilon \boldsymbol{L}\boldsymbol{y}_{i-1}. \end{aligned} \tag{3.21}$$

Now $\boldsymbol{x}_i$ can be updated independent to $\boldsymbol{y}_i$ as

$$\boldsymbol{x}_i = (2I - \varepsilon^2 \boldsymbol{L}^2)\boldsymbol{x}_{i-1} - \boldsymbol{x}_{i-2}, \tag{3.22}$$

with $\boldsymbol{x}_0 = \boldsymbol{a}_0$ and $\boldsymbol{x}_1$ as $\boldsymbol{c}_0 := (I - \varepsilon^2\boldsymbol{L}^2/2)\boldsymbol{a}_0 - \varepsilon \boldsymbol{L}\boldsymbol{b}_0$. From (3.20) and the explanation given there, it is known that the solution $\{\boldsymbol{x}_i\}$ is sufficient to obtain the eigenspectrum. Such an implementation will also avoid the four level synchronism needed in (3.18)-(3.19). If required, one can also form $\{\boldsymbol{x}_i + i\boldsymbol{y}_i\}$ with more cost and observe the Fourier transform directly.

The diffusion technique is presented in Algorithm 3.12.1. The crux of the algorithm is the computation of the associated polynomial in z, $\boldsymbol{x}(z) = \sum_{i=0}^{s} z^\ell \boldsymbol{x}_i$.

At each node k we find the polynomial $x(z)$ which corresponds to kth row in $\boldsymbol{x}(z)$ (we drop the dependence on k in the notation for ease of use). The node k will start with initial fluids $C(z) = \boldsymbol{c}_0[k]z$ and $C_2(z) = 0$. Copies of these fluids are diffused to all of its neighbors with the complex weight $i\varepsilon\,\ell_{k,h}$ for $h \in \mathcal{N}(k)$. The temporary polynomial $C_2(z)$ is an artifact which implements the diffusion of the matrix $\varepsilon^2\boldsymbol{L}^2$. Indeed, the fluid must be retransmitted towards a relay before being added to $x(z)$. This is the reason why the number of iterations must be raised to $2s$. The generalization to higher order integrators is straightforward since it consists of the diffusion of matrix $(\varepsilon\boldsymbol{L})^r$.

We assume that $\boldsymbol{c}_0$, which is $\boldsymbol{x}_1$, is calculated apriori from $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$. In the algorithm the procedures Send(h, f) and Send-2(h, f) transmit fluid f to node h and are responsible for updating the polynimials $C(z)$ and $C_2(z)$ at the receiver node. Similarly Receive(h) and Receive-2(h) collect fluid from h for $C(z)$ and $C_2(z)$. The procedure Mod(n, p) returns the remainder after dividing n by p and Coeff$(x(z), z^d)$ gives coefficient of the term z^d in $x(z)$.

Algorithm 3.12.1: Diffusion-SI$(k, \boldsymbol{c}_0, \boldsymbol{a}_0)$

```
$C(z) \leftarrow \boldsymbol{c}_0[k]z$
$C_2(z) \leftarrow 0$
$x(z) \leftarrow \boldsymbol{a}_0[k]$
$n \leftarrow 3$
while ($n \leqslant 2s$)
  do
    for each $h \in \mathcal{N}(k)$
      do
        Send($h, \ell_{k,h}\varepsilon i\, C_2(z)$)
        Send-2($h, \ell_{k,h}\varepsilon i\, C(z)$)
    $C_2(z) \leftarrow \ell_{k,k}\varepsilon i C(z)$
    $C(z) \leftarrow 2zC(z)$
    for each $h \in \mathcal{N}(k)$
      do
        $C(z) \leftarrow C(z) + z$Receive($h$)
        $C_2(z) \leftarrow C_2(z) +$ Receive-2($h$)
    if Mod($n, 2$) $= 0$
      then {$C(z) \leftarrow C(z) - z^{n/2}$ Coeff($x(z), z^{n/2-1}$)
    $x(z) \leftarrow x(z) + C(z)$
    $n \leftarrow n + 1$
return ($x(z)$)
```

3.13 Numerical Results on Symplectic Integrators

Here we present results from simulations on real-world networks using the algorithms based on Leapforg and symplectic integrators. The parameters ε and s are chosen in the numerical studies satisfying the constraints in (3.13). We assume that the maximum degree is known to us.

Note that if the only purpose is to detect eigenvalues, not to compute the eigenvectors, then instead of taking real part of the FFT in the Hamiltonian solution, it is clearly better to compute the absolute value of the complex quantity to get higher peaks. But in the following simulations we look for eigenvectors as well.

For the numerical studies, in order to show the effectiveness of the distributed implementation, we focus on one particular node and plot the spectrum observed at this node. In the plots, $f_\theta(k)$ indicates the approximated spectrum at frequency θ observed on node k.

3.13.1 Les Misérables network

In Les Misérables network, the number of nodes is 77 and number of edges is 254. We look for the spectral plot at a specific node called Valjean (with node ID 11), a character in the associated novel.

The instability of the Euler method is clear from Figure 3.9, whereas Figure 3.10 shows the guaranteed stability of Hamiltonian SI. Figure 3.11 shows the Lagrangian Leapfrog method given in Section 3.10. It can be observed that very few smallest eigenvalues are detected using order-2 Leapfrog compared to the SI technique (order-2) in Figure 3.12. Figure 3.13 shows order-4 SI with much less number of iterations. The precision in order-4 plot can be significantly improved further by increasing the number of iterations.

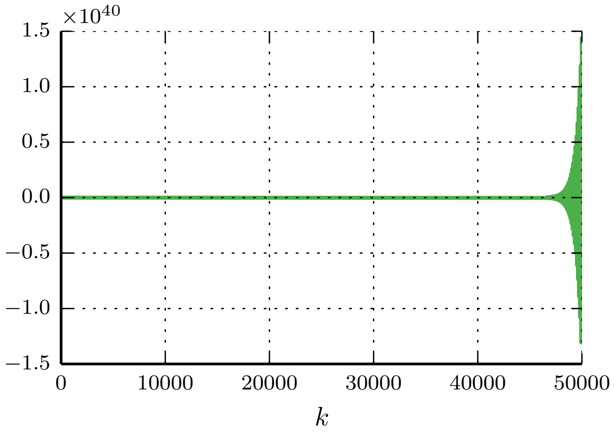


Figure 3.9: Euler method trajectory

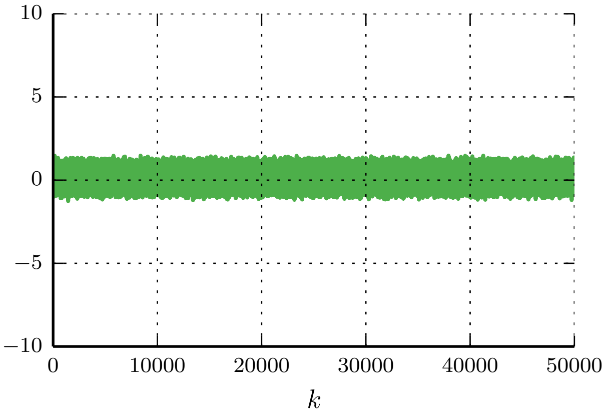


Figure 3.10: Hamiltonian order-2 SI trajectory

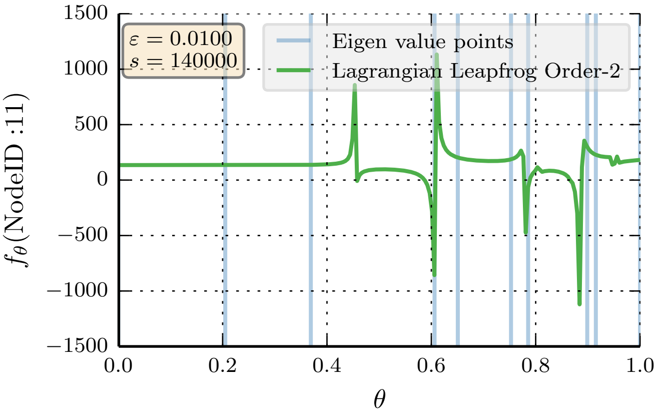


Figure 3.11: Les Misérables network: Order-2 Leapfrog

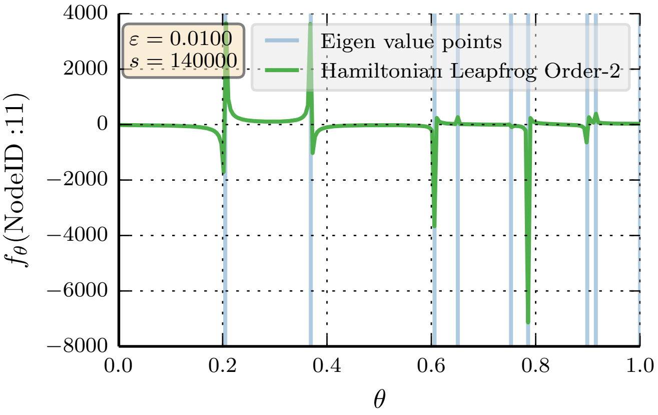


Figure 3.12: Les Misérables network: Order-2 SI

3.13.2 Coauthorship graph in network science

The coauthorship graph represents a collaborative network of scientists working in network science as compiled by M. Newman [Newman 2006]. The numerical

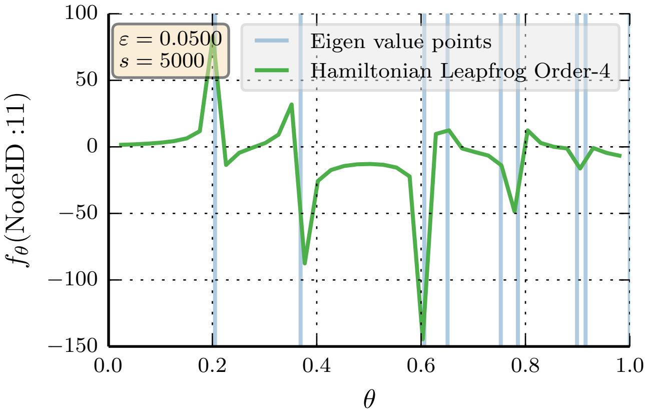


Figure 3.13: Les Misérables network: Order-4 SI

experiments are done on the largest connected component with $|V| = 379$ and $|E| = 914$. Figure 3.14 displays the order-4 SI simulation and it can be seen that even though the eigenvalues are very close, the algorithm is able to distinguish them clearly.

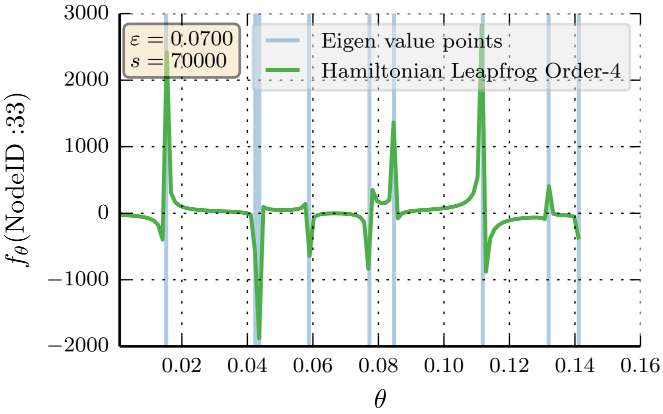


Figure 3.14: Coauthorship graph in Network Science: Order-4 SI

3.13.3 Arxiv HEP-TH graph

Arxiv HEP-TH (High Energy Physics-Theory) collaboration network [Leskovec & Krevl 2014] is created from the arXiv and it represents scientific collaborations between authors whose papers are submitted to High Energy Physics-Theory category. Nodes are the authors and the connections indicate that the two authors have written a paper together. The largest connected component is taken with number of nodes as $8,638$ and number of edges as $24,827$. The results of order-4 SI is shown in Figure 3.15.

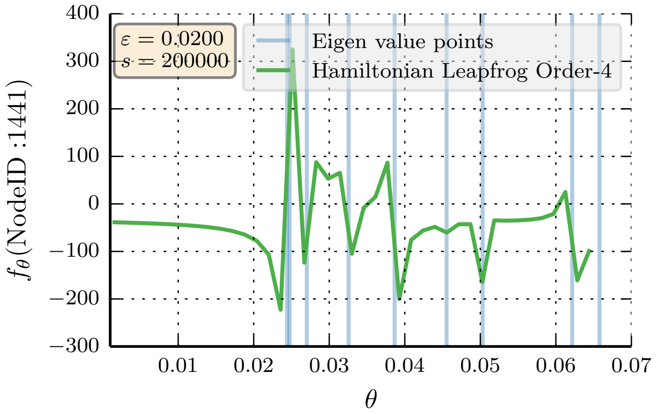


Figure 3.15: Arxiv HEP-TH graph: Order-4 SI

3.13.4 Enron email network

The nodes in this network are the email ID's of the employees in a company called Enron and the edges are formed when two employees communicated through email [Leskovec & Krevl 2014]. Since the graph is not connected, we take the largest connected component with 33, 696 nodes and 180, 811 edges. The node under focus is the highest degree node in that component. Simulation result is shown in Figure 3.16.

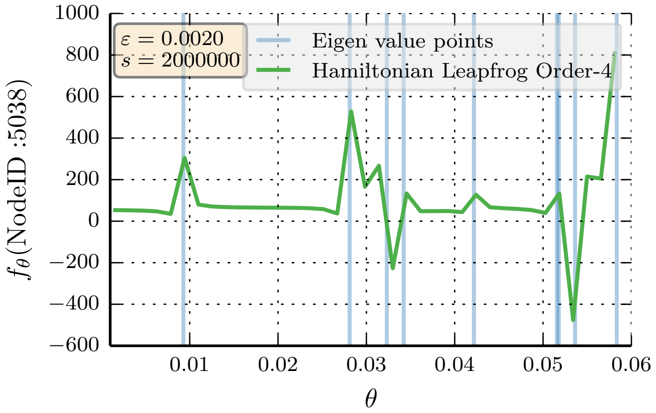


Figure 3.16: Enron graph: Order-4 SI

3.14 Conclusions

This chapter proposed some new approaches for distributed spectral decomposition of graph matrices. The fourier analysis of complex exponential of the graph matrices, with Gaussian smoothing, turns out to have a simple interpretation of the spectrum in terms of peaks at eigenvalue points. This exposition led us to develop two efficient distributed algorithms based on "complex power iterations": complex diffusion if the nodes can collect data from all the neighbors and complex gossiping when data is taken from only one random neighbor. Then we detailed the connection of complex exponential of graph matrices to quantum random walk techniques. We derived the rate of convergence of algorithms and it is found that the algorithms are scalable in proportion to the maximum degree of the graph. Numerical simulations on real-world networks of varying order of magnitude in size, showed the effectiveness and scalability of our various algorithms. Later in the chapter, we developed algorithms based on Hamiltonian dynamics and symplectic integrators to mitigate the high resolution issues.

Chapter 4
Inference in OSNs via Lightweight Partial Crawls

What is the fraction of male-female connections against that of female-female connections in a given OSN? Is the OSN assortative or disassortative? Edge, triangle, and node statistics of OSNs find applications in computational social science (see e.g. [Ottoni et al. 2013]), epidemiology [Pastor-Satorras & Vespignani 2001], and computer science [Benevenuto et al. 2009, Gjoka et al. 2013]. Computing these statistics is a key capability in large-scale social network analysis and machine learning applications. But because data collection in the wild is often limited to partial OSN crawls through API requests, observational studies of OSNs – for research purposes or market analysis – depend in great part on our ability to compute network statistics with incomplete data. Case in point, most datasets available to researchers in widely popular public repositories are partial OSN crawls[1]. Unfortunately, these incomplete datasets have unknown biases and no statistical guarantees regarding the accuracy of their statistics. To date, the best performing methods for crawling networks ([Avrachenkov et al. 2010, Gjoka et al. 2011, Ribeiro & Towsley 2010]) show good real-world performance but only provide statistical guarantees asymptotically (i.e., when the entire OSN network is collected).

This work addresses the fundamental problem of obtaining unbiased and reliable node, edge, and triangle statistics of OSNs via partial crawling. *To the best of our knowledge our method is the first to provide a practical solution to the problem of computing OSN statistics with strong theoretical guarantees from a partial network crawl.* More specifically, we (a) provide a provable finite-sample unbiased estimate of network statistics (and their spectral-gap derived variance) and (b) provide the approximate posterior of our estimates that performs remarkably well in all tested real-world scenarios.

More precisely, let $G = (V, E)$ be an undirected labeled network – not necessarily connected – where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. Unlike the usual definition of E where each edge is only present once, to simplify our notation we consider that if $(u, v) \in E$ then $(v, u) \in E$. Both edges and nodes can have labels. Network G is unknown to us except for $n > 0$ *arbitrary* initial seed nodes in $I_n \subseteq V$. Nodes in I_n must span all the different connected components of G. From the seed nodes we crawl the network starting from I_n and obtain a set of

[1] Public repositories such as SNAP [Leskovec & Krevl 2014], KONECT [Kunegis 2013] and datasets from LAW laboratory [Boldi et al. 2011] contain a majority of partial website crawls, not complete datasets or uniform samples.

crawled edges $\mathcal{D}_m(I_n)$, where $m > 0$ is a parameter that regulates the number of website API requests. With the crawled edges $\mathcal{D}_m(I_n)$ we seek an unbiased estimate of

$$\mu(G) = \sum_{(u,v)\in E} g(u,v)\,, \tag{4.1}$$

for any function $g(u,v)$ over the node pair (u,v). Note that functions of the form eq. (4.1) are general enough to compute node statistics and statistics of triangles (see the formulas (1.3), (1.4)).

Organization

The remainder of the chapter is organized as follows. Sections 4.1 lists some related works. In Section 4.2 we introduce key concepts and defines the notation used throughout this chapter. Section 4.3 presents the algorithms to build the super-node and proves the equivalence between them. The frequentist estimators and their properties are explained in Section 4.4. Section 4.5 contains the main result of the posterior distribution in Bayesian framework. Section 4.6 consists of experimental results over real-world networks. Section 4.7 extends the estimator to the average function $\bar{\nu}(G)$ and shows numerical results using this estimator. Finally, in Section 4.8 we present our conclusions.

4.1 Related Work

The works of [Massoulié et al. 2006] and [Cooper et al. 2013] are the ones closest to ours. The paper [Massoulié et al. 2006] estimates the size of a network based on the return times of random walk tours. The work in [Cooper et al. 2013] estimates number of triangles, network size, and subgraph counts from weighted random walk tours using results of [Aldous & Fill 2002, Chapter 2 and 3]. The previous works on finite-sample inference of network statistics from incomplete network crawls [Goel & Salganik 2009, Koskinen et al. 2010, Koskinen et al. 2013, Handcock & Gile 2010, Heckathorn 1997, Ligo et al. 2014, Thompson 2006] need to fit the partial observed data to a probabilistic graph model such as ERGMs (exponential family of random graphs models). Our work advances the state-of-the-art in estimating network statistics from partial crawls because: (a) we estimate statistics of arbitrary edge functions without assumptions about the graph model or the underlying graph; (b) we do not need to bias the random walk with weights as in Cooper et al.; this is particularly useful when estimating multiple statistics reusing the same observations; (c) we derive upper and lower bounds on the variance of estimator, which both show the connection with the spectral gap; and, finally, (d) we compute a posterior over our estimates to give practitioners a way to access the confidence in the estimates without relying on unobtainable quantities like the spectral gap and without assuming a probabilistic graph model.

In our work we provide a partial crawling strategy using short dynamically adjustable random walk tours starting at a “virtual” super-node without invoking the notion of lumpability [Kemeny & Snell 1983]. A random walk tour is a random walk sample path that starts and ends at the same node on the graph. We use these tours to compute a frequentist unbiased estimator of $\mu(G)$ (including its variance) regardless of the number of nodes, $n > 0$, in the seed set and regardless of the value of $m > 0$, unlike previous asymptotically unbiased methods [Avrachenkov et al. 2010, Gjoka et al. 2011, Lee et al. 2012, Ribeiro & Towsley 2010, Ribeiro et al. 2012]. We also provide a Bayesian approximation of the posterior of $\mu(G)$ given the observed tours $\mathbb{P}[\mu(G)|\mathcal{D}_m(I_n)]$, which is shown to be consistent. In our experiments we note that the posterior is remarkably accurate using a variety of networks large and small. Furthermore, when the network is formed by randomly wiring connections while preserving degrees and attributes of the nodes in the observed network, we devise an estimation technique for the expected true value with partial knowledge of the original graph.

4.2 Super-node Rationale

In this section we present definitions and concepts used throughout the reminder of the chapter. Then we substantiate an intuitive reasoning that our random walk tours are shorter than the "regular random walk tours" because the "node" that they start from is an amalgamation of a multitude of nodes in the graph.

Preliminaries

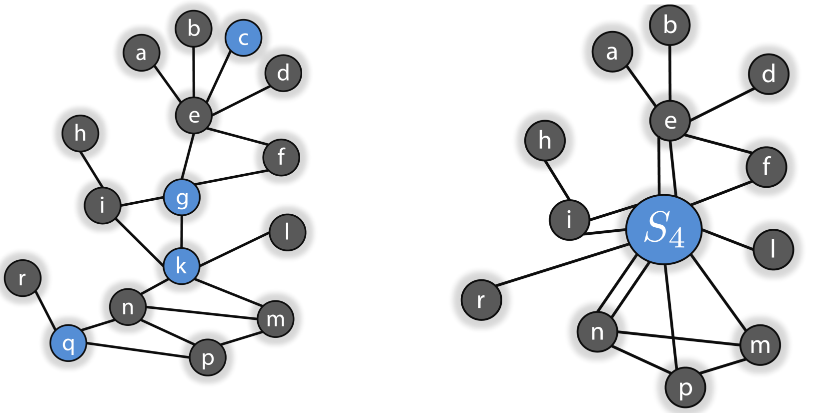
Let $G = (V, E)$ be an unknown undirected graph with $|V|$ as the number of nodes and $|E|$ as the number of edges (with a slight abuse of notation, counting the pairs $(u, v) \in E$ and $(v, u) \in E$ only once). Our goal is to find an unbiased estimate of $\mu(G)$ in eq. (4.1) and its posterior by crawling a small fraction of G. We are given a set of $n > 0$ initial *arbitrary* nodes denoted $I_n \subset V$. If G has disconnected components I_n must span all the different connected components of G.

Unless stated otherwise our network crawler is a classical random walk (RW) over the following augmented multigraph $G' = (V', E')$ with $|V'|$ nodes and $|E'|$ edges. A multigraph is a graph that can have multiple edges between two nodes. In $G'(I_n)$ we aggregate all nodes of I_n into a single node, denoted hereafter S_n, the *super-node*. Thus, $V'(I_n) = \{V \backslash I_n\} \cup \{S_n\}$. The edges of $G'(I_n)$ are $E'(I_n) = E \backslash \{E \cap \{I_n \times V\}\} \cup \{(S_n, v) : \forall (u, v) \in E, \text{ s.t. } u \in I_n \text{ and } v \in V \backslash I_n\}$, i.e., $E'(I_n)$ contains all the edges in E including the edges from the nodes in I_n to other nodes, and I_n is merged into the super-node S_n. Note that $G'(I_n)$ is necessarily connected as I_n spans all the connected components of G. For compactness of notation we sometimes refer to $G'(I_n)$ as G' when I_n is clear from the context. *We also use S_n and I_n interchangeably to denote both the super-node at $G'(I_n)$ and each individual node of I_n at G.*

A random walk on $G'(I_n)$ has transition probability from node u to an adjacent node v, $p_{uv} := \alpha_{u,v}/d_u$, where d_u is the degree of u and $\alpha_{u,v}$ is the number of edges between $u \in V'$ and $v \in V'$. Let $\boldsymbol{P} = \{p_{uv}\}$. We note that the theory presented in the chapter can be extended to more sophisticated random walks as well. Let $\{\pi_i\}$ be the stationary distribution at node i in the random walk on $G'(I_n)$.

A random walk *tour* is defined as the sequence of nodes $X_1^{(k)}, \ldots, X_{\xi_k}^{(k)}$ visited by the random walk during successive k-th and $k+1$-st visits to the super-node S_n. Here $\{\xi_k\}_{k \geqslant 1}$ denote the inter return times associated with successive returns to S_n. Tours have a key property: from the renewal theorem tours are independent since the returning times act as renewal epochs. Moreover, let $Y_1, Y_2, \ldots, Y_n$ be a random walk on $G'(I_n)$ in steady state.

Note that the random walk on $G'(I_n)$ is equivalent to a random walk on G where all the nodes in I_n are treated as *one single node*. Figure 4.1 shows an example of the formation of $G'(I_n)$.



(a) The original graph G with $I_4 = \{c, g, j, q\}$.

(b) The modified graph $G'(I_n)$ with super-node S_4.

Figure 4.1: Graph modification with the super-node

Why a super-node

The introduction of super-node is primary motivated by the following closely-related reasons:

- *Tackling disconnected or low-conductance graphs:* When the graph is not well connected or has many connected components, forming a super-node with representatives from each of the components make the modified graph connected and suitable for applying random walk theory. Even when the graph is connected, it might not be well-knit, i.e., it has low conductance. Since the conductance is closely related to mixing time of Markov chains, such graph will prolong the mixing of random walks. But with proper choice of super-node, we can reduce the mixing time and, as we show, improve the estimation accuracy.

  If we consider self loops from S_n to itself while forming the modified graph G', i.e. all the connections between I_n $(E \cap \{I_n \times I_n\})$, then G' becomes a contracted graph [Chung 1996]. Then [Chung 1996, Lemma 1.15] says that if S_n is formed from $n = 2$ vertices, the spectral gaps of the two graphs $\delta_{G'}$ and δ_G are related as follows: $\delta_{G'} \geqslant \delta_G$. The spectral gap $\delta_{G'} = 1 - \lambda_2$, where λ_2 is the second largest eigenvalue of $\boldsymbol{P}$, and δ_G can be defined accordingly on the random walk on the original graph. The above argument with spectral gaps can be extended to $n > 2$ by induction and hence $\delta_{G'} \geqslant \delta_G$ follows. The improvement in the spectral gap proves that the modified graph will become well-knit (high conductance). Note that G' in our case does not involve self loops around S_n, but this is for the ease of computation as the function values over the self loops are known (from the knowledge of I_n and further API queries with them), and hence allowing self loops will only slower down the estimation of $\mu(G)$ outside the super-node.

- *No lumpability for random walks:* The theory of lumpability [Kemeny & Snell 1983, Section 6.3] provides ways to exactly partition a Markov chain. Unfortunately, lumping states in a classical random walk will not give an accurate representation of the random walk Markov chain and, thus, we consider a super-node S_n where all the nodes inside the super-node are merged into one node rather than partitioning the states. The edges from S_n are the collection of all the edges from the nodes inside the super-node which are connected to nodes outside super-node, and Markov chain property still holds on this formation with random walks on the new graph G'. The graph modification with S_n is illustrated with an example in Figure 4.1.

- *Faster estimate with shorter tours:* The expected value of the k-th tour length $\mathbb{E}[\xi_k] = 1/\pi_{S_n}$ is inversely proportional to the degree of the super-node d_{S_n}. Hence, by forming a massive-degree super-node we can significantly shorten the average tour length. This property is of great practical importance as it reduces the number of API queries required per tour.

4.3 Static and Dynamic Super-nodes

In what follows we describe the algorithms to build super-nodes. The static super-node technique selects the nodes I_n before starting the experiment, while the dynamic super-node recruits the nodes on the fly.

4.3.1 Static super-node Algorithm

The static super-node is selected by n nodes from G without replacement to form I_n. If the graph is disconnected, I_n must contain at least one node of each component of interest. To construct I_n we can crawl each component of the graph G. For instance, one can make I_n be the n largest degree nodes seen in a set of random walks with a total of $k > n$ steps (as in Avrachenkov et al. [Avrachenkov et al. 2014b]). Because random walks are biased towards large-degree nodes the resulting super-node S_n tends to have large degrees.

Once I_n is chosen we form the virtual graph $G'(I_n)$ and start m random walk tours from the virtual super-node S_n. We stop each tour when the walk comes back to S_n. One practical issue in building I_n is knowing how many nodes we need to recruit to keep the random walk tours short. To ameliorate the situation in what follows we consider a dynamic algorithm to select the super-node.

4.3.2 Dynamic super-node Algorithm

In a dynamic super-node, nodes are added into the super-node *on-demand* using a different random walk called the super-node recruiting walk. The super-node S_j starts with $j \geqslant 1$ nodes. S_j must span nodes in all graph components. The algorithm is as follows:

1. Run a super-node recruiting walk independent of all previous tours starting from S_n, $n \geqslant j$. Once a node of interest i, or set of nodes, are reached, stop the super-node recruiting walk.

2. Add a newly recruited node i to the super-node S_n, $n \geqslant j$, $S_{n+1} = S_n \cup \{i\}$. If node i appears in any of the previous tours, break these tour into multiple tours where i either ends or starts a new tour.

3. Generate a random number k_{redo} from the negative binomial distribution with number of successes as the number of previous tours (not counting the broken tours) and probability of success $d_{S_n}/d_{S_{n+1}}$, where $d_{S_{n+1}}$ is the degree of the new super-node that includes i and d_{S_n} is the degree of the super-node without i.

4. Perform $k_{\text{redo}} - k_{\text{broken}} > 0$ tours, where k_{broken} is the number of broken tours that start with node i and have length greater than two. These tours start at node i in $G'(S_i)$ with a first step into nodes in $\mathcal{N}(i)\backslash S_{i+1}$, where $\mathcal{N}(i)$ are the neighbors of i in $G'(S_i)$. The tour ends at either S_n or i. Note that in $G'(S_{i+1})$ these tours start at S_{i+1} and end at S_{i+1} with length greater than two. If $k_{\text{redo}} - k_{\text{broken}} < 0$ then randomly remove tours starting at i until only k_{redo} tours remain.

5. Redo steps 2–4 until all recruited nodes are added to the super-node.

6. We can now proceed normally with new super-node tours (or recruit more nodes if necessary by redoing steps 1–4).

The step 4 calculates the number of tours that might have happened in the past when the new node i was part of S_n. This retrospective event can be recreated by sampling from a negative binomial distribution with appropriate parameters.

4.3.3 Equivalence between dynamic and static super-node sample paths

In what follows we show that the tours of a dynamic super-node S_n^{dyn} and the tours of the same super-node as a static super-node have the same probability distribution.

Theorem 4.1. *Let $\mathcal{D}_m^{(dyn)}$ denote the set of m tours according to the super-node dynamic algorithm over $n \geqslant 1$ steps, resulting in super-node S_n and $\mathcal{D}_m^{(st)}$ denote the set of m tours according to the static super-node algorithm using super-node S_n. The dynamic super-node sample paths and the static super-node sample paths are equivalent in distribution, that is, $\mathbb{P}[\mathcal{D}_m^{(dyn)}(S_n) = Q] = \mathbb{P}[\mathcal{D}_m^{(st)}(S_n) = Q]$, $n \geqslant 1, \forall S_n \subset V, \forall Q$, where $m > 1$ is the number of tours.*

Proof. We prove by induction. Let $\sigma(\omega, S, \mathcal{E})$ be a deterministic function that is given an infinite random vector ω, where $\omega(1), \omega(2), \ldots \sim \text{Uniform}(0,1)$ are i.i.d. random variables, and a vector of starting nodes S and terminal nodes $\mathcal{E}$ as inputs and outputs a sample path of a random walk on the original graph G that starts at a node $u \in S$ with probability proportional to d_u and ends when it reaches any node in $\mathcal{E}$.

In what follows I_i denotes a set of i nodes as well as a vector of i nodes, $i \geqslant 1$. We add an arbitrary node outside I_i, $v \in V \backslash I_i$, into the first position $I_{i+1} = (v, I_i)$ and consider the deterministic sample path function:

$$\sigma^{(\text{dyn})}(\omega, I_i, v) = \begin{cases} (v, \sigma') & \text{, if } \omega(1) \leqslant d_v/\text{vol}(I_{i+1}) \\ \sigma(\omega', I_i, I_{i+1}) & \text{, otherwise,} \end{cases}$$

where $\text{vol}(S) = \sum_{t \in S} d_t$, $\sigma' = \sigma((\omega(2), \ldots), \{v\}, I_{i+1})$, $\omega' = ((\omega(1) - p_v)/(1 - p_v), \omega(2), \ldots)$, with $p_v = d_v/\text{vol}(I_{i+1})$. Note that by construction

$$\mathcal{D}_m^{(\text{st})}(I_i) = \{\sigma(\omega_k, I_i, I_i) : k = 1, \ldots, m, |\sigma(\omega_k, I_i, I_i)| > 2\}$$

and if we aggregate the nodes I_i into a single super-node S_i these are independent sample paths of a random walk on the super-node graph $G'(I_i)$ starting from super-node S_i. Similarly, if the choice of nodes in I_i are independent of the random vectors $\{\omega_k\}_{k=1}^m$ then

$$\mathcal{D}_m^{(\text{dyn})}(I_i) = \{r : r := \sigma^{\text{dyn}}(\omega_k, I_{i-1}, u), k = 1, \ldots, m, |r| > 2\},$$

where $u \in I_i \backslash I_{i-1}$, and $\mathcal{D}_m^{(\text{dyn})}(I_i)$ are the sample paths of the random walk described by the dynamic super-node algorithm with node addition sequence I_i.

Our proof is by induction on i. For $S_1 = \{v\}$, $v \in V$ it is clear that $\sigma^{\text{dyn}}(\omega, \varnothing, v) = \sigma(\omega, I_1, I_1)$, $\forall \omega$. By induction assume that $\sigma^{\text{dyn}}(\omega, I_{i-1}, u) = \sigma(\omega, I_i, I_i)$, $\forall \omega$, $i \geqslant 2$ and $u \in I_i \backslash I_{i-1}$. By construction the only possible difference between the sample paths of σ and σ^{dyn} is how they select the sample paths starting with u, the first node in the vector I_i. But by our induction assumption these two deterministic functions are equivalent and u is selected with probability $d_u/\mathrm{vol}(I_i)$. Thus, using the same deterministic rule σ selects v, the first element of vector I_{i+1}, with probability $d_v/\mathrm{vol}(I_{i+1})$ making $\sigma^{\mathrm{dyn}}(\omega, I_i, v) = \sigma(\omega, I_{i+1}, I_{i+1})$ and yielding

$$\mathbb{P}[\mathcal{D}_m^{(\mathrm{dyn})}(I_n) = Q] = \mathbb{P}[\mathcal{D}_m^{(\mathrm{st})}(I_n) = Q], \quad \forall n, I_n, Q.$$

To finish the proof note that for $v \in I_{i+1} \backslash I_i$, the the deterministic rule σ guarantees that we are selecting tours starting from v with probability $d_v/\mathrm{vol}(I_{i+1})$. This rule is equivalent to the dynamic super-node algorithm that starts k_{redo} tours from v once v is added to the super-node, where k_{redo} is a negative binomial random variable with success probability $\mathrm{vol}(I_i)/\mathrm{vol}(I_{i+1})$. The success probability in the algorithm is $d_{I_i}/d_{I_{i+1}}$ because by definition the algorithm, like our definition of $\mathcal{D}_m^{(\mathrm{dyn})}(I_{i+1})$, disregards tours of size two which only contain nodes in I_{i+1}. □

4.4 Frequentist approach

In what follows we present our main results for the estimators.

4.4.1 Estimator of $\mu(G)$

Theorem 4.2 below proposes an unbiased estimator of $\mu(G)$ in equation (4.1) via random walk tours. Later in Section 4.5 we present the approximate posterior distribution of this unbiased estimator.

To compute an estimate of $\mu(G)$ using super-node tours we define a function f and a set H over the modified graph $G'(I_n)$ as follows.

(a) If $\forall (u, v) \in E$, s.t. $u \in I_n$, $v \in V \backslash I_n$ the function $g(u, v)$ can be obtained with little overhead (using extra API queries to find all the neighbors of $u \in I_n$ and further querying them to find the function $g(u, v)$), then we define f as

$$f(u, v) = \begin{cases} g(u, v) & \text{, if } u \neq S_n, v \neq S_n \\ 0 & \text{, if } u \text{ or } v = S_n \text{ .} \end{cases} \tag{4.2}$$

Define $H = \{(u, v) : (u, v) \in E \text{ s.t. } u \in I_n \textbf{ or } v \in I_n\}$.

(b) Otherwise we define f as

$$f(u, v) = \begin{cases} g(u, v) & \text{if } u \neq S_n, v \neq S_n \\ \frac{1}{k_{xS}} \sum_{w \in I_n} g(u, w) & \text{if } u \text{ or } v = S_n, \end{cases} \tag{4.3}$$

where $x = u$ if $u \neq S_n, x = v$ if $v \neq S_n$ and k_{yS} is the number of neighbors of node $y \in V \backslash I_n$ that are in I_n. Define $H = \{(u, v) : (u, v) \in E \text{ s.t. } u, v \in I_n\}$.

The notation d_{S_n} indicates the degree of super-node which is defined as

$$d_{S_n} = |\{(u, v) : (u, v) \in E \text{ s.t. } u \in I_n, v \notin I_n\}|.$$

Let $\mu(G')$ be contribution from G' to the true value $\mu(G)$ and $\mu(G') = \sum\limits_{(u,v)\in E'} f(u,v)$.

Theorem 4.2. *Let G be an unknown undirected graph where $n > 0$ initial* arbitrary *set of nodes is known $I_n \subseteq V$ which span all the different connected components of G. Consider a random walk on the augmented multigraph G' described in Section 4.2 starting at super-node S_n. Let $(X_t^{(k)})_{t=1}^{\xi_k}$ be the k-th random walk tour until the walk first returns to S_n and let $\mathcal{D}_m(S_n)$ denote the collection of all nodes in $m \geqslant 1$ such tours, $\mathcal{D}_m(S_n) = \left((X_t^{(k)})_{t=1}^{\xi_k}\right)_{k=1}^{m}$. Then,*

$$\hat{\mu}(\mathcal{D}_m(S_n)) = \overbrace{\frac{d_{S_n}}{2m}\sum_{k=1}^{m}\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})}^{\textit{Estimate from crawls}} \quad \overbrace{+\sum_{(u,v)\in H} g(u,v)}^{\substack{\textit{Given knowledge}\\ \textit{from nodes in } I_n}}, \tag{4.4}$$

is an unbiased estimate of $\mu(G)$, i.e., $\mathbb{E}[\hat{\mu}(\mathcal{D}_m(S_n))] = \mu(G)$. Moreover the estimator is strongly consistent, i.e., $\hat{\mu}(\mathcal{D}_m(S_n)) \to \mu(G)$ a.s. for $m \to \infty$.

Note that the notation for the estimator $\hat{\mu}(\mathcal{D}_m(S_n))$ is different from what we have defined in Chapter 1. This is to make distinction between the number of tours m, size of the super-node n, and data from the tours $\mathcal{D}_m(S_n)$.

The proof of Theorem 4.2 is provided in Section 4.9. Theorem 4.2 provides an unbiased estimate of network statistics from random walk tours. The length of tour k is short if it starts at a massive super-node as the expected tour length is inversely proportional to the degree of the super-node, $\mathbb{E}[\xi_k] \propto 1/d_{S_n}$. This provides a practical way to compute unbiased estimates of node, edge, and triangle statistics using $\hat{\mu}(\mathcal{D}_m(S_n))$ (eq. (4.4)) while observing only a small fraction of the original graph. Because random walk tours can have arbitrary lengths, we show in Lemma 4.4, Section 4.4.4, that there are upper and lower bounds on the variance of $\hat{\mu}(\mathcal{D}_m(S_n))$. For a bounded function f, the upper bounds are shown to be always finite.

4.4.2 Confidence interval of the estimator

In what follows we give confidence intervals for the estimator presented in Theorem 4.2. Let

$$\bar{f}_m = m^{-1}\sum_{k=1}^{m}\left(\frac{d_{S_n}}{2}\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\right),$$

$$\hat{\sigma}_m^2 = m^{-1}\sum_{k=1}^{m}\left(\frac{d_{S_n}}{2}\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}) - \bar{f}_m\right)^2.$$

If $\Phi(x)$ is the CDF of the standard Gaussian distribution, then for a known constant $c > 0$ [Bentkus & Götze 1996]:

$$\sup_x \left|\mathbb{P}\left\{\sqrt{m}\left(\frac{\bar{f}_m - \mu(G')}{\hat{\sigma}_m}\right) < x\right\} - \Phi(x)\right| \leqslant \frac{c\beta}{\sigma^3\sqrt{m}}, \tag{4.5}$$

where

$$\beta = \mathbb{E}\left[\left|\frac{d_{S_n}}{2}\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}) - \mu(G')\right|^3\right],$$

$$\sigma^2 = \text{Var}\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\right].$$

Moreover, $\sigma^2 < \infty$ for a bounded function f as we will prove in Lemma 4.4(i) in Section 4.4.4 and $\beta < \infty$ through the C_r inequality [Gut 2012, Chapter 3, Theorem 2.2].

Therefore, with $\varepsilon > 0$ and large m (number of tours) within the confidence interval $[\hat{\mu}(\mathcal{D}_m(S_n)) - \varepsilon, \hat{\mu}(\mathcal{D}_m(S_n)) + \varepsilon]$ yields

$$\mathbb{P}\left(|\mu(G) - \hat{\mu}(\mathcal{D}_m(S_n))| \leqslant \varepsilon\right) \approx 1 - 2\Phi\left(\frac{\varepsilon\sqrt{m}}{\hat{\sigma}_m}\right).$$

with the rate of convergence given by equation (4.5).

4.4.3 Estimation and hypothesis testing in random graph models

Here we study $\mu(G)$ when the connections in graph G are made randomly while keeping the node attributes and node degrees the same as in the observed graph. Two types of random graph generation are considered here: configuration model and Chung-Lu model. These models can be regarded as null hypotheses in graph hypothesis testing problem. First we estimate the expected value $\mathbb{E}[\mu(G)]$ in these random graph models. Then we seek, *with how much certainty* the value $\mu(G)$ of the observed graph could possibly belong to a random network with the same node attributes and degrees as that of the observed graph, all this with partial knowledge of the original graph.

4.4.3.1 Estimators for configuration model and Chung-Lu random graphs

Configuration model is an important class of random graph models. For a given degree sequence over the nodes, the configuration model creates random edge connections by uniformly selecting pairs of half edges. We assume that the number of nodes $|V|$ and number of edges $|E|$ are known (The estimation of $|V|$ and $|E|$ can be done explicitly, for instance using the techniques in [Massoulié et al. 2006] and [Cooper et al. 2013]). The probability that the nodes u and v are connected in the configuration model is $d_u d_v/(2|E|-1)$ if $u \neq v$ and the probability of a self-edge from node u to itself is $\binom{d_u}{2}/(2|E|-1)$.

Another important model, Chung-Lu random graph [Chung & Lu 2003] is a generalized version of Erdős-Renyi graphs and is closely related to configuration model. Chung-Lu model takes the positive weights $w_1, \ldots, w_{|V|}$ corresponding to nodes $1, \ldots, |V|$ as input and generates a graph with average degrees as these weights. The edges are created between any two vertices u and v independently of all others with probability $w_u w_v / \sum_{k=1}^{|V|} w_k$, when $u \neq v$, and for the self loops at node u, with a probability $\binom{w_u}{2} / \sum_{k=1}^{n} w_k$.

In the case of Chung-Lu random graphs, from [Avrachenkov et al. 2015a], it is known that the weights $w_1, \dots, w_{|V|}$ in fact becomes the actual degrees $d_1, \dots, d_{|V|}$ asymptotically and the following concentration bound exists: for $c > 0$,

$$\mathbb{P}\left(\max_{1 \leqslant i \leqslant |V|} \left| \frac{d_i}{w_i} - 1 \right| \geqslant \beta \right) \leqslant \frac{2}{|V|^{c/4} - 1}, \text{ if } \beta \geqslant \frac{c \log |V|}{w_{\min}} = o(1).$$

Thus we take the sequence $\{w_k\}$ as $\{d_k\}$ of the observed graph. One main advantage in using Chung-Lu model compared to configuration model is that the edges are independent to each other.

For brevity we will use G_{obs} for the observed graph, G_{conf} for an instance of the configuration model with the same degree sequence $\{d_k\}$ as that of G and $G_{\text{C-L}}$ for the Chung-Lu graph sample with weights as $\{d_k\}$. Note that $\mu(G_{\text{conf}})$ and $\mu(G_{\text{C-L}})$ are random variables. Thus we look for $\mathbb{E}[\mu(G_{\text{conf}})]$ and $\mathbb{E}[\mu(G_{\text{C-L}})]$, where the expectation is with respect to the probability distribution of the configuration model and Chung-Lu model respectively. The values of $\mathbb{E}[\mu(G_{\text{C-L}})]$ and $\mathbb{E}[\mu(G_{\text{conf}})]$ are nearly the same, but for higher moments the values are different since configuration model introduces correlation between edges.

Now the expected value in the Chung-Lu model is

$$\mathbb{E}[\mu(G_{\text{C-L}})] = \sum_{\substack{(u,v) \in E \cup E^c \\ u \neq v}} g(u,v) \frac{d_u d_v}{2|E|} + \sum_{\substack{(u,v) \in E \cup E^c \\ u = v}} g(u,v) \frac{\binom{d_u}{2}}{2|E|}. \tag{4.6}$$

In order to calculate $\mathbb{E}[\mu(G_{\text{C-L}})]$, we need to know the missing edge set E^c. The set E^c is revealed once the entire graph is crawled, which is not possible in the context of this work. The idea is to estimate $\mathbb{E}[\mu(G_{\text{C-L}})]$ from the tours of a random walk. Since the classical random walk which we have used so far, could sample only from E, we resort to a new random walk that could make samples from E^c as well.

We use random walk with uniform restarts (RWuR) [Avrachenkov et al. 2010] in which if the crawler is at a node i, with a probability $d_i/(d_i + \alpha)$ the crawler chooses one of the neighbors uniformly (RW strategy) and with a probability $\alpha/(d_i + \alpha)$, the crawler chooses the next node by uniformly sampling all the nodes. The parameter $\alpha > 0$ controls the rate of uniform sampling (which has higher cost in many OSNs).

Define a new function f' whose value depends on the crawling strategy as follows:

let u, v be the nodes chosen by the crawling technique (RWuR or RW) in order,

$$f'(u,v) = \begin{cases} g(u,v)\dfrac{d_u d_v}{2|E|-1} & \text{if } u,v \neq S_n \\ \sum_{w\in I_n} g(u,w)\dfrac{d_u d_w}{2|E|-1} & \text{if } u \neq S_n, v = S_n \\ & (u,v) \text{ selected} \\ & \text{by unif. sampling} \\ \dfrac{1}{k_{uS}}\sum_{w\in I_n} g(u,w)\dfrac{d_u d_w}{2|E|-1} & \text{if } u \neq S_n, v = S_n \\ & (u,v) \text{ selected by RW} \end{cases} \tag{4.7}$$

where k_{uS} is defined in (4.3). In the new graph G' there will be k_{uS} multiple edges between u and S_n and k_{uS} is introduced in the last term is to take into account this. In case of classical random walk, the second case does not exist.

We denote $W'_k = \sum_{t=2}^{\xi_k} f'(X_{t-1}^{(k)}, X_t^{(k)})$ when RWuR is employed as the random walk technique for crawling the graph and $W''_k = \sum_{t=2}^{\xi_k} f'(X_{t-1}^{(k)}, X_t^{(k)})$, when the classical random walk is used for crawling. Let

$$R = \frac{1}{2}\sum_{\substack{(u,v)\in I_n\times I_n \\ u\neq v}} g(u,v)\frac{d_u d_v}{(2|E|-1)} + \frac{1}{2}\sum_{\substack{(u,v)\in I_n\times I_n \\ u=v}} g(u,v)\frac{\binom{d_u}{2}}{(2|E|-1)}.$$

The value R can be calculated a priori from the knowledge of I_n. In the lemma below we propose an estimator for $\mathbb{E}[\mu(G_{\text{C-L}})]$ and proves that it is unbiased.

Lemma 4.3. *The estimator*

$$\hat{\mu}_{\text{C-L}}(\mathcal{D}_m(S_n)) = \frac{1}{m}\sum_{k=1}^{m}\Big[\frac{|V'|(d_{S_n}+\alpha)}{2\alpha}W'_k - \frac{|V'|d_{S_n}}{2\alpha}W''_k + R\Big],$$

is an unbiased estimator of $\mathbb{E}[\mu(G_{\text{C-L}})]$ *of the Chung-Lu model.*

Proof. See Section 4.10 □

4.4.3.2 A hypothesis testing problem for the Chung-Lu model

The Chung-Lu model or configuration model can be regarded as a null hypothesis model and comparing $\mu(G_{\text{obs}})$ to $\mathbb{E}[\mu(G_{\text{C-L}})]$ or $\mathbb{E}[\mu(G_{\text{conf}})]$ answers many questions like whether the connections are formed based on degrees alone with no other influence or whether the edges are formed purely at random?

Let $G_{\text{C-L}}(V_{\text{C-L}}, E_{\text{C-L}})$ be a sample of the Chung-Lu model with weights $\{d_k\}$. Like the estimator of $\mathbb{E}[\mu(G_{\text{C-L}})]$, the estimator of $\text{Var}(\mu(G_{\text{C-L}}))$, $\widehat{\text{Var}}_{\text{C-L}}(\mathcal{D}_m(S_n))$ can be constructed as follows: modify $g(u,v)\dfrac{d_u d_v}{2|E|}$ to $g^2(u,v)\dfrac{d_u d_v}{2|E|}\Big(1-\dfrac{d_u d_v}{2|E|}\Big)$ in the function f' in (4.7).

By invoking Lindeberg central limit theorem [Gut 2012, Chapter 7, Section 2], for independent non-identically distributed Bernoulli random variables[2], we get

$$\sum_{(u,v)\in E_{\text{C-L}}} f(u,v) \sim \text{Normal}\left(\mathbb{E}[\mu(G_{\text{C-L}})], \text{Var}(\mu(G_{\text{C-L}}))\right).$$

Hence a natural check is whether $\mu(G_{\text{obs}})$ is a sample from the above distribution. Since we have only one sample $\mu(G_{\text{obs}})$, a simple test is to check whether

$$|\hat{\mu}(\mathcal{D}_m(S_n)) - \hat{\mu}_{\text{C-L}}(\mathcal{D}_m(S_n))| \leqslant a\sqrt{\widehat{\text{Var}}_{\text{C-L}}(\mathcal{D}_m(S_n))},$$

holds for any $a = 1, 2, 3$ and for a large value of m . This condition is satisfied by a Gaussian sample with probabilities 0.6827, 0.9545, or 0.9973 respectively. On the other hand, the lower a is, the more certain that the sample belongs to this particular Gaussian distribution.

4.4.4 Impact of spectral gap on variance

In this section we derive results on higher moments of the estimator $\hat{\mu}(\mathcal{D}(S_n))$. Lemma 4.4, which follows, introduces upper and lower bounds on the variance of the i.i.d. the tour sum $\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})$, and also shows that all the moments exist. Moreover, the results in the lemma establish a connection between the estimator variance and the spectral gap.

Let $\boldsymbol{S} = \tilde{\boldsymbol{D}}^{1/2}\boldsymbol{P}\tilde{\boldsymbol{D}}^{-1/2}$, where $\boldsymbol{P} = \{p_{uv}\}$ is the random walk transition probability matrix as defined in Section 4.2 and $\tilde{\boldsymbol{D}} = \text{diag}(d_1, d_2, \ldots, d_{|V'|})$ is a diagonal matrix with the node degrees of G'. The eigenvalues $\{\lambda_i\}$ of $\boldsymbol{P}$ and $\boldsymbol{S}$ are same and $1 = \lambda_1 > \lambda_2 \geqslant \ldots \geqslant \lambda_{|V'|} \geqslant -1$. Let jth eigenvector of $\boldsymbol{S}$ be $(w_{ji}), 1 \leqslant i \leqslant |V|$. Let δ be the spectral gap, $\delta := 1 - \lambda_2$. Let the left and right eigenvectors of $\boldsymbol{P}$ be v_j and u_j respectively. $d_{tot} := \sum_{v\in V'} d_v$. Define $\langle \boldsymbol{r}, \boldsymbol{s}\rangle_{\hat{\pi}} = \sum_{(u,v)\in E'} \hat{\boldsymbol{\pi}}_{uv}\boldsymbol{r}(u,v)\boldsymbol{s}(u,v)$, with $\hat{\boldsymbol{\pi}}_{uv} = \boldsymbol{\pi}_u p_{uv}$, and matrix $\boldsymbol{P}^*$ with (j,i)th element as $p^*_{ji} = p_{ji}f(j,i)$. Also let $\hat{f}$ be the vector with $\hat{f}(j) = \sum_{i\in V'} p^*_{ji}$.

Lemma 4.4. *The following holds*

(i). Assuming the function f is bounded, $\max_{(i,j)\in E'} f(i,j) \leqslant B < \infty$, $B > 0$ and for tour $k \geqslant 1$,

$$\begin{aligned}\text{Var}&\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\right] \\ &\leqslant \frac{1}{d_{S_n}^2}\left(2d_{tot}^2 B^2 \sum_{i\geqslant 2} \frac{w_{S_n i}^2}{(1-\lambda_i)} - 4\mu^2(G_{S_n})\right) - \frac{1}{d_{S_n}} B^2 d_{tot} + B^2 \\ &< B^2\left(\frac{2d_{tot}^2}{d_{S_n}^2\delta} + 1\right).\end{aligned}$$

[2] The necessary condition to hold this central limit theorem, the Lindeberg condition is satisfied by the sequence of independent Bernoulli random variables with different success probabilities $\{p_k\}$, if $0 < p_k < 1$. This is always true in our case when we assume $d_k > 0$ for all k.

Moreover,

$$\mathbb{E}\left[\left(\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\right)^l\right] < \infty \quad \forall l \geqslant 0.$$

(ii).

$$\begin{aligned}
&\mathrm{Var}\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}))\right] \\
&\qquad \geqslant 2\frac{d_{tot}}{d_{S_n}} \sum_{i=2}^{r} \frac{\lambda_i}{1-\lambda_i} \langle f, v_i\rangle_{\hat{\pi}}\,(u_i^\intercal \hat{f}) + \frac{1}{d_{S_n}} \sum_{(u,v)\in E'} f(u,v)^2 \\
&\qquad + \frac{1}{d_{tot} d_{S_n}} \left\{\sum_{(u,v)\in E'} f(u,v)^2\right\}^2 + \frac{1}{d_{tot} d_{S_n}} \sum_{u\in V'} d_u \Big\{\sum_{u\sim v} f(u,v)\Big\}^2 \\
&\qquad - \frac{4}{d_{S_n}^2} \left\{\sum_{(u,v)\in E'} f(u,v)\right\}^2 - \frac{8}{d_{tot}} \left\{\sum_{(u,v)\in E'} f(u,v)\right\}^2 \sum_{i\geqslant 2} \frac{w_{S_n i}^2}{(1-\lambda_i)} \\
&\qquad - \frac{4}{d_{tot} d_{S_n}} \left\{\sum_{(u,v)\in E'} f(u,v)\right\}^2. \qquad (4.8)
\end{aligned}$$

Proof. See Section 4.11. □

4.5 Bayesian Approach

In this section we consider Bayesian formulation of our problem and derive the posterior of $\mu(G)$ given the tours and provide a consistent maximum a posteriori estimator (MAP).

Approximate posterior

For the same scenario of Theorem 4.2 for $m \geqslant 2$ tours let

$$\hat{F}_h = \frac{d_{S_n}}{2\lfloor\sqrt{m}\rfloor} \sum_{k=((h-1)\lfloor\sqrt{m}\rfloor+1)}^{h\lfloor\sqrt{m}\rfloor} \sum_{t=2}^{\xi_h} f(X_{t-1}^{(k)}, X_t^{(k)}) + \sum_{(u,v)\in H} g(u,v)\,,$$

which is similar to equation (4.4) but first sums a range of $\lfloor\sqrt{m}\rfloor$ tours rather than all m tours. Let σ_F^2 be the variance of $\hat{F}_h$. Assuming priors

$$\begin{aligned}
\mu(G)|\sigma_F^2 &\sim \mathrm{Normal}(\mu_0, \sigma_F^2/m_0) \\
\sigma_F^2 &\sim \text{Inverse-gamma}(\zeta_0/2, \zeta_0\sigma_0^2/2),
\end{aligned}$$

then for large values of m, the marginal posterior density of $\mu(G)$ can be approximated by a *non-standardized t -distribution*

$$\phi(x|\zeta, \tilde{\mu}, \tilde{\sigma}) = \frac{\Gamma\left(\frac{\zeta+1}{2}\right)}{\Gamma\left(\frac{\zeta}{2}\right)\,\tilde{\sigma}\sqrt{\pi\zeta}} \left(1 + \frac{(x-\tilde{\mu})^2}{\tilde{\sigma}^2\zeta}\right)^{-\frac{\zeta+1}{2}}\,, \qquad (4.9)$$

with degrees of freedom parameter

$$\zeta = \zeta_0 + \lfloor\sqrt{m}\rfloor,$$

location parameter

$$\widetilde{\mu} = \frac{m_0\mu_0 + \lfloor\sqrt{m}\rfloor\hat{\mu}(\mathcal{D}_m(S_n))}{m_0 + \lfloor\sqrt{m}\rfloor},$$

and scale parameter

$$\widetilde{\sigma}^2 = \frac{\zeta_0\sigma_0^2 + \sum_{k=1}^{\lfloor\sqrt{m}\rfloor}(\hat{F}_k - \hat{\mu}(\mathcal{D}_m(S_n)))^2 + \frac{m_0\lfloor\sqrt{m}\rfloor(\hat{\mu}(\mathcal{D}_m(S_n))-\mu_0)^2}{m_0+\lfloor\sqrt{m}\rfloor}}{(\zeta_0 + \lfloor\sqrt{m}\rfloor)(m_0 + \lfloor\sqrt{m}\rfloor)}.$$

The derivation is detailed in Section 4.5.1 later.

Remark 4.5. Note that the approximation (4.9) is Bayesian and Theorem 4.2 is its frequentist counterpart. In fact, the motivation of our Bayesian approach comes from the frequentist estimator. From the approximate posterior in (4.9), the Bayesian MAP estimator is

$$\hat{\mu}_{\text{MAP}} = \arg\max_x \phi(x|v, \widetilde{\mu}, \widetilde{\sigma}) = \widetilde{\mu}.$$

Thus for large values of m, the Bayesian estimator $\hat{\mu}_{\text{MAP}}$ is essentially the frequentist estimator $\hat{\mu}(\mathcal{D}_m(S_n))$, which is unbiased, and hence the MAP estimator is consistent.

The above remark shows that the approximate posterior in (4.9) provides a way to access the confidence in the estimate $\hat{\mu}(\mathcal{D}_m(S_n))$. The Normal prior for the average gives the largest variance given a given mean. The inverse-gamma is a non-informative conjugate prior if the variance of the estimator is not too small [Gelman 2006], which is generally the case in our application. Other choices of prior, such as uniform, are also possible yielding different posteriors without closed-form solutions [Gelman 2006].

Remark 4.6. Another asymptotic result in Bayesian analysis, the classical Bernstein-von Mosses Theorem [Van der Vaart 2000, Chapter 10] is not useful in our scenario. The Bernstein-von Mosses Theorem states that irrespective of the prior distribution, when μ is the random parameter of likelihood, then posterior distribution of $\sqrt{m}(\mu - \hat{\mu}_m^{\text{MLE}})$ converges to Normal$(0, I(\mu_0)^{-1})$, where $\hat{\mu}_m^{\text{MLE}}$ is the maximum likelihood estimator (MLE) and $I(\mu_0)$ is the Fisher information at the true value μ_0. But note that in cases like ours, where the distribution of $W_k = \sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})$ is unknown, $k \geqslant 1$, the Fisher information is also unknown. In contrast, our approximate posterior of $\mu(G)$ uses only the available information and does not need to guess the distribution of W_k.

4.5.1 Derivation of approximate posterior

In this section we derive the approximation (4.9) of the posterior. The approximation relies first on showing that $\hat{\mu}(\mathcal{D}_m(S_n))$ has finite second moment. By Lemma 4.4 the variance of $\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})$, $k \geqslant 1$, is also finite.

We are now ready to give the approximation in equation (4.9). Let $m' = \lfloor \sqrt{m} \rfloor$,

$$\hat{F}_h = \frac{d_{S_n}}{2m'} \sum_{k=((h-1)m'+1)}^{hm'} \sum_{t=2}^{\xi_h} f(X_{t-1}^{(k)}, X_t^{(k)}) + \sum_{(u,v)\in H} g(u,v)\,.$$

and $\{\hat{F}_h\}_{h=1}^{m'}$ and because the tours are i.i.d. the marginal posterior density of μ is

$$\mathbb{P}[\mu|\{\hat{F}_h\}_{h=1}^{m'}] = \int_0^\infty \mathbb{P}[\mu|\sigma_F^2, \{\hat{F}_h\}_{h=1}^{m'}]\mathbb{P}[\sigma_F^2|\{\hat{F}_h\}_{h=1}^{m'}]d\sigma_F^2\,.$$

For now assume that $\{\hat{F}_h\}_{h=1}^{m'}$ are i.i.d. normally distributed random variables, and let

$$\hat{\sigma}_{m'} = \sum_{h=1}^{m'} (\hat{F}_h - \hat{\mu}(\mathcal{D}_m(S_n)))^2,$$

then [Jackman 2009, Proposition C.4]

$$\mu|\sigma_F^2, \{\hat{F}_h\}_{h=1}^{m'} \sim \text{Normal}\left(\frac{m_0\mu_0 + \sum_{h=1}^{m'} \hat{F}_h}{m_0 + m'}, \frac{\sigma_F^2}{m_0+m'}\right),$$

$$\sigma_F^2|\{\hat{F}_h\}_{h=1}^{m'} \sim \text{Inverse-Gamma}\left(\frac{\zeta_0 + m'}{2}, \frac{\zeta_0\sigma_0^2 + \hat{\sigma}_{m'} + \frac{m_0 m'}{m_0+m'}(\mu_0 - \hat{\mu}(\mathcal{D}_m(S_n)))^2}{2}\right),$$

are the posteriors of parameters μ and σ_F^2, respectively. The non-standardized t-distribution can be seen as a mixture of normal distributions with equal mean and random variance inverse-gamma distributed [Jackman 2009, Proposition C.6]. Thus, if $\{\hat{F}_h\}_{h=1}^{m'}$ are i.i.d. normally distributed then the posterior of $\mu(G)$ given $\mathcal{D}_{m'}(S_n)$ is a non-standardized t-distributed with parameters

$$\tilde{\mu} = \frac{m_0\mu_0 + \sum_{h=1}^{m'} \hat{F}_h}{m0 + m'}, \tilde{\sigma}^2 = \frac{\zeta_0\sigma_0^2 + \sum_{k=1}^{m'}(\hat{F}_k - \hat{\mu}(\mathcal{D}_m(S_n)))^2 + \frac{m_0 m'(\hat{\mu}(\mathcal{D}_m(S_n))-\mu_0)^2}{m_0+m'}}{(\zeta_0+m')(m_0+m')}, \quad \zeta = \zeta_0 + m'$$

where $\tilde{\mu}, \tilde{\sigma}^2$ and ζ are location, scale and degree of freedom parameters of the student-t distribution. Left to show is that $\{\hat{F}_h\}_{h=1}^{m'}$ converge *in distribution* to i.i.d. normal random variables as $m \to \infty$. As the spectral gap of $G'(I_n)$ is greater than zero, $|\lambda_1 - \lambda_2| > 0$, Lemma 4.4 shows that for $W_k = \sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})$,

$\sigma_W^2 = \text{Var}(W_k) < \infty$, $\forall k$. From the renewal theorem we know that $\{W_k\}_{k=1}^m$ are i.i.d. random variables and thus any subset of these variables is also i.i.d.. By construction $\hat{F}_1, \ldots, \hat{F}_{m'}$ are also i.i.d. with mean $\mu(G)$ and finite variance. Applying the Lindeberg-Lévy central limit theorem [Cramér 1946, Section 17.4] yields

$$\sqrt{m'}(\hat{F}_h - \mu(G)) \xrightarrow{d} \text{Normal}(0, \sigma_W^2), \quad \forall h\,.$$

Thus, for large values of m (recall that $m' = \lfloor \sqrt{m} \rfloor$), $\{\hat{F}_h\}_{h=1}^{m'}$ are approximately i.i.d. normally distributed

$$\hat{F}_h \sim \text{Normal}(\mu(G), \sigma_W^2/m')\,, \quad \forall h\,.$$

This completes the derivation of the approximation (4.9). In what follows we present our results over real-world networks.

4.6 Experiments on Real-world Networks

In this section we demonstrate the effectiveness of the theory developed with the experiments on real data sets of various social networks. We have chosen to work with the datasets where the value $\mu(G)$ is available. This way it is possible to check the correctness of the results obtained via experiments. We assume the contribution from super-node to the true value is known a priori and hence we look for $\mu(G')$ in the experiments. In the case that the edges of the super-node are unknown, the estimation problem is easier and can be taken care of separately.

In the figures we display the approximate posterior generated from $\hat{F}_h$ with only one run of the experiment and empirical posterior created from multiple runs. For the approximate posterior, we have used the following parameters $m_0 =, \zeta_0 = 0, \mu_0 = 0.1, \sigma_0 = 1$ (see (4.9)). The green line in the plots shows the actual value $\mu(G')$.

We have used the dynamic super-node algorithm explained in Section 4.3.2. From the experiments, it is observed that the static super-node and dynamic super-node produces similar results which is in corroboration with Theorem 4.1. In the experiments, we opt a simple strategy to decide when to run super-node recruiting walk: run the super-node recruiting walk when the number of original tours reaches multiples of a fixed integer and it stops when a node of degree exceeding a specific threshold is reached.

4.6.1 Friendster

First we study a network of moderate size, a connected subgraph of Friendster network with 64,600 nodes and with 1,246,479 edges (data publicly available at the SNAP repository [Leskovec & Krevl 2014]). Friendster is an online social networking website where nodes are the individuals and edges indicate friendship. We consider two types of functions:

1. $f_1 = d_{X_t}.d_{X_{t+1}}$

2. $f_2 = \begin{cases} 1 \text{ if } d_{X_t} + d_{X_{t+1}} > 50 \\ 0 \text{ otherwise} \end{cases}$

These functions reflect assortative nature of the network. The final super-node size is 10,000. Figures 4.2 and 4.3 display the results for functions f_1 and f_2, respectively. A good match between the approximate and empirical posteriors can be observed from the figures. Moreover the true value $\mu(G')$ is also fitting well with the plots. The percentage of graph crawled is 7.43% in terms of edges and is 18.52% in terms of nodes.

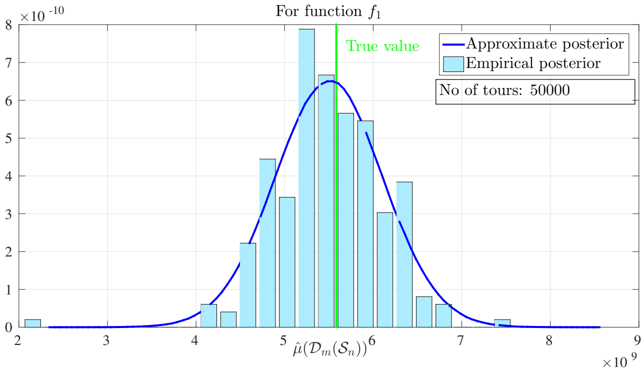


Figure 4.2: Friendster subgraph, function f_1

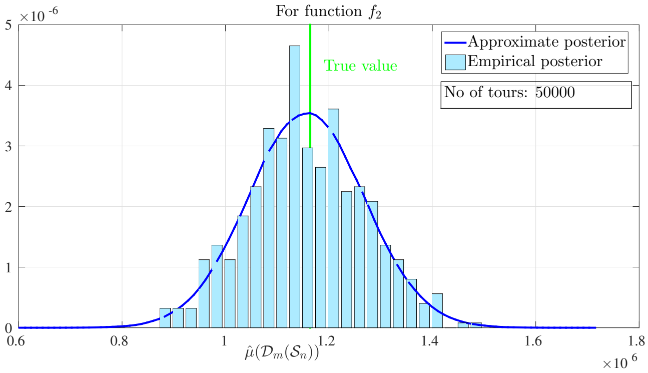


Figure 4.3: Friendster subgraph, function f_2

4.6.2 Dogster network

The aim of this example is to check whether there is any affinity for making connections between the owners of same breed dogs [Dogster & friendships network dataset KONECT 2015]. The network data is based on the social networking website Dogster. Each user (node) indicates the dog breed; the friendships between dogs' owners form the edges. Number of nodes is 415,431 and number of edges is 8,265,511.

In Figure 4.4, two cases are plotted. Function f_1 counts the number of connections with different breed pals and function f_2 counts connections between same breed pals. The final super-node size is 10,000. The percentage of the graph crawled in terms of edges is 2.72% and in terms of nodes is 14.86%. While using the static super-node technique with uniform sampling, the graph crawled is 5.02% (in terms of edges) and 37.17% (in terms of nodes) with the same super-node size. These values can be reduced much further if we allow a bit less precision in the match between approximate distribution and histogram.

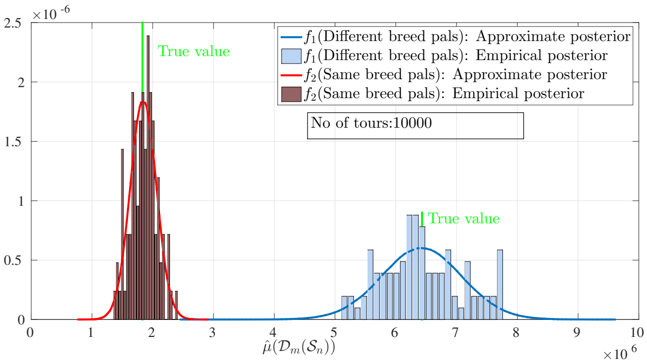


Figure 4.4: Dog pals network

In order to better understand the correlation in forming edges, we now consider the *configuration model*. We use the estimator $\hat{\mu}_C(\mathcal{D}_m(S_n))$ proposed in Section 4.4.3.1 (note that the estimator is same for Chung-Lu model and configuration model). It is important to recollect that this estimator does not require the knowledge of the complete network (in fact the Figures 4.5 and 4.6 are based on estimates from RWuR crawls which covered 8.9% of the graph). This is shown in blue line in Figure 4.5 and red line in Figure 4.6, and the true value is the net expected value given by (4.6). Moreover we run our original estimator $\hat{\mu}(\mathcal{D}_m(S_n))$ and calculated the approximate posterior on one random instance of the configuration model with same degree sequence of the original graph. Figure 4.5 compare function f_2 for the configuration model and original graph. The figure shows that in the correlated case (original graph), the affinity to form connection between same breed owners is around 7.5 times more than that in the uncorrelated case. Figure 4.6 shows similar figure in case of f_1.

4.6.3 ADD Health data

Though our main result in the approximation (4.9) holds when the number of tours is large, in this section we check with a small dataset. We consider ADD network project[3], a friendship network among high school students in the US. The graph has 1545 nodes and 4003 edges.

We take two types of functions. Figure 4.7 shows the affinity in same gender or different gender friendships and Figure 4.8 displays the inclination towards same race or different race in friendships. The random walk tours covered around 10% of the graph. We find that the theory works reasonably well for this network data, for

[3] http://www.cpc.unc.edu/projects/addhealth

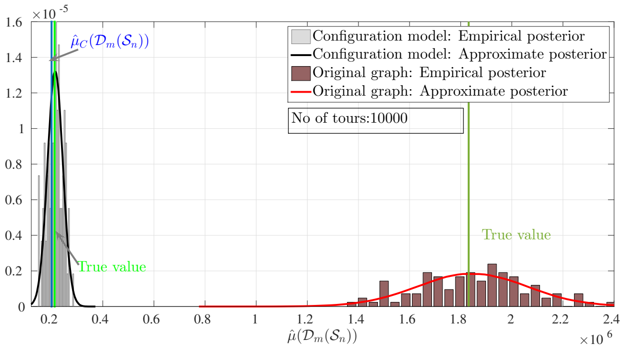


Figure 4.5: Dog pals network: Comparison between configuration model and original graph for f_2, number of connection between same breeds

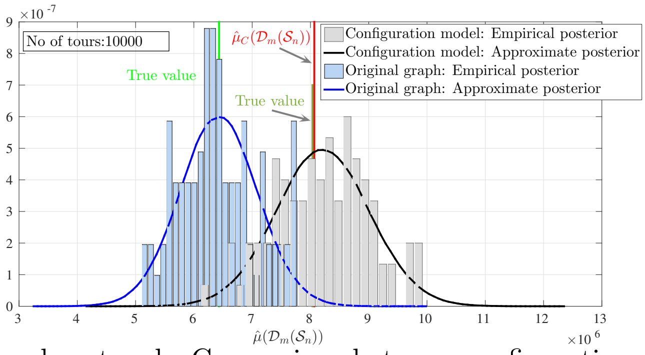


Figure 4.6: Dog pals network: Comparison between configuration model and original graph for f_1, number of connection between different breeds

instance, the true values in both the cases in Figure 4.7 are nearly the same, and this is evident from the approximate posterior calculated from only one run. We have not added the empirical posterior in the figures since for such small sample sizes, the empirical distribution does not lead to a meaningful histogram.

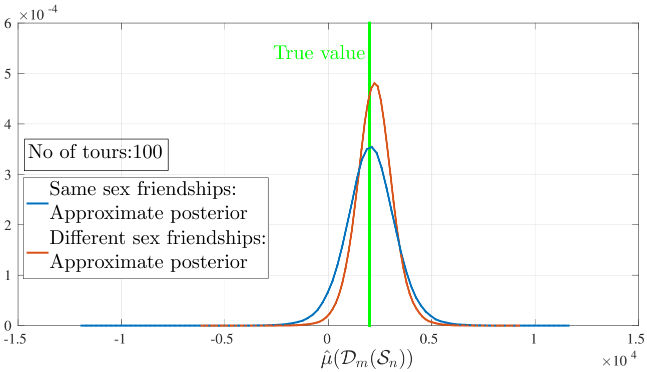


Figure 4.7: ADD network: effect of gender in relationships

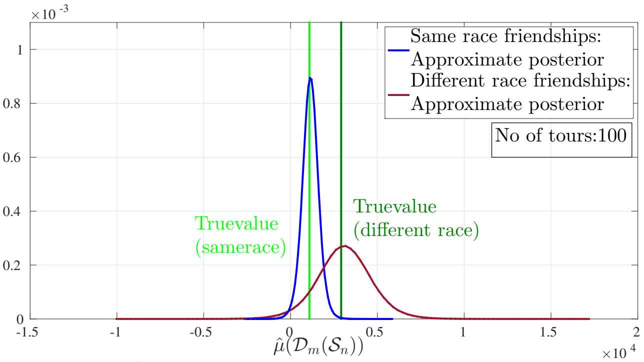


Figure 4.8: ADD network: effect of race in friendships

4.6.4 Check for Chung-Lu random graph model in Dogester

We use the same dataset and functions as in Section 4.6.2. Consider the function f_1, which is one when the owners of different breed dogs form connection, zero otherwise. For the Chung-Lu model, $\hat{\mu}_C(\mathcal{D}_m(S_n))$, the estimator of $\mathbb{E}[\mu(G_{\text{C-L}})]$ is 8.066×10^6 and $\widehat{\text{Var}}_C(\mathcal{D}_m(S_n))$, the estimator of $\text{Var}_C[\mu(G_{\text{C-L}})]$ is 6.3938×10^{11}. For the original graph, the estimated value $\mu(\mathcal{D}_m(S_n)) = 6.432 \times 10^6$. Now

$$|\hat{\mu}_C(\mathcal{D}_m(S_n)) - \mu(\mathcal{D}_m(S_n))| \leqslant a\sqrt{\widehat{\text{Var}}_C(\mathcal{D}_m(S_n))},$$

is satisfied for $a = 3$, but not for $a = 1, 2$. This implies there is a slight probability (0.0428) that $\mu(G)$ belongs to the values from different configurations of Chung-Lu model.

For function f_2, which is one when the owners of same breed dogs form connection, zero otherwise, $\hat{\mu}_C(\mathcal{D}_m(S_n)) = 1.995 \times 10^5$, $\widehat{\text{Var}}_C(\mathcal{D}_m(S_n)) = 2.9919 \times 10^4$ and for the original graph $\mu(\mathcal{D}_m(S_n)) = 1.831 \times 10^6$. We find that

$$|\hat{\mu}_C(\mathcal{D}_m(S_n)) - \mu(\mathcal{D}_m(S_n))| \nleqslant a\sqrt{\widehat{\text{Var}}_C(\mathcal{D}_m(S_n))} \text{ for } a = 1, 2, 3.$$

Hence the probability that $\mu(G)$ belongs to the values generated by the random network made from Chung-Lu model is less than 0.0027, which is negligible. These two inferences can also be observed in Figures 4.5 and 4.6.

4.7 Ratio with Tours Estimator (R-T estimator)

In this section we extend the idea of regeneration and tours introduced in this chapter to estimate the average function

$$\bar{\nu}(G) = \frac{1}{|V|}\sum_{u \in V} g(u).$$

We consider node functions for explaining the estimator and it can be easily extended to edge functions. Now since the idea of tours can only help us to create an unbiased estimator for $\nu(G)$, in order to find $\bar{\nu}(G)$ we form another similar estimator for $|V|$ with the same samples. With a sampling budget B, the estimator is given by

$$\widehat{\bar{\nu}}(\mathcal{D}_{m(B)}(S_n)) = \frac{\sum_{k=1}^{m(B)} \sum_{t=1}^{\xi_k - 1} \frac{f(X_t^{(k)})}{d_{X_t^{(k)}}} + \frac{\sum_{i \in I_n} g(i)}{d_{S_n}}}{\sum_{k=1}^{m(B)} \sum_{t=1}^{\xi_k - 1} \frac{1}{d_{X_t^{(k)}}} + \frac{n}{d_{S_n}}}, \tag{4.10}$$

where $m(B)$ is the number of tours until the budget B,

$$m(B) := \max\{k : \sum_{j=1}^{k} \xi_j \leqslant B\}.$$

The function $f(u) := g(u)$ if $u \notin I_n$, otherwise $f(u) = 0$.

This estimator is very close to respondent driven sampling (RDS), explained in Section 2.2.3, except that we miss $B - \sum_{k=1}^{m(B)} \xi_k$ samples for the estimation purpose. An advantage of R-T estimator is that we could leverage all the advantages of super-node and we claim that this would highly improve the performance. We show this via numerical simulations in the next section, and theoretical properties will be studied in future.

4.7.1 Numerical results

The following numerical studies compare RDS and R-T estimator. The choice of RDS for comparison is motivated by the results shown in the later Chapter 5 that it outperforms other samplings considered in this thesis in terms of asymptotic variance and mean squared error. For the figures given below, the x-axis represents the budget B which is the number of allowed samples. We use the normalized root mean squared error (NRMSE) for comparison for a given B and is defined as

$$\text{NRMSE} := \sqrt{\text{MSE}}/\bar{\nu}(G), \quad \text{where MSE} = \mathbb{E}\left[\left(\widehat{\bar{\nu}}^{(n)}(G) - \bar{\nu}(G)\right)^2\right].$$

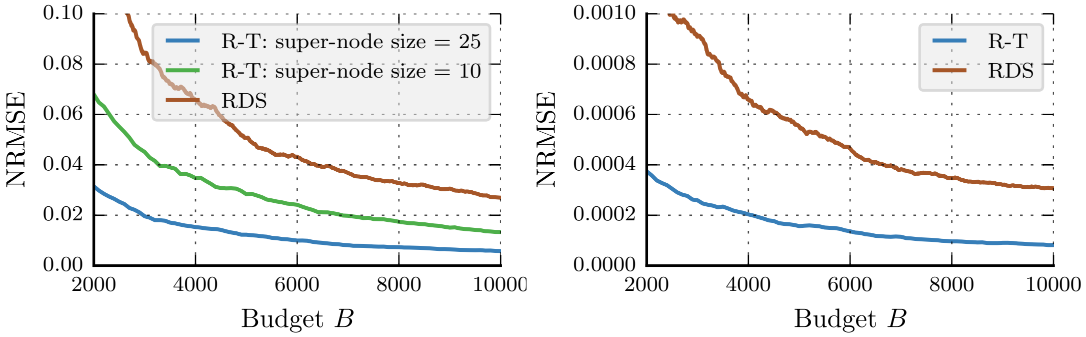
The super-node is formed from uniform sampling.

Les Misérables network: $|V| = 77, |E| = 254$

Figure 4.9 presents the comparison of R-T estimator with RDS estimator. Even small super-node size works for certain functions, as shown in Figure 4.9b for average clustering coefficient with super-node size 15.

Friendster network: $|V| = 64,600, |E| = 1,246,479$

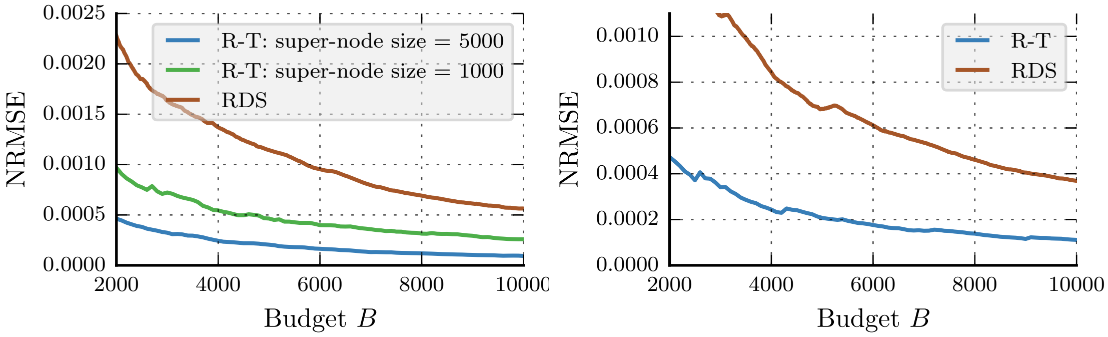
Figure 4.10 shows the results. In comparison to Les Misérables network, one can see that the performance is improved even for small super-node size.



(a) $g(v) = d_v$

(b) $g(v) = c(v)$: super-node size = 15

Figure 4.9: Les Misérables network: Comparison between RDS and R-T estimators
.



(a) $g(v) = \mathbb{I}\{d_v > 50\}$

(b) $g(v) = c(v)$: super-node size = 1000

Figure 4.10: Friendster network: Comparison between RDS and R-T estimators
.

4.8 Conclusions

In this work we have introduced a method that by crawling a fraction of a large network can produce, to the best of our knowledge, the first non-asymptotic unbiased estimates of network node and edge characteristics. Our method is based on random walk tours and a dynamic super-node algorithm. We derive lower and upper bounds of variance of this estimator and show its connection to the spectral gap of a random walk on the graph. One of our contributions is introducing an approximate Bayesian posterior of the network metric of interest using crawled data (random walk tours). We also derived a technique to study how a network looks “random” to a metric by estimating the same metric if the network was drawn from a Chung-Lu network or a configuration model with the same node labels and node degrees, all using random walk crawls without ever knowing the full original network. Our simulations over real-world networks show great accuracy of our estimators and approximations. In particular, the simulations clearly show that the derived posterior distribution fits very well with the data even when as few as 2.7%

of the edges and less than 15% of the nodes in the network are observed by the crawl. We have also found via numerical simulations that a natural extension of the estimator for sum function to average function performs better than the respondent driven sampling.

4.9 Proof of Theorem 4.2

First, in Lemma 4.7 we show that the estimate of $\mu(G')$ from each tour is unbiased.

Lemma 4.7. *Let $X_1^{(k)}, \dots, X_{\xi_k}^{(k)}$ be the nodes traversed by the k-th random walk tour on G', $k \geqslant 1$ starting at super-node S_n. Then the following holds, $\forall k$,*

$$\mathbb{E}\Big[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\Big] = \frac{2}{d_{S_n}} \mu(G'). \tag{4.11}$$

Proof. The random walk starts from the super-node S_n, thus

$$\mathbb{E}\Big[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\Big] = \sum_{(u,v)\in E'} \mathbb{E}\Big[\Big(\text{No. of times Markov chain crosses } (u,v) \text{ in the tour}\Big) f(u,v)\Big]. \tag{4.12}$$

Consider a renewal reward process with inter-renewal time distributed as $\xi_k, k \geqslant 1$ and reward as the number of times Markov chain crosses (u,v). From renewal reward theorem,

$$\begin{aligned}&\{\text{Asymptotic frequency of transitions from } u \text{ to } v\}\\ &\quad = \mathbb{E}[\xi_k]^{-1} \mathbb{E}\Big[\Big(\text{No. of times Markov chaincrosses } (u,v) \text{ in the tour}\Big) f(u,v)\Big].\end{aligned}$$

The left-hand side is essentially $2\pi_u p_{uv}$. Now (4.12) becomes

$$\begin{aligned}\mathbb{E}\Big[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\Big] &= \sum_{(u,v)\in E'} f(u,v)\, 2\pi_u\, p_{uv}\, \mathbb{E}[\xi_k] \\ &= 2 \sum_{(u,v)\in E'} f(u,v) \frac{d_u}{\sum_j d_j} \frac{1}{d_u} \frac{\sum_j d_j}{d_{S_n}} \\ &= \frac{2}{d_{S_n}} \sum_{(u,v)\in E'} f(u,v),\end{aligned}$$

which concludes our proof. □

In what follows we prove Theorem 4.2 using Lemma 4.7.

Proof of Theorem 4.2. By Lemma 4.7 the estimator $W_k = \sum_{t=2}^{\xi_k - 1} f(X_{t-1}^{(k)}, X_t^{(k)})$ is an unbiased estimate of $(2/d_{S_n})\mu(G')$. By the linearity of expectation the average estimator $\bar{W}(m) = m^{-1}\sum_{k=1}^{m} W_k$ is also unbiased.

We now consider two cases depending on f is defined as (4.2) or (4.3). When f is as in (4.2), it is trivial. For the function described in (4.3), $\mathbb{E}[W_k]$ can be rewritten as,

$$\frac{2}{d_{S_n}}\mathbb{E}[W_k] = \sum_{\substack{(u,v)\in E' \\ u\neq S_n, v\neq S_n}} g(u,v) + \sum_{\substack{(u,v)\in E' \\ u\neq S_n, v=S_n}} \frac{1}{k_{u\mathcal{S}}} \sum_{w\in I_n} g(u,w).$$

Note that in the graph G' there are $k_{u\mathcal{S}}$ multiple edges between u and S_n, when u and S_n are connected, and each contributes $\sum_{w\in I_n} g(u,w)$ to the net expectation. Moreover the multiplicative factor of two in the left-hand side of the above expression takes into account edges in both the directions since the random walk is reversible and graph is undirected. Hence

$$\frac{2}{d_{S_n}}\mathbb{E}[W_k] = \sum_{\substack{(u,v)\in E \\ u\notin I_n, v\notin I_n}} g(u,v) + \sum_{\substack{(u,v)\in E \\ u\notin I_n, v\in I_n}} g(u,v).$$

Finally for the estimator

$$\hat{\mu}(\mathcal{D}_m(S_n)) = \frac{d_{S_n}}{2m}\bar{W}(m) + \sum_{\substack{(u,v)\in E \\ \text{s.t. } u,v\in I_n}} f(u,v).$$

has average

$$\mathbb{E}[\hat{\mu}(\mathcal{D}_m(S_n))] = \sum_{\substack{(u,v)\in E \\ \text{s.t. } u\notin I_n \text{ or } v\notin I_n}} g(u,v) + \sum_{\substack{(u,v)\in E \\ \text{s.t. } u,v\in I_n}} g(u,v) = \mu(G).$$

Furthermore, by strong law of large numbers with $\mathbb{E}[W_k] < \infty$, $\hat{\mu}(\mathcal{D}_m(S_n)) \to \mu(G)$ a.s. as $m \to \infty$. This completes our proof. □

4.10 Proof of Lemma 4.3

Proof. For RWuR, stationary distribution of node u, $\hat{\pi}_u = \frac{d_u+\alpha}{2|E'|+|V'|\alpha}$ and transition probability from node u to v, $\hat{p}_{uv} = \frac{\alpha/|V'|+1}{d_u+\alpha}$ if u and v are connected, $\frac{\alpha/|V'|}{d_u+\alpha}$ otherwise [Avrachenkov et al. 2010].

Let $f''(u,v) = g(u,v)\frac{d_u d_w}{2|E'|-1}$ and f' as defined in (4.7). Let $V'' = V - I_n$. We have

$$\begin{aligned} V \times V &= \{V'' \cup I_n\} \times \{V'' \cup I_n\} \\ &= \{V'' \times V''\} \cup \{V'' \times I_n\} \cup \{I_n \times V''\} \cup \{I_n \times I_n\}. \end{aligned}$$

Now the value in the set $\{V'' \times I_n\}$ can be expressed in terms of $V' \times V'$ as,

$$\begin{aligned}\sum_{(u,v)\in V''\times I_n} f''(u,v) &= \sum_{u\in V''}\sum_{w\in I_n} f''(u,w)\\ &= \sum_{\substack{u\neq S_n, v=S_n\\ (u,v)\in V'\times V'}}\sum_{w\in I_n} f''(u,w).\end{aligned}$$

$$\begin{aligned}\mathbb{E}[W'_k] &= \sum_{(u,v)\in E'} f'(u,v)2\hat{\pi}_u\hat{p}_{uv}\,\mathbb{E}[\xi_k] + \sum_{(u,v)\in (E')^c} f'(u,v)2\hat{\pi}_u\hat{p}_{uv}\,\mathbb{E}[\xi_k]\\
&= \frac{2\alpha/|V'|}{d_{S_n}+\alpha}\sum_{(u,v)\in E'\cup(E')^c} f'(u,v) + \frac{2}{d_{S_n}+\alpha}\sum_{(u,v)\in E'} f'(u,v)\\
&= 2\sum_{\substack{(u,v)\in E'\\ u\neq S_n, v\neq S_n}} f''(u,v)\frac{\alpha/|V'|+1}{d_{S_n}+\alpha} + 2\sum_{\substack{(u,v)\in E'\\ u\neq S_n, v=S_n}} \frac{\alpha/|V'|+k_{u\mathcal{S}}}{d_{S_n}+\alpha}\sum_{w\in I_n} f''(u,w)\\
&\quad + 2\sum_{\substack{(u,v)\in (E')^c\\ u\neq S_n, v\neq S_n}} \frac{\alpha/|V'|}{d_{S_n}+\alpha} f''(u,v) + 2\sum_{\substack{(u,v)\in (E')^c\\ u\neq S_n, v=S_n}} \frac{\alpha/|V'|}{d_{S_n}+\alpha}\sum_{w\in I_n} f''(u,w)\\
&= \frac{\alpha/|V'|}{d_{S_n}+\alpha}\left[\sum_{(u,v)\in V''\times V''} f''(u,v) + \sum_{(u,v)\in V''\times I_n} f''(u,v)\right]\\
&\quad + \frac{1}{d_{S_n}+\alpha}\left[\sum_{\substack{(u,v)\in E'\\ u\neq S_n, v=S_n}} k_{u\mathcal{S}}\left(\frac{1}{k_{u\mathcal{S}}}\sum_{w\in I_n} f''(u,w)\right)\right].\end{aligned}$$

A multiplicative factor of 2 will be added to the first term in the above expression since RWuR is reversible and the graph under consideration is undirected. The last term can be removed by using classical random walk tours $\{W''_k\}$ with appropriate bias. The unbiasedness of the estimator then follows from the linearity of expectation. □

4.11 Proof of Lemma 4.4

(i). The variance of the estimator at tour $k \geqslant 1$ starting from node S_n is

$$\mathrm{Var}_{S_n}\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}))\right] \leqslant B^2\mathbb{E}[(\xi_k-1)^2] - \left(\mathbb{E}\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}))\right]\right)^2. \quad (4.13)$$

It is known from [Aldous & Fill 2002, Chapter 2 and 3] that

$$\mathbb{E}[\xi_k^2] \quad = \quad \frac{2\sum_{i\geqslant 2} w_{S_n i}^2(1-\lambda_i)^{-1}+1}{\pi_{S_n}^2}.$$

Using Theorem 4.7 eq. (4.13) can be written as

$$\begin{aligned}&\operatorname{Var}\left[\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)}))\right] \\ &\quad\leqslant \frac{1}{d_{S_n}^2}\left(2d_{\text{tot}}^2 B^2 (\sum_{i\geqslant 2} w_{S_n m}^2 (1-\lambda_i)^{-1}) - 4\mu^2(G')\right) - \frac{1}{d_{S_n}} B^2 d_{\text{tot}} + B^2.\end{aligned}$$

The latter can be upper-bounded by $B^2(2d_{\text{tot}}^2/(d_i^2\delta)+1)$.

For the second part, we have

$$\begin{aligned}\mathbb{E}\left[\left(\sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})\right)^l\right] &\leqslant B^l \mathbb{E}[(\xi_k - 1)^l)] \\ &\leqslant C(\mathbb{E}[(\xi_k)^l] + 1),\end{aligned}$$

for a constant $C > 0$ using C_r inequality [Gut 2012, Chapter 3, Theorem 2.2]. From [Meyn & Tweedie 2012], it is known that there exists an $a > 0$, such that $\mathbb{E}[\exp(a\,\xi_k)] < \infty$, and this implies that $\mathbb{E}[(\xi_k)^l] < \infty$ for all $l \geqslant 0$. This proves the theorem.

(ii). We denote $\mathbb{E}_\pi f$ for $\mathbb{E}_\pi[f(Y_1, Y_2)]$ and Normal(a, b) indicates Gaussian distribution with mean a and variance b. With the trivial extension of the central limit theorem of Markov chains [Meyn & Tweedie 2012] of node functions to edge functions, we have for the ergodic estimator $\bar{f}_n = n^{-1}\sum_{t=2}^n f(Y_{t-1}, Y_t)$,

$$\sqrt{n}(\bar{f}_n - \mathbb{E}_\pi f) \xrightarrow{d} \text{Normal}(0, \sigma_a^2), \tag{4.14}$$

where

$$\sigma_a^2 = \operatorname{Var}(f(Y_1, Y_2)) + 2\sum_{l=2}^{n-1} \frac{(n-1)-l}{n} \operatorname{Cov}(f(Y_0, Y_1), f(Y_{l-1}, Y_l)) < \infty.$$

We derive σ_a^2 in Lemma 4.8. Note that σ_a^2 is also the asymptotic variance of the ergodic estimator of edge functions.

Consider a renewal reward process at its k-th renewal, $k \geqslant 1$, with inter-renewal time ξ_k and reward as $W_k = \sum_{t=2}^{\xi_k} f(X_{t-1}^{(k)}, X_t^{(k)})$. Let $\bar{W}(n)$ be the average cumulative reward gained up to m-th renewal, i.e., $\bar{W}(m) = m^{-1}\sum_{k=1}^m W_k$. From the central limit theorem for the renewal reward process [Tijms 2003, Theorem 2.2.5], with $l_n = \operatorname{argmax}_k \sum_{j=1}^k \mathbf{1}(\xi_j \leqslant n)$, after n total number of steps yields

$$\sqrt{n}(\bar{W}(l_n) - \mathbb{E}_\pi f) \xrightarrow{d} \text{Normal}(0, \sigma_b^2), \tag{4.15}$$

with $\sigma_b^2 = \dfrac{\zeta^2}{\mathbb{E}[\xi_k]}$ and

$$\begin{aligned}\zeta^2 &= \mathbb{E}[(W_k - \xi_k \mathbb{E}_\pi f)^2] = \mathbb{E}_i\left[\left(W_k - \xi_k\frac{\mathbb{E}[W_k]}{\mathbb{E}[\xi_k]}\right)^2\right] \\ &= \operatorname{Var}_{S_n}(W_k) + (\mathbb{E}[W_k])^2 + \left(\frac{\mathbb{E}[W_k]}{\mathbb{E}[\xi_k]}\right)^2 \mathbb{E}[(\xi_k)^2] \\ &\quad -2\frac{\mathbb{E}[W_k]}{\mathbb{E}[\xi_k]}\mathbb{E}[W_k\xi_k].\end{aligned}$$

In fact it can be shown that (see [Meyn & Tweedie 2012, Proof of Theorem 17.2.2])

$$|\sqrt{n}(\bar{f}_n - \mathbb{E}_\pi f) - \sqrt{n}(\bar{W}(l_n) - \mathbb{E}_\pi f)| \to 0 \quad \text{a.s.}.$$

Therefore $\sigma_a^2 = \sigma_b^2$. Combining this result with Lemma 4.8 shown below we get (4.8). □

Lemma 4.8.

$$\begin{aligned}\lim_{n\to\infty} \frac{1}{n}\mathrm{Var}_\pi\left(\sum_{k=2}^{n} f(Y_{k-1}, Y_k)\right)& \\ &= 2\sum_{i=2}^{r} \frac{\lambda_i}{1-\lambda_i}\langle f, v_i\rangle_{\hat{\pi}}\,(u_i^\intercal \hat{f}) + \frac{1}{d_{tot}}\sum_{(i,j)\in E} f(i,j)^2 \\ &\quad + \frac{1}{d_{tot}^2}(\sum_{(i.j)\in E} f(i,j)^2)^2 + \frac{1}{d_{tot}^2}\sum_{i\in V} d_i\Big(\sum_{i\sim j} f(i,j)\Big)^2\end{aligned}$$

Proof. We extend the arguments in the proof of [Brémaud 1999, Theorem 6.5] to the edge functions. When the initial distribution is $\boldsymbol{\pi}$, we have

$$\begin{aligned}\lim_{n\to\infty} \frac{1}{n}\mathrm{Var}_\pi\left(\sum_{k=2}^{n} f(Y_{k-1}, Y_k)\right)& \\ &= \frac{1}{n}\Big(\sum_{k=2}^{n}\mathrm{Var}_\pi(f(Y_{k-1},Y_k)) + 2\sum_{\substack{k,j=2\\k<j}}\mathrm{Cov}_\pi(f(Y_{k-1},Y_k), f(Y_{j-1},Y_j))\Big) \\ &= \mathrm{Var}_\pi(f(Y_{k-1},Y_k)) + 2\sum_{l=2}^{n-1}\frac{(n-1)-l}{n}\mathrm{Cov}_\pi(f(Y_0,Y_1), f(Y_{l-1},Y_l)).\end{aligned} \tag{4.16}$$

Now the first term in (4.16) is

$$\mathrm{Var}_\pi(f(Y_{k-1},Y_k)) = \langle f, f\rangle_{\hat{\pi}} - \langle f, \boldsymbol{\Pi}\hat{f}\rangle_{\hat{\pi}}, \tag{4.17}$$

where $\boldsymbol{\Pi} = \mathbf{1}\boldsymbol{\pi}^\intercal$.

For the second term in (4.16),

$$\mathrm{Cov}_\pi(f(Y_0,Y_1), f(Y_{l-1},Y_l)) = \mathbb{E}_\pi(f(Y_0,Y_1), f(Y_{l-1},Y_l)) - (\mathbb{E}_\pi[f(Y_0,Y_1)])^2.$$

$$\begin{aligned}\mathbb{E}_\pi(f(Y_0,Y_1), f(Y_{l-1},Y_l)) &= \sum_i\sum_j\sum_k\sum_m \boldsymbol{\pi}_i\, p_{ij}\, p_{jk}^{(l-2)}\, p_{km} f(i,j) f(k,m) \\ &= \sum_i\sum_j\sum_k \boldsymbol{\pi}_i\, p_{ij}\, f(i,j)\, p_{jk}^{(l-2)}\,\hat{f}(k) \\ &= \sum_{(i,j)\in E} \hat{\boldsymbol{\pi}}_{ij}\; f(i,j)\,(\boldsymbol{P}^{(l-2)}\hat{f})(j) \\ &= \langle f, \boldsymbol{P}^{(l-2)}\hat{f}\rangle_{\hat{\pi}}.\end{aligned} \tag{4.18}$$

Therefore,

$$\mathrm{Cov}_\pi(f(Y_0,Y_1),f(Y_{l-1},Y_l)) = \langle f, (\boldsymbol{P}^{(l-2)} - \boldsymbol{\Pi})\hat{f}\rangle_{\hat{\pi}}.$$

Taking limits, we get

$$\begin{aligned}
&\lim_{n\to\infty}\sum_{l=2}^{n-1}\frac{n-l-1}{n}(\boldsymbol{P}^{(l-2)} - \boldsymbol{\Pi}) \\
&\qquad = \lim_{n\to\infty}\sum_{k=1}^{n-3}\frac{n-k-3}{n}(\boldsymbol{P}^{k} - \boldsymbol{\Pi}) + (\boldsymbol{I} - \boldsymbol{\Pi}) \\
&\qquad \overset{(a)}{=} \lim_{n\to\infty}\sum_{k=1}^{n-1}\frac{n-k}{n}(\boldsymbol{P}^{k} - \boldsymbol{\Pi}) + (\boldsymbol{I} - \boldsymbol{\Pi}) - \lim_{n\to\infty}\frac{3}{n}\sum_{k=1}^{n-3}(\boldsymbol{P}^{k} - \boldsymbol{\Pi}) \\
&\qquad = (\boldsymbol{Z} - \boldsymbol{I}) + (\boldsymbol{I} - \boldsymbol{\Pi}) = \boldsymbol{Z} - \boldsymbol{\Pi}, \qquad (4.19)
\end{aligned}$$

where the first term in (a) follows from the proof of [Brémaud 1999, Theorem 6.5] and since $\lim_{n\to\infty}(\boldsymbol{P}^n - \boldsymbol{\Pi}) = 0$, the last term is zero using Cesaro's lemma [Brémaud 1999, Theorem 1.5 of Appendix].

We have,

$$\boldsymbol{Z} = \boldsymbol{I} + \sum_{i=2}^{r}\frac{\lambda_i}{1-\lambda_i}v_i u_i^{\intercal},$$

Thus

$$\begin{aligned}
&\lim_{n\to\infty}\frac{1}{n}\mathrm{Var}_\pi\left(\sum_{k=2}^{n} f(Y_{k-1},Y_k)\right) \\
&\qquad = \langle f,f\rangle_{\hat{\pi}} - \langle f, \boldsymbol{\Pi}\hat{f}\rangle_{\hat{\pi}} + 2\langle f, \left(\boldsymbol{I} + \sum_{i=2}^{r}\frac{\lambda_i}{1-\lambda_i}v_i u_i^{\intercal} - \boldsymbol{\Pi}\right)\hat{f}\rangle_{\hat{\pi}} \\
&\qquad = \langle f,f\rangle_{\hat{\pi}} + \langle f, \boldsymbol{\Pi}\hat{f}\rangle_{\hat{\pi}} + 2\langle f,\hat{f}\rangle_{\hat{\pi}} + 2\sum_{i=2}^{r}\frac{\lambda_i}{1-\lambda_i}\langle f, v_i\rangle_{\hat{\pi}}\,(u_i^{\intercal}\hat{f}) \\
&\qquad = \frac{1}{d_{tot}}\sum_{(i,j)\in E} f(i,j)^2 + \frac{1}{d_{tot}^2}(\sum_{(i.j)\in E} f(i,j)^2)^2 \\
&\qquad\qquad + \frac{1}{d_{tot}^2}\sum_{i\in V} d_i\Big(\sum_{i\sim j} f(i,j)\Big)^2 + 2\sum_{i=2}^{r}\frac{\lambda_i}{1-\lambda_i}\langle f, v_i\rangle_{\hat{\pi}}\,(u_i^{\intercal}\hat{f})
\end{aligned}$$

□

Chapter 5

Reinforcement Learning based Estimation in Networks

In this chapter, we revisit the estimation of

$$\bar{\nu}(G) = \frac{1}{|V|} \sum_{u \in V} g(u).$$

We will introduce a reinforcement learning approach based on stochastic approximation. The underlying idea relies on the idea of tours and regeneration introduced in Chapter 4. We will compare the mean squared error of the new estimator with classical estimators and see how the stability of the sample paths can be controlled.

Organization

Section 5.1 introduces the reinforcement learning based sampling and estimation. It also details a modification for an easy implementation and compares with the algorithm in Chapter 4. Section 5.2 contains the numerical simulations on real-world networks and makes several observations about the new algorithms.

5.1 Network Sampling with Reinforcement Learning (RL-technique)

Consider an undirected connected graph G with node set V and edge set E. Let $V_0 \subset V$ with $|V_0| << |V|$. Consider the simple random walk (according to the definition in Section 2.2.1) $\{X_n\}$ on G with transition probabilities $p(j|i) = 1/d_i$ if $(i, j) \in E$ and zero otherwise. Define $Y_n := X_{\tau_n}$ for $\tau_n :=$ successive times to visit V_0. Then $\{(Y_n, \tau_n)\}$ is a semi-Markov process on V_0 [Ross 1992, Chapter 5]. In particular, $\{Y_n\}$ is a Markov chain on V_0 with transition probability matrix (say) $[p_Y(j|i)]$. Let $\xi_1 := \min\{n > 0 : X_n \in V_0\}$ similar to the definition in Chapter 4 and for a prescribed $f : V \mapsto \mathbb{R}$, define

$$\begin{aligned} T_i &:= \mathbb{E}_i[\xi], \\ h(i) &:= \mathbb{E}_i\left[\sum_{m=1}^{\xi} f(X_m)\right], \; i \in V_0. \end{aligned}$$

Then the Poisson equation for the semi-Markov process (Y_n, τ_n) is [Ross 1992, Chapter 7]

$$\mathcal{V}(i) = h(i) - \beta T_i + \sum_{j \in V_0} p_Y(j|i)\mathcal{V}(j), \; i \in V_0. \tag{5.1}$$

Here β is the stationary average of f, $\mathbb{E}_\pi[f(X_0)]$, and $\mathcal{V}$ indicates the value function. Let $\{Z_n\}$ be i.i.d. uniform on V_0. For each $n \geqslant 1$, generate an independent copy $\{X_m^n\}$ of $\{X_m\}$ with $X_0^n = Z_n$ for $0 \leqslant m \leqslant \xi_n :=$ the inter return time to V_0. A learning algorithm for (5.1) along the lines of [Abounadi et al. 2001] then is, for $i \in V_0$,

$$\begin{aligned}\mathcal{V}_{n+1}(i) &= \mathcal{V}_n(i) + a(n)\mathbb{I}\{Z_n = i\} \times \\ &\left[\left(\sum_{m=1}^{\xi(n)} f(X_m^n)\right) - \mathcal{V}_n(i_0)\xi(n) + \mathcal{V}_n(X_{\xi_n}^n) - \mathcal{V}_n(i)\right], \qquad (5.2)\end{aligned}$$

where $a(n) > 0$ are stepsizes satisfying $\sum_n a(n) = \infty$, $\sum_n a(n)^2 < \infty$. (One good choice is $a(n) = 1/\lceil \frac{n}{N} \rceil$ for $N = 50$ or 100.) Here $\mathbb{I}\{A\}$ denotes indicator function for the set A. Also, i_0 is a prescribed element of V_0. One can use other normalizations in place of $\mathcal{V}_n(i_0)$, such as $\frac{1}{|V_0|}\sum_j \mathcal{V}_n(j)$ or $\min_i \mathcal{V}_n(i)$, etc. Then this normalizing term ($\mathcal{V}_n(i_0)$ in (5.2)) converges to β as n increases to ∞. With the underlying random walk as the Metropolis-Hastings, the normalizing term forms our estimator $\hat{\mu}_{RL}^{(n)}(G)$ in RL based approach.

The relative value iteration algorithm to solve (5.1) is

$$\mathcal{V}_{n+1}(i) = h(i) - \mathcal{V}_n(i_0)T_i + \sum_j p_Y(j|i)V_n(j),$$

and (5.2) is the stochastic approximation analog of it which replaces conditional expectation w.r.t. transition probabilities with an actual sample. And then makes an incremental correction based on it, with a slowly decreasing stepwise function that ensures averaging. The latter is a standard aspect of stochastic approximation theory. The smaller the stepwise the less the fluctuations but slower the speed, thus there is a trade-off between the two.

RL methods can be thought of as a cross between a pure deterministic iteration such as the relative value iteration above and pure MCMC, trading off variance against per iterate computation. The gain is significant if the number of neighbors of a node is much smaller than the number of nodes, because we are essentially replacing averaging over the latter by averaging over neighbors. The $\mathcal{V}$-dependent terms can be thought of as control variates to reduce variance.

Mean squared error of RL-technique

For the RL-technique the following concentration bound is true [Borkar 2008]:

$$\mathbb{P}\left\{|\hat{\mu}_{RL}^{(N)}(G) - \bar{\nu}(G)| \geqslant \varepsilon\right\} \leqslant K\exp(-k\varepsilon^2 N).$$

Thus it follows that MSE is $\mathcal{O}(\frac{1}{\sqrt{N}})$ because

$$\begin{aligned}\mathbb{E}|\hat{\mu}_{RL}^{(N)}(G) - \bar{\nu}(G)|^2 &= \int_0^\infty \mathbb{P}\left\{|\hat{\mu}_{RL}^{(N)}(G) - \bar{\nu}(G)|^2 \geqslant \varepsilon\right\} d\varepsilon \\ &= \int_0^\infty \mathbb{P}\left\{|\hat{\mu}_{RL}^{(N)}(G) - \bar{\nu}(G)| \geqslant \varepsilon^{1/2}\right\} d\varepsilon \\ &\leqslant \int_0^\infty K \exp(-k\varepsilon N)\, d\varepsilon = \mathcal{O}\left(\frac{1}{N}\right).\end{aligned}$$

5.1.1 Extension of RL-technique to uniform stationary average case

The stochastic approximation iteration in (5.2) converges to β, which is $\mathbb{E}_\pi[f(X_0)]$. To make it converge to $\bar{\nu}(G)$, we use the Metroplois-Hastings random walk. But without using Metropolis-Hastings algorithm, the following modification, motivated from importance sampling, achieves the convergence to $\bar{\nu}(G)$ with a standard random walk.

$$\begin{aligned}\mathcal{V}_{n+1}(i) = \mathcal{V}_n(i) \ &+ a(n)\mathbb{I}\{z = i\} \times \Gamma_{\xi_n}^{(n)} \times \\ &\left[\left(\sum_{m=1}^{\xi(n)} f(X_m^{(n)})\right) - \mathcal{V}_n(i_0)\xi(n) + \mathcal{V}_n(X_{\xi_n}^{(n)}) - \mathcal{V}_n(i)\right],\end{aligned}$$

where

$$\Gamma_m^{(n)} = \prod_{k=1}^{m}\left(\frac{p(X_k^{(n)}|X_{k-1}^{(n)})}{q(X_k^{(n)}|X_{k-1}^{(n)})}\right).$$

Here $q(\cdot|\cdot)$ is the transition probability of the random walk with which we simulate the algorithm and $p(\cdot|\cdot)$ corresponds to the transition probability of the random walk with respect to which we need the stationary average. The transition probability p can belong to any random walk having uniform stationary distribution such that $q(\cdot|\cdot) > 0$ whenever $p(\cdot|\cdot) > 0$. One example is to use p as the transition probability of Metroplis-Hastings algorithm with target stationary distribution as uniform and q as the transition probability of a lazy version of simple random walk, i.e., with transition probability matrix $(\boldsymbol{I} + \boldsymbol{P}_{\text{SRW}})/2$. Such a modification avoids the use of Metropolis-Hastings dynamics which require extra API requests for probing the degree of the neighboring node.

5.1.2 Advantages

The RL-technique extends the idea of regeneration, tours and super-node introduced in Chapter 4 to the average function $\bar{\nu}(G)$ from $\nu(G)$. Though the RL-technique does not have the non-asymptotic property as in the algorithm in Chapter 4, it has the following advantages:

1. It does not need to wait until burn-in time to collect samples.

2. Note the similarity in the algorithms: the super-node in Chapter 4 is a single node, an amalgamation of the node set V_0. But such a direction assumes that the contribution of all the edges inside the induced subgraph of V_0 to $\bar{\nu}(G)$ be completely known. It could have been avoided if we could make use of the techniques for partitioning state space of a Markov chain (called *lumpability* in [Kemeny & Snell 1983]). The conditions stated in [Kemeny & Snell 1983, Theorem 6.3.2] are not satisfied here and hence can not invoke the such techniques. On the other hand, the RL-technique, without using the *lumpability* arguments, need not know the edge functions of the subgraph induced by V_0.

3. RL-technique along with the extension in Section 5.1.1 can be extended to the directed graph case provided the graph is strongly connected. But for the estimator from Chapter 4, recollect that it requires knowledge of the stationary distribution of the underlying random walk to unbias and form the estimator, which does not have an easy expression as far as we know and this applies to a simple random walk on directed graphs as well.

4. As explained before, the main advantage of RL-estimator is its ability to control the stability of sample paths and its position as a cross between deterministic and MCMC iteration. We will see more about this in the numerical section.

5.2 Numerical Comparison

The algorithms RL-technique, resondent driven sampling (RDS) and Metropolis-Hastings sampling (MH-MCMC) (see Sections 2.2.2 and 2.2.3) are compared in this section using simulations on two real-world networks. For the figures given below, the x-axis represents the budget B which is the number of allowed samples, and is the same for all the techniques. We use the normalized root mean squared error (NRMSE) for comparison for a given B and is defined as

$$\text{NRMSE} := \sqrt{\text{MSE}}/\bar{\nu}(G), \quad \text{where MSE} = \mathbb{E}\left[\left(\widehat{\bar{\nu}}^{(n)}(G) - \bar{\nu}(G)\right)^2\right].$$

Recall that $\text{MSE} = \text{Var}[\widehat{\bar{\nu}}^{(n)}(G)] + \left(\mathbb{E}[\widehat{\bar{\nu}}^{(n)}(G)] - \bar{\nu}(G)\right)^2$. We also study the asymptotic variance σ_g^2 (see (2.1)) of the random walk based estimators including RL-technique in terms of $n \times \text{MSE}$, since the bias $|\mathbb{E}[\widehat{\bar{\nu}}^{(n)}(G)] - \bar{\nu}(G)| \to 0$ as $n \to \infty$.

For the RL-technique we choose the initial or super-node V_0 by uniformly sampling nodes assuming the size of V_0 is given a priori.

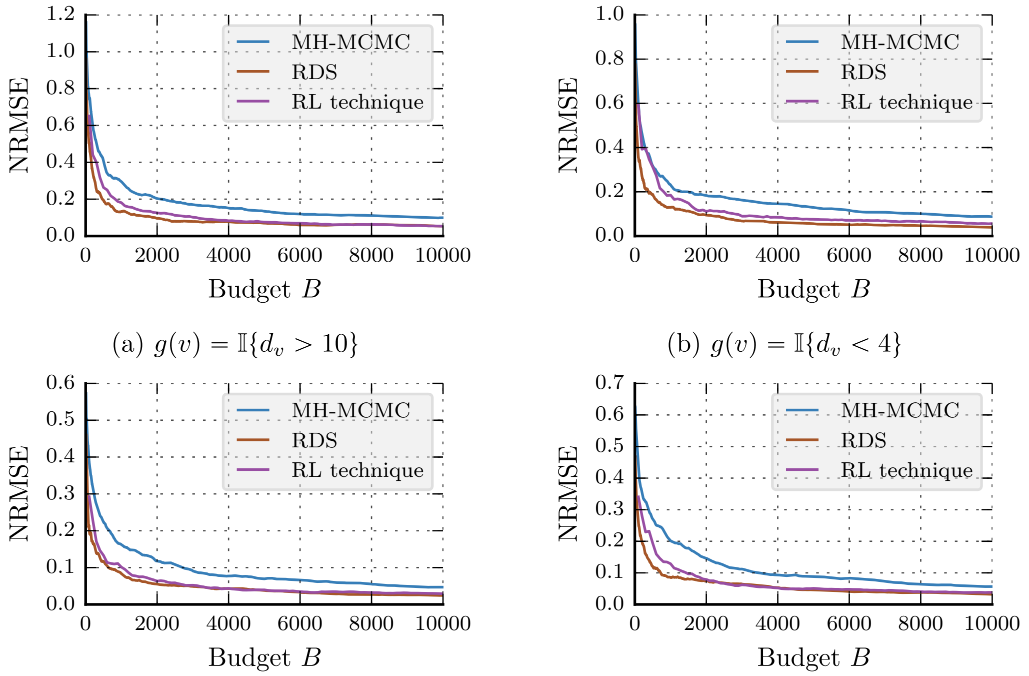
5.2.1 Les Misérables network

In Les Misérables network, nodes are the characters of the novel and edges are formed if two characters appear in the same chapter in the novel. The number of nodes is 77 and number of edges is 254. We have chosen this rather small network in order to compare all the three methods in terms of theoretical limiting variance. Here we consider four demonstrative functions: a) $g(v) = \mathbb{I}\{d_v > 10\}$ b) $g(v) = \mathbb{I}\{d_v < 4\}$ c) $g(v) = d_v$, where $\mathbb{I}\{A\}$ is the indicator function for set A and d) for calculating $\bar{\nu}(G)$ as the average clustering coefficient

$$C := \frac{1}{|V|}\sum_{v\in V} c(v), \quad \text{where } c(v) = \begin{cases} t(v)/\binom{d_v}{2} & \text{if } d_v \geqslant 2 \\ 0 & \text{otherwise,} \end{cases} \tag{5.3}$$

with $t(v)$ as the number of triangles that contain node v. Then $g(v)$ is taken as $c(v)$ itself.

The average in MSE is calculated from multiple runs of the simulations. The simulations on Les Misérables network is shown in Figure 5.1 with $a(n) = 1/\lceil \frac{n}{10} \rceil$ and the super-node size as 25.



(a) $g(v) = \mathbb{I}\{d_v > 10\}$

(b) $g(v) = \mathbb{I}\{d_v < 4\}$

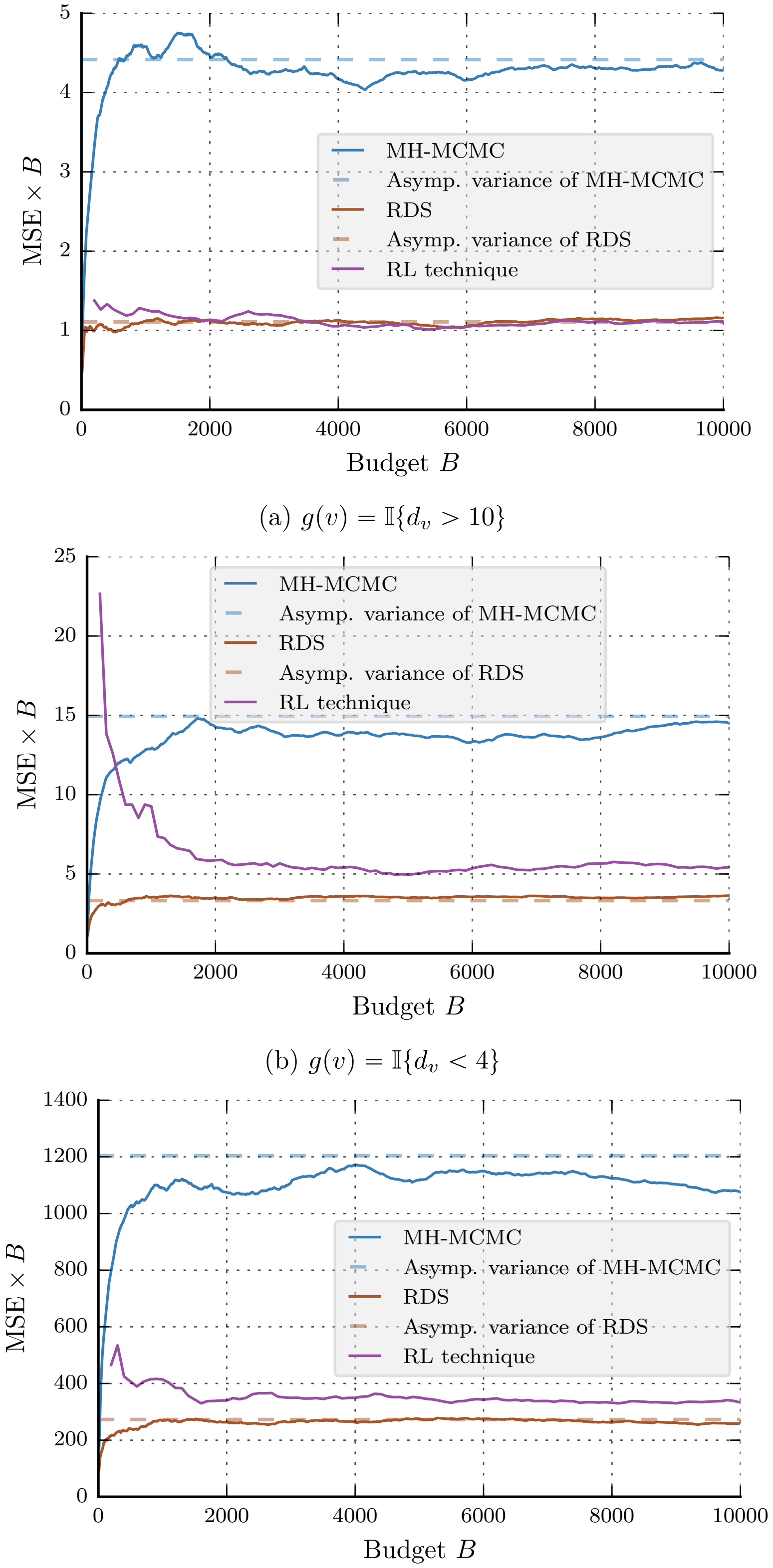
(c) $g(v) = d_v$

(d) $g(v) = c(v)$, $c(v)$ defined in (5.3)

Figure 5.1: Les Misérables network: NRMSE comparisons

5.2.1.1 Study of asymptotic MSE:

In order to show the asymptotic MSE expressions derived in Propositions 2.6 and 2.8, we plot the sample MSE as MSE × B in Figures 5.2a, 5.3b and 5.2c. These figures correspond to the three different functions we have considered. It can be seen that asymptotic MSE expressions match well with the estimated ones.



(a) $g(v) = \mathbb{I}\{d_v > 10\}$

(b) $g(v) = \mathbb{I}\{d_v < 4\}$

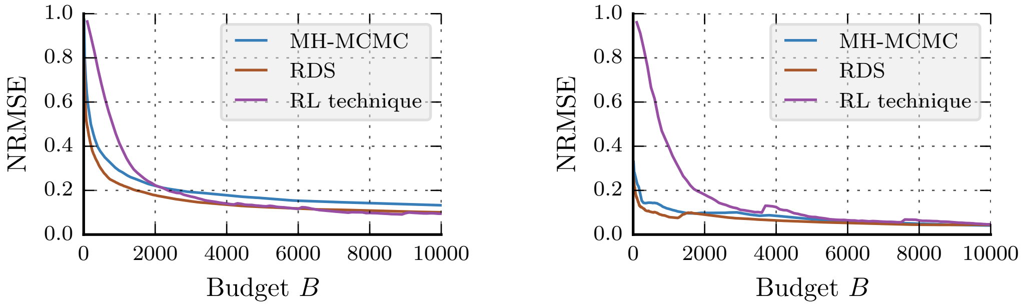
(c) $g(v) = d_v$

Figure 5.2: Les Misérables network: asymptotic MSE comparisons (contd.)

5.2.2 Friendster network

We consider a larger graph here, a connected subgraph of an online social network called Friendster with 64,600 nodes and 1,246,479 edges. The nodes in Friendster are individuals and edges indicate friendship. We consider the functions a). $g(v) = \mathbb{I}\{d_v > 50\}$ and b). $g(v) = c(v)$ (see (5.3)) used to estimate the average clustering coefficient. The plot in Figure 5.3a shows the results for Friendster graph with super-node size 1000. Here the sequence $a(n)$ is taken as $1/\lceil \frac{n}{25} \rceil$.

Now we concentrate on *single* sample path properties of the algorithms. Hence the numerator of NRMSE becomes absolute error. Figure 5.3c shows the effect of increasing super-node size while fixing step size $a(n)$ and Figure 5.3d shows the effect of changing $a(n)$ when super-node is fixed. In both the cases, the green curve of RL-technique shows much stability compared to the other techniques.



(a) $g(v) = \mathbb{I}\{d_v > 50\}$ (b) $g(v) = c(v)$, $c(v)$ defined in(5.3)

Figure 5.3: Friendster network: (a) & (b) NRMSE comparison

.

5.2.3 Observations

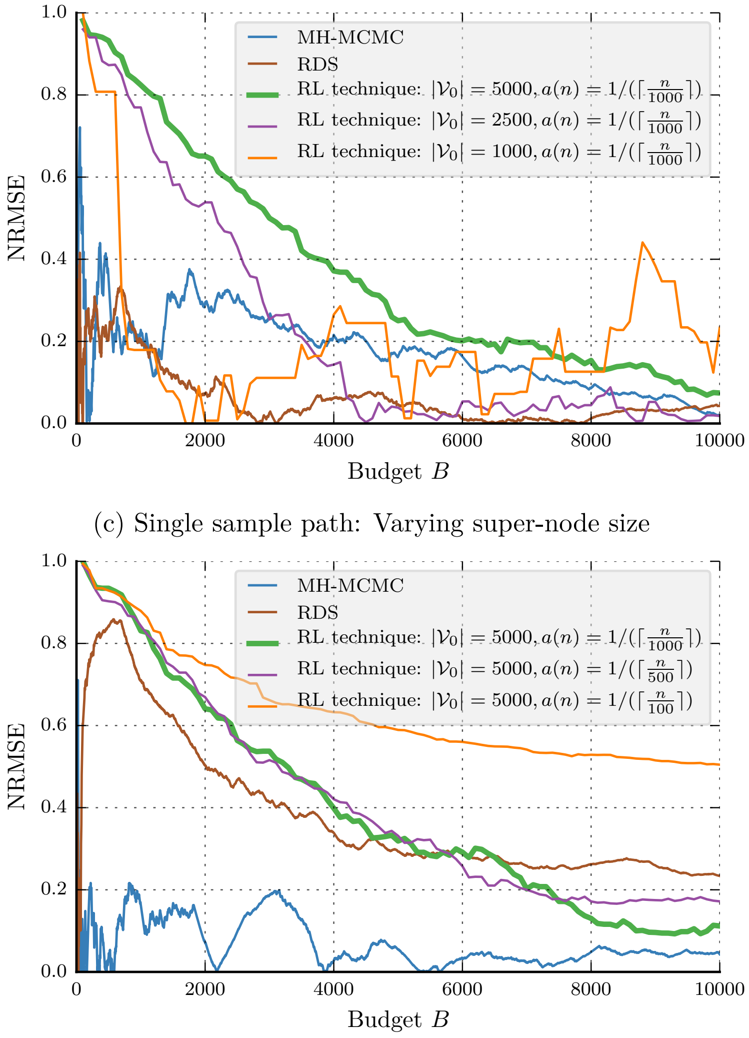
Some observations from the numerical experiments are as follows:

1. With respect to the limiting variance, RDS always outperforms the other two methods tested. However, with a good choice of parameters the performance of RL is not far from that of RDS.

2. In the RL-technique, we find that the normalizing term $1/|V_0| \sum_j \mathcal{V}_n(j)$ converges much faster than the other two options, $\mathcal{V}_t(i_0)$ and $\min_i \mathcal{V}_t(i)$;

3. When the size of the super-node decreases, the RL-technique requires smaller step size $a(n)$. For instance in case of Les Misérables network, if the super-node size is less than 10, RL-technique does not converge with $a(n) = 1/(\lceil \frac{n}{50} \rceil + 1)$ and requires $a(n) = 1/(\lceil \frac{n}{5} \rceil)$;

4. If step size $a(n)$ decreases or the super node size increases, RL fluctuates less but with slower convergence. In general, RL has less fluctuations than MH-MCMC or RDS.



(c) Single sample path: Varying super-node size

(d) Single sample path: Varying step size

Figure 5.3: Friendster network (contd.): (c) & (d) Single sample path comparison with $g(v) = \mathbb{I}\{d_v > 50\}$

.

5.3 Conclusions

In this chapter we studied and compared the performances of a sampling technique based on reinforcement learning for average function estimation with RDS and MH sampling. We found that in terms of asymptotic mean squared error (MSE), the RDS technique outperforms the other methods considered. However, RL-technique with small step size displays a more stable sample path in terms of MSE and its performance is comparable with respect to RDS. In the extended version of the work we plan to test the methods on larger graphs and involve more methods for comparison.

Chapter 6

Extremes in Sampling Processes

Data from real complex networks shows that correlations exist in various forms, for instance the existence of social relationships and interests in social networks. Degree correlations between neighbors, correlations in income, followers of users and number of likes of specific pages in social networks are some examples, to name a few. These kind of correlations have several implications in network structure. For example, degree-degree correlation manifests itself in assortativity or disassortativity of the network [Barrat et al. 2008].

Additionally, we consider large complex networks where it is impractical to have a complete picture a priori. Crawling or sampling techniques can be employed in practice to explore such networks by making use of API calls or HTML scrapping. We look into the stationary samples genereated by any randomized sampling techniques.

Taking into account the correlation in the graph and the correlation introduced by the stationary samples, this chapter is devoted to the study of distribution and dependence of extremal events in network sampling processes.

Organization

Section 6.1 introduces the extremal index, a key parameter in extreme value theory which we will explore in detail in this chapter. In Section 6.2, methods to derive EI are presented. Section 6.3 considers the case of degree correlations. In Section 6.3.1 the graph model and correlated graph generation technique are presented. Section 6.3.2 explains the different types of random walks studied and derives associated transition kernels and joint degree distributions. EI is analytically calculated for different sampling techniques later in Section 6.3.3. In Section 6.4 we provide several applications of EI in graph sampling techniques. In Section 6.5 we estimate EI and perform numerical comparisons. Finally Section 6.6 concludes the chapter.

6.1 Extremal Index (EI)

We focus on the extremal properties in the correlated and stationary sequence of characteristics of interest $X_1, \dots, X_n$ which is a function of the node sequence, the one actually generated by sampling algorithms. The characteristics of interest, for instance, can be node degrees, node income, number of followers of the node in OSNs, etc. Among the properties, clusters of exceedances of such sequences over high thresholds are studied in particular. The cluster of exceedances is roughly defined as the consecutive exceedances of $\{X_n\}$ over the threshold $\{u_n\}$ between two consecutive non-exceedances. For more rigorous definitions, see [Beirlant et al. 2006, Ferro & Segers 2003, Markovich 2014]. It is important to investigate stochastic nature of extremes since it allows us to collect statistics or opinions more effectively in the clustered (network sampling) process.

The dependence structure of sampled sequence exceeding sufficiently high thresholds is measured using a parameter called extremal index (EI), θ. It is defined in extreme value theory as follows.

Definition 6.1. [Leadbetter et al. 1983, p. 53] The stationary sequence $\{X_n\}_{n\geqslant 1}$, with F as the marginal distribution function and $M_n = \max\{X_1, \ldots, X_n\}$, is said to have the extremal index $\theta \in [0, 1]$ if for each $0 < \tau < \infty$ there is a sequence of real numbers (thresholds) $u_n = u_n(\tau)$ such that

$$\lim_{n\to\infty} n(1 - F(u_n)) = \tau \text{ and} \tag{6.1}$$

$$\lim_{n\to\infty} \mathbb{P}\{M_n \leqslant u_n\} = e^{-\theta\tau}. \tag{6.2}$$

The maximum M_n is related to EI more clearly as [Beirlant et al. 2006, p. 381][1]

$$\mathbb{P}\{M_n \leqslant u_n\} = F^{n\theta}(u_n) + o(1). \tag{6.3}$$

When $\{X_n\}_{n\geqslant 1}$ is i.i.d. (for instance, in uniform independent node sampling), $\theta = 1$ and point processes of exceedances over threshold u_n converges weakly to homogeneous Poisson process with rate τ as $n \to \infty$ [Beirlant et al. 2006, Chapter 5]. But when $0 \leqslant \theta < 1$, point processes of exceedances converges weakly to compound Poisson process with rate $\theta\tau$ and this implies that exceedances of high threshold values u_n tend to occur in clusters for dependent data [Beirlant et al. 2006, Chapter 10].

EI has many useful interpretations and applications like

- Finding distribution of order statistics of the sampled sequence. These can be used to find quantiles and predict the kth largest value which arise with a certain probability. Specifically for the distribution of maximum, equation (6.3) is available and the quantile of maximum is proportional to EI. Hence in case of samples with lower EI, lower values of maximum can be expected. When sampled sequence is the sequence of node degrees, these give many useful results.

- Close relation to the distribution and expectation of the size of clusters of exceedances (see for e.g. [Beirlant et al. 2006, Markovich 2014]).

- Characterization of the first hitting time of the sampled sequence to (u_n, ∞). Thus in case of applications where the aim is to detect large values of samples

[1] $F^k(.)$ indicates kth power of $F(.)$ throughout the chapter except when $k = -1$ where it denotes the inverse function.

quickly, without actually employing sampling (which might be very costly), we can compare different sampling procedures by EI: smaller EI leads to longer waiting of the first hitting time.

These interpretations are explained later in the chapter. The network topology as well as the sampling method determines the stationary distribution of the characteristics of interest under a sampling technique and is reflected on the EI.

6.2 Calculation of Extremal Index

We consider networks represented by an undirected graph $G = (V, E)$ with $|V|$ vertices and $|E|$ edges. The edges of a graph are assumed to be unordered pairs of nodes. Since the networks under consideration are huge, we assume it is impossible to describe them completely, i.e., no adjacency matrix is given beforehand. Assume any randomized sampling procedure is employed and let the sampled sequence $\{X_i\}$ be any general sequence.

This section explains a way to calculate EI from the bivariate joint distribution if the sampled sequence admits two mixing conditions.

Condition $(D(u_n))$**.**

$$\Big|\mathbb{P}(X_{i_1} \leqslant u_n, \dots, X_{i_p} \leqslant u_n, X_{j_1} \leqslant u_n, \dots, X_{j_q} \leqslant u_n) \\ - \mathbb{P}(X_{i_1} \leqslant u_n, \dots, X_{i_p} \leqslant u_n)\mathbb{P}(X_{j_1} \leqslant u_n, \dots, X_{j_q} \leqslant u_n)\Big| \leqslant \alpha_{n,l_n},$$

where $\alpha_{n,l_n} \to 0$ for some sequence $l_n = o(n)$ as $n \to \infty$, for any integers $i_1 \leqslant \ldots < i_p < j_1 < \ldots \leqslant j_q$ with $j_1 - i_p > l_n$.

Condition $(D''(u_n))$**.**

$$\lim_{n\to\infty} n \sum_{m=3}^{r_n} \mathbb{P}(X_1 > u_n \geqslant X_2, X_m > u_n) = 0,$$

where $(n/r_n)\alpha_{n,l_n} \to 0$ and $l_n/r_n \to 0$ with α_{n,l_n}, l_n as in Condition $D(u_n)$ and r_n as $o(n)$.

Let $C(u, v)$ be a bivariate copula [Nelsen 2007] ($[0,1]^2 \to [0,1]$) and $\mathbf{1}\cdot\nabla C(u, v)$ is its directional derivative along the direction $(1, 1)$. Using Sklar's theorem [Nelsen 2007, p. 18], with F as the marginal stationary distribution function of the sampling process, the copula is given by

$$C(u, v) = \mathbb{P}(X_1 \leqslant F^{-1}(u), X_2 \leqslant F^{-1}(v)),$$

where F^{-1} denotes the inverse function of F. This representation is unique if the stationary distribution $F(x)$ is continuous.

Theorem 6.2. *If the sampled sequence is stationary and satisfies conditions $D(u_n)$ and $D''(u_n)$, and the limits in (6.1) and (6.2) take place, then the extremal index is given by*

$$\theta = \mathbf{1} \cdot \nabla C(1,1) - 1, \tag{6.4}$$

and $0 \leqslant \theta \leqslant 1$.

Proof. For a stationary sequence $\{X_n\}$ holding conditions $D(u_n)$ and $D''(u_n)$, if the limits in (6.1) and (6.2) take place, $\theta = \lim_{n\to\infty} \mathbb{P}(X_2 \leqslant u_n | X_1 > u_n)$ [Leadbetter & Nandagopalan 1989]. Then, we have

$$\begin{aligned}
\theta &= \lim_{n\to\infty} \frac{\mathbb{P}(X_2 \leqslant u_n, X_1 > u_n)}{\mathbb{P}(X_1 > u_n)} \\
&= \lim_{n\to\infty} \frac{\mathbb{P}(X_2 \leqslant u_n) - \mathbb{P}(X_1 \leqslant u_n, X_2 \leqslant u_n)}{\mathbb{P}(X_1 > u_n)} \\
&= \lim_{n\to\infty} \frac{\mathbb{P}(X_2 \leqslant u_n) - C(\mathbb{P}(X_1 \leqslant u_n), \mathbb{P}(X_2 \leqslant u_n))}{1 - \mathbb{P}(X_1 \leqslant u_n)} \\
&= \lim_{x\to 1} \frac{x - C(x,x)}{1-x} \\
&= \mathbf{1} \cdot \nabla C(1,1) - 1,
\end{aligned}$$

which completes the proof. □

Remark 6.3. Condition $D''(u_n)$ can be made weaker to $D^{(k)}(u_n)$ presented in [Chernick et al. 1991],

$$\lim_{n\to\infty} n\mathbb{P}\left(X_1 > u_n \geqslant \max_{2\leqslant i \leqslant k} X_i, \max_{k+1 \leqslant j \leqslant r_n} X_j > u_n\right) = 0,$$

where r_n is defined as in $D''(u_n)$. For the stationary sequence $D^{(2)}(u_n)$ is identical to $D''(u_n)$. If we assume $D^{(k)}$ is satisfied for some $k \geqslant 2$ along with $D(u_n)$, then following the proof of Theorem 6.2, EI can be derived as

$$\theta = \mathbf{1} \cdot \nabla C_k(1, \dots, 1) - \mathbf{1} \cdot \nabla C_{k-1}(1, \dots, 1),$$

where $C_k(x_1, \dots, x_k)$ represents the copula of k-dimensional vector $(x_1, \dots, x_k)$, C_{k-1} is its $(k-1)$th marginal, $C_{k-1}(x) = C_{k-1}(x_1, \dots, x_{k-1}, 1)$ and $\mathbf{1} \cdot \nabla C_k(x_1, \dots, x_k)$ denotes the directional derivative of $C_k(x_1, \dots, x_k)$ along the k-dimensional vector $(1, 1, \dots, 1)$.

In some cases it is easy to work with the joint tail distribution. Survival copula $\widehat{C}(\cdot, \cdot)$ which corresponds to

$$\mathbb{P}(X_1 > x, X_2 > x) = \widehat{C}(\overline{F}(x), \overline{F}(x)),$$

with $\overline{F}(x) = 1 - F(x)$, can also be used to calculate θ. It is related to copula as $\widehat{C}(u,u) = C(1-u, 1-u) + 2u - 1$ [Nelsen 2007, p. 32]. Hence $\theta = \mathbf{1} \cdot \nabla C(1,1) - 1 = 1 - \mathbf{1} \cdot \nabla \widehat{C}(0,0)$.

Lower tail dependence function of survival copula is defined as [Weng & Zhang 2012]

$$\lambda(u_1, u_2) = \lim_{t \to 0^+} \frac{\widehat{C}(tu_1, tu_2)}{t}.$$

Hence $\mathbf{1} \cdot \nabla \widehat{C}(0,0) = \lambda(1,1)$. λ can be calculated for different copula families. In particular, if $\widehat{C}$ is a bivariate Archimedean copula, then it can be represented as, $\widehat{C}(u_1, u_2) = \psi(\psi^{-1}(u_1) + \psi^{-1}(u_2))$, where ψ is the generator function and ψ^{-1} is its inverse with $\psi : [0, \infty] \to [0, 1]$ meeting several other conditions. If ψ is a regularly varying distribution with index $-\beta$, $\beta > 0$, then $\lambda(x_1, x_2) = (x_1^{-\beta^{-1}} + x_2^{-\beta^{-1}})^{-\beta}$ and (X_1, X_2) has a bivariate regularly varying distribution [Weng & Zhang 2012]. Therefore, for Archimedean copula family, EI is given by

$$\theta = 1 - 1/2^{\beta}. \tag{6.5}$$

As an example, bivariate Pareto distribution of the form $\mathbb{P}(X_1 > x_1, X_2 > x_2) = (1 + x_1 + x_2)^{-\gamma}$, $\gamma > 0$ has Archimedean copula with generator function $\psi(x) = (1 + x)^{-\gamma}$. This gives $\theta = 1 - 1/2^{\gamma}$. Bivariate exponential distribution of the form

$$\mathbb{P}(X_1 > x_1, X_2 > x_2) = 1 - e^{-x_1} - e^{-x_2} + e^{-(x_1 + x_2 + \eta x_1 x_2)},$$

$0 \leqslant \eta \leqslant 1$, also admits Archimedian copula.

6.2.1 Check of conditions $D(u_n)$ and $D''(u_n)$ for functions of Markov samples

If the sampling technique is assumed to be based on a Markov chain and the sampled sequence is a measurable function of stationary Markov samples, then such a sequence is stationary and [O'Brien 1987] proved that another mixing condition $AIM(u_n)$ which implies $D(u_n)$ is satisfied. Condition $D''(u_n)$ allows clusters with consecutive exceedances and eliminates the possibility of clusters with upcrossing of the threshold u_n ($X_i \leqslant u_n < X_{i+1}$). Hence in those cases, where it is tedious to check the condition $D''(u_n)$ analytically, we can use numerical procedures to measure ratio of number of consecutive exceedances to number of exceedances and the ratio of number of upcrossings to number of consecutive exceedances in small intervals. Such an example is provided in Section 6.3.3.

Remark 6.4. The EI derived in [Ferreira & Ferreira 2007] has the same expression as in (6.4). But [Ferreira & Ferreira 2007] assumes $\{X_n\}$ is sampled from a first order Markov chain. We relax the Markov property requirement to D and D'' conditions and the example below demonstrates a hidden Markov chain can satisfy D and D''.

Let us consider a hidden Markov chain with the observations $\{X_k\}_{k \geqslant 1}$ and the underlying homogeneous Markov chain as $\{Y_k\}_{k \geqslant 1}$ in stationarity. The underlying Markov chain has finite state space (since we are interested in sampling in finite graphs), but the conditional distributions of the observations $\mathbb{P}(X_k \leqslant x | Y_k = y) = F_y(x)$ have infinite support and condition (6.1) holds for $F_y(x)$.

Proposition 6.5. *When condition* (6.1) *holds for $F_y(x)$, the observation sequence $\{X_k\}_{k\geqslant 1}$ of the hidden Markov chain satisfies Condition D''.*

Proof. Let the transition probability matrix of $\{Y_k\}_{k\geqslant 1}$ be P (with $\mathbb{P}(Y_2 = j|Y_1 = i) = P_{ij}$) and the stationary distribution be π (with $\mathbb{P}(Y_1 = i) = \pi_i$). We have,

$$\begin{aligned}
&\mathbb{P}(X_1 > u_n \geqslant X_2, X_m > u_n)\\
&\quad= \sum_{i,j,k} \mathbb{P}(Y_1 = i, Y_2 = j, Y_m = k)\mathbb{P}(X_1 > u_n \geqslant X_2, X_m > u_n|Y_1, Y_2, Y_m)\\
&\quad= \sum_{i,j,k} \pi_i P_{ij} P_{jk}^{(m-2)} \mathbb{P}(X_1 > u_n|Y_1 = i)\mathbb{P}(X_2 \leqslant u_n|Y_2 = j)\mathbb{P}(X_m > u_n|Y_m = k)\\
&\quad\sim \sum_{i,j,k} \pi_i P_{ij} P_{jk}^{(m-2)} \frac{\tau}{n}\left(1 - \frac{\tau}{n}\right)\frac{\tau}{n}, \quad n \to \infty.
\end{aligned}$$

Thus

$$\lim_{n\to\infty} n \sum_{m=3}^{r_n} \mathbb{P}(X_1 > u_n \geqslant X_2, X_m > u_n) = 0,$$

since $r_n = o(n)$, which completes the proof. □

Proposition 6.5 essentially tells that if the graph is explored by a Markov chain based sampling algorithm and the samples are taken as any measurable functions of the underlying Markov chain, satisfying condition (6.1), then Condition D'' holds. Measurable functions, for example, can represent various attributes of the nodes such as income or frequency of messages in social networks.

6.3 Degree Correlations

The results established in Section 6.2 for finding EI is very general, applicable to any sampling techniques and any sequence of samples which satisfy certain conditions. In this section we illustrate the calculation of EI for dependencies among degrees. We revise different sampling techniques. We focus on the sampled degree sequence and denote the sampled sequence $\{X_i\}$ as $\{D_i\}$ in this section.

6.3.1 Description of the configuration model with degree-degree correlation

To test the proposed approaches and the derived formulas, we use a synthetically generated Configuration type random graph with a given joint degree-degree probability distribution, which takes into account correlation in degrees between neighbor nodes. The dependence structure in the graph is described by the joint degree-degree probability density function $f(d_1, d_2)$ with d_1 and d_2 indicating the degrees of adjacent nodes or equivalently by the corresponding tail distribution function $\overline{F}(d_1, d_2) = \mathbb{P}(D_1 \geqslant d_1, D_2 \geqslant d_2)$ with D_1 and D_2 representing the degree random variables (see e.g., [Barrat et al. 2008, Boguna et al. 2003, Goltsev et al. 2008]).

The probability that a randomly chosen edge has the end vertices with degrees $d_1 \leqslant d \leqslant d_1 + \Delta(d_1)$ and $d_2 \leqslant d \leqslant d_2 + \Delta(d_2)$ is $(2 - \delta_{d_1 d_2}) f(d_1, d_2) \Delta(d_1) \Delta(d_2)$. Here $\delta_{d_1 d_2} = 1$ if $d_1 = d_2$, zero otherwise. The multiplying factor 2 appears on the above expression when $d_1 \neq d_2$ because of the symmetry in $f(d_1, d_2)$, $f(d_1, d_2) = f(d_2, d_1)$ due to the undirected nature of the underlying graph, and the fact that both $f(d_1, d_2)$ and $f(d_2, d_1)$ contribute to the edge probability under consideration.

The degree density $f_d(d_1)$ can be related to the marginal of $f(d_1, d_2)$ as follows:

$$f(d_1) = \int_{d_2} f(d_1, d_2) d(d_2) \approx \frac{d_1 f_d(d_1)}{\mathbb{E}[D]}, \tag{6.6}$$

where $\mathbb{E}[D]$ denotes the mean node degree,

$$\mathbb{E}[D] = \left[\int \int \left(\frac{f(d_1, d_2)}{d_1} \right) d(d_1) d(d_2) \right]^{-1}.$$

$f(.)$ can be interpreted as the degree density of a vertex reached by following a randomly chosen edge. The approximation for $f(d_1)$ is obtained as follows: in the R.H.S. of (6.6), roughly, $d_1 f_d(d_1)|V|$ is the number of half edges from nodes with degree around d_1 and $\mathbb{E}[D]|V|$ is the total number of half edges. For discrete distributions, (6.6) becomes equality.

From the above description, it can be noted that the knowledge of $f(d_1, d_2)$ is sufficient to describe this random graph model and for its generation.

Most of the results in this chapter are derived assuming continuous probability distributions for $f(d_1, d_2)$ and $f_d(d_1)$ because an easy and unique way to calculate EI exists for continuous distributions in our setup (more details in Section 6.2). Also the EI might not exist for many discrete valued distributions [Leadbetter et al. 1983].

6.3.1.1 Random graph generation

A random graph with bivariate joint degree-degree distribution can be generated as follows ([Newman 2002]):

1. Degree sequence is generated according to the degree distribution, $f_d(d) = \frac{f(d)E[D]}{d}$

2. An uncorrelated random graph is generated with the generated degree sequence using configuration model ([Barrat et al. 2008, Van Der Hofstad 2016])

3. Metropolis dynamics is now applied on the generated graph: choose two edges randomly (denoted by the vertex pairs (v_1, w_1) and (v_2, w_2)) and measure the degrees, (j_1, k_1) and (j_2, k_2) correspond to these vertex pairs. Generate a random number, y, according to uniform distribution in $[0, 1]$. If $y \leqslant \min(1, (f(j_1, j_2) f(k_1, k_2))/(f(j_1, k_1) f(j_2, k_2)))$, then remove the selected edges and construct news ones as (v_1, v_2) and (w_1, w_2). Otherwise keep the selected edges intact. This dynamics will generate an instance of the random graph

with the required joint degree-degree distribution. Run Metropolis dynamics long enough to mix the generating process.

As an example, we shall often use the following bivariate Pareto model for the joint degree-degree tail function of the graph,

$$\bar{F}(d_1, d_2) = \left(1 + \frac{d_1 - \mu}{\sigma} + \frac{d_2 - \mu}{\sigma}\right)^{-\gamma}, \tag{6.7}$$

where σ, μ and γ are positive values. The use of the bivariate Pareto distribution can be justified by the statistical analysis in [Zhukovskiy et al. 2012].

6.3.2 Description of random walk based sampling processes

In this section, we explain three different one dimensional random walk based algorithms for exploring the network: simple random walk, PageRank and random walk with jumps. They have been extensively studied in previous works [Avrachenkov et al. 2010, Brin & Page 1998, Lovász 1993] (also briefly mentioned in Chapter 2) where they are formulated with vertex set as the state space of the underlying Markov chain on graph. The walker in these algorithms, after reaching each node, moves to another node randomly by following the transition kernel of the Markov chain. However, the quantity of interest is generally a measurable function of the Markov chain. As a case study, let us again take the degree sequence. We use $f_{\mathscr{X}}$ and $\mathbb{P}_{\mathscr{X}}$to represent the probability density function and probability measure under the algorithm $\mathscr{X}$ with the exception that f_d represents the probability density function of degrees.

6.3.2.1 Simple Random walk (SRW)

In a simple random walk, the next node to visit is chosen uniformly among the neighbors of the current node. Let $V_1, V_2, \ldots$ be the nodes crawled by the SRW and $D_1, D_2, \ldots$ be the degree sequence corresponding to the sequence $V_1, V_2, \ldots$.

Theorem 6.6. *The following relation holds in the stationary regime*

$$f_{\text{SRW}}(d_1, d_2) = f(d_1, d_2), \tag{6.8}$$

where $f(d_1, d_2)$ is the joint degree-degree distribution and $f_{SRW}(d_1, d_2)$ is the bivariate joint distribution of the degree sequences generated by the standard simple random walk.

Proof. We note that the sequence $\{(V_i, V_{i+1})\}_{i \geqslant 1}$ also forms a Markov chain. With the assumption that the graph is connected, the ergodicity holds for any function g, i.e.,

$$\frac{1}{T}\sum_{i=1}^{T} g(V_i, V_{i+1}) \to \mathbb{E}_\pi[g(V_\xi, V_{\xi+1})], \quad T \to \infty,$$

where $\mathbb{E}_\pi$ is the expectation under stationary distribution π of $\{(V_i, V_{i+1})\}$ (which is uniform over edges) and $(V_\xi, V_{\xi+1})$ indicates a randomly picked edge. The ergodicity can then be extended to functions of the degree sequence $\{(D_i, D_{i+1})\}$ corresponding to $\{(V_i, V_{i+1})\}$, and in particular

$$\begin{aligned}\frac{1}{T}\sum_{i=1}^{T}\mathbb{I}\{D_i = d_1, D_{i+1} = d_2\} \rightarrow \mathbb{E}_{\pi}[\mathbb{I}\{D_{\xi} = d_1, D_{\xi+1} = d_2\}], \quad T \rightarrow \infty \\ = \frac{1}{|E|}\sum_{(p,q)\in E}\mathbb{I}\{D_p = d_1, D_q = d_2\} \\ = f(d_1, d_2), \qquad (6.9)\end{aligned}$$

where $\mathbb{I}\{\mathcal{A}\}$ denotes the indicator function for the event $\mathcal{A}$. L.H.S. of (6.9) is an estimator of $f_{\text{SRW}}(d_1, d_2)$. This means that when the SRW is in stationary regime $\mathbb{E}[\mathbb{I}\{D_i = d_1, D_{i+1} = d_2\}] = \mathbb{E}_{\pi}[\mathbb{I}\{D_{\xi} = d_1, D_{\xi+1} = d_2\}]$ and hence (6.8) holds. □

Using (6.6) we can approximate the degree sequence by a simple random walk on degree space with the following transition kernel:

$$f_{\text{SRW}}(d_{t+1}|d_t) = \frac{\mathbb{E}[D]f(d_t, d_{t+1})}{d_t f_d(d_t)}, \qquad (6.10)$$

where the present node has degree d_t and the next node is with degree d_{t+1}. The above relation holds with equality for discrete degree distribution, but some care needs to be taken if one uses continuous version for the degree distributions.

If the simple random walk on the vertex set is in the stationary regime, its stationary distribution (probability of staying at a particular vertex i) is proportional to the degree (see e.g., [Lovász 1993]) and is given by $d_i/2|E|$, $|E|$ being the number of edges. Then in the simple random walk on degree set, the stationary distribution of staying at any node with degree around d_1 can be approximated as $|V|f_d(d_1)\,(d_1/2|E|)$, with $|V|$ as the number of nodes. Thus

$$f_{\text{SRW}}(d_1) = \frac{d_1}{\mathbb{E}[D]}f_d(d_1).$$

Check of the approximation

We provide comparison of simulated values and theoretical values of transition kernel of SRW in Figure 6.1. To be specific, we use the bivariate Pareto distribution given (6.7). In the figure, $|V|$ is 5,000. $\mu = 10$, $\gamma = 1.2$ and $\sigma = 15$. These choices of parameters provide $E[D] = 21.0052$. At each instant Metropolis dynamics will choose two edges and it has run 200,000 times (provides sufficient mixing). The figure shows satisfactory fitting of the approximation.

6.3.2.2 PageRank (PR)

PageRank is a modification of the simple random walk which with a fixed probability $1 - c$ samples a random node with uniform distribution and with a probability c,

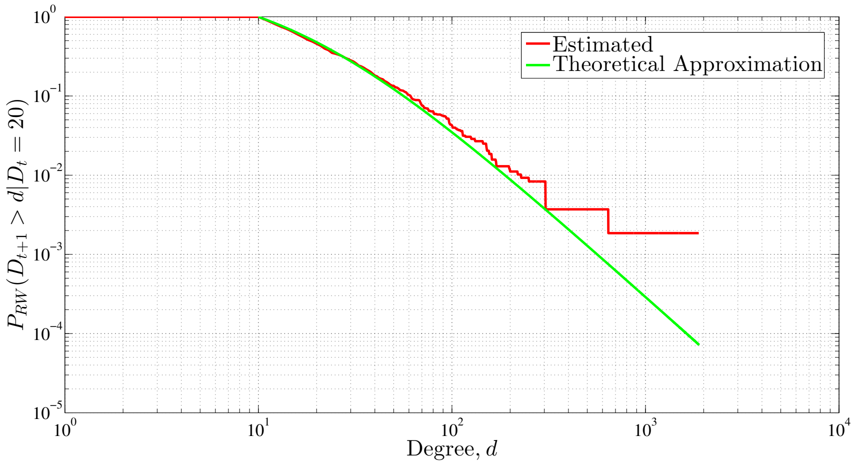


Figure 6.1: Transition kernel comparison

it follows the simple random walk transition [Brin & Page 1998]. Its evolution on degree state space can be described as follows:

$$\begin{aligned} f_{\text{PR}}(d_{t+1}|d_t) &= c\, f_{\text{SRW}}(d_{t+1}|d_t) + (1-c)\frac{1}{|V|}|V|f_d(d_{t+1}) \\ &= c\, f_{\text{SRW}}(d_{t+1}|d_t) + (1-c)f_d(d_{t+1}). \end{aligned} \tag{6.11}$$

Here the $1/|V|$ corresponds to the uniform sampling on vertex set and $\frac{1}{|V|}|V|f_d(d_{t+1})$ indicates the net probability of jumping to all the nodes with degree around d_{t+1}.

Consistency with PageRank value distribution

We make a consistency check of the approximation derived for transition kernel by studying tail behavior of degree distribution and PageRank value distribution. It is known that under some strict conditions, for a directed graph, PageRank and Indegree have same tail exponents [Litvak et al. 2007]. In our formulation in terms of degrees, for *uncorrelated* and undirected graph, PageRank for a given degree d, $PR(d)$, can be approximated from the basic definition as,

$$PR(d) = f_{\text{PR}}(d) = c\, f_{\text{SRW}}(d) + (1-c)\, f_d(d).$$

This is a deterministic quantity. We are interested in the distribution of the random variable $PR(D)$, PageRank of a randomly choosen degree class D. PageRank $PR(d)$ is also the long term proportion or probability that PageRank process ends in a degree class with degree d. This can be scaled suitably to provide a rank-type information. Its tail distribution is

$$\mathbb{P}(PR(D) > x) = \mathbb{P}\left(c.f_{\text{SRW}}(D) + (1-c).f_d(D) > x\right),$$

where $D \sim f_d(.)$. The PageRank of any vertex inside the degree class d is $PR(d)/(|V|f_d(d))$. The distribution of Page Rank of a randomly chosen vertex

i, $\mathbb{P}(PR(i) > x)$ after appropriate scaling for comparison with degree distribution is $\mathbb{P}(|V|.PR(i) > \hat{d})$, where $\hat{d} = |V|x$. Now

$$\begin{aligned}\mathbb{P}(|V|.PR(i) > \hat{d}) &= \mathbb{P}\left(|V|\frac{PR(D)}{|V|f_d(D)} > \hat{d}\right)\\ &= \mathbb{P}\left(D > \frac{\mathbb{E}[D]}{c}\left[\hat{d} - (1-c)\right]\right).\end{aligned}$$

This of the form $\mathbb{P}(D > A\hat{d} + B)$ with A and B as appropriate constants and hence will have the same exponent of degree distribution tail when the graph is *uncorrelated.*

There is no convenient expression for the stationary distribution of PageRank, to the best of our knowledge, and it is difficult to come up with an easy to handle expression for the joint distribution. Therefore, along with other advantages, we consider another modification of the simple random walk.

6.3.2.3 Random walk with jumps (RWJ)

SRW sampling leads to many practical issues like the possibility to get stuck in a disconnected component, biased estimators, etc. RWJ overcomes such problems ([Avrachenkov et al. 2010]).

In this algorithm we follow simple random walk on a modified graph which is a superposition of the original graph and complete graph on same vertex set of the original graph with weight $\alpha/|V|$ on each artificially added edge, $\alpha \in [0, \infty]$ being a design parameter ([Avrachenkov et al. 2010]). The algorithm can be shown to be equivalent to select $c = \alpha/(d_t + \alpha)$ in the PageRank algorithm, where d_t is the degree of the node at time t[2]. The larger the node's degree, the less likely is the artificial jump of the process. This modification makes the underlying Markov chain time reversible, significantly reduces mixing time, improves estimation error and leads to a closed form expression for stationary distribution.

Before proceeding to formulate the next theorem, we recall that the degree distribution $f_d(d_1)$ is different from the marginal of $f(d_1, d_2)$, $f(d_1)$.

Theorem 6.7. *The following relation holds in the stationary regime*

$$f_{\mathrm{RWJ}}(d_1, d_2) = \frac{\mathbb{E}[D]}{\mathbb{E}[D] + \alpha} f(d_1, d_2) + \frac{\alpha}{\mathbb{E}[D] + \alpha} f_d(d_1) f_d(d_2), \qquad (6.12)$$

where $f(d_1, d_2)$ is the joint degree-degree distribution, $f_d(d_1)$ is the degree distribution and $f_{RWJ}(d_1, d_2)$ is the bi-variate joint distribution of the degree sequences generated by the random walk with jumps.

[2] This is a slight abuse of notation. In the previous chapters we used d_v to indicate degree of node v. Here, as we are directly focusing on the degree sequence, it makes more sense to define d_t as the degree of the node sampled at time t

Proof. In the similar lines in the analysis of SRW, $f_{\mathrm{RWJ}}(d_1, d_2)$ can be calculated as follows. The stationary distribution, $f_{\mathrm{RWJ}}(p)$, for node p (on the vertex set) is $(d_p+\alpha)/(2|E|+|V|\alpha)$. The transition probability from node p to node q, $f_{\mathrm{RWJ}}(q|p)$, is $(\alpha/|V|+1)/(d_p+\alpha)$ when there is a link from p to q, and when there is no link, it is $(\alpha/|V|)/(d_p+\alpha)$ ([Avrachenkov et al. 2010]). Then the joint distribution between nodes is given by

$$f_{\mathrm{RWJ}}(p,q) = f_{\mathrm{RWJ}}(q|p) f_{\mathrm{RWJ}}(p) = \begin{cases} \frac{\frac{\alpha}{|V|}+1}{2|E|+|V|\alpha} & \text{if } p \text{ has link to } q, \\ \frac{\frac{\alpha}{|V|}}{2|E|+|V|\alpha} & \text{if } p \text{ does not have link to } q. \end{cases}$$

Therefore

$$\begin{aligned} & f_{\mathrm{RWJ}}(d_1, d_2) \\ & = \mathbb{E}_\pi[\mathbb{I}\{D_\xi = d_1, D_{\xi+1} = d_2\}] \\ & \overset{(a)}{=} 2\,\frac{\frac{\alpha}{|V|}+1}{2|E|+|V|\alpha} \sum_{(p,q)\in E} \mathbb{I}\{D_p = d_1, D_q = d_2\} \\ & \qquad\qquad +2\,\frac{\frac{\alpha}{|V|}}{2|E|+|V|\alpha} \sum_{(p,q)\notin E} \mathbb{I}\{D_p = d_1, D_q = d_2\} \\ & \overset{(b)}{=} 2\,\frac{\frac{\alpha}{|V|}+1}{2|E|+|V|\alpha} |E| f(d_1, d_2) \\ & \qquad +2\,\frac{\frac{\alpha}{|V|}}{2|E|+|V|\alpha} \Big(\frac{1}{2}\sum_{p\in V} \mathbb{I}\{D_p = d_1\} \sum_{q\in V} \mathbb{I}\{D_q = d_2\} - |E| f(d_1, d_2)\Big) \\ & = \frac{\mathbb{E}[D]}{\mathbb{E}[D]+\alpha} f(d_1, d_2) + \frac{\alpha}{\mathbb{E}[D]+\alpha} f_d(d_1) f_d(d_2). \end{aligned}$$

Here $\mathbb{E}[D] = 2|E|/|V|$. The multiplying factor 2 is introduced in (a) because of the symmetry in the joint distribution $f_{\mathrm{RWJ}}(p,q)$ over the nodes, terms outside summation in the R.H.S. The factor 1/2 in R.H.S. in (b) is to take into account the fact that only half of the combinations of (p,q) is needed. □

We also have the following. The stationary distribution on degree set by collecting all the nodes with same degree is

$$\begin{aligned} f_{\mathrm{RWJ}}(d_1) &= \Big(\frac{d_1+\alpha}{2|E|+|V|\alpha}\Big) N f_d(d_1) \\ &= \frac{(d_1+\alpha) f_d(d_1)}{\mathbb{E}[D]+\alpha}. \end{aligned} \tag{6.13}$$

Moreover the associated tail distribution has a simple form,

$$f_{\mathrm{RWJ}}(D_{t+1} > d_{t+1}, D_t > d_t) = \frac{\mathbb{E}[D]\overline{F}(d_{t+1}, d_t) + \alpha \overline{F}_d(d_{t+1})\overline{F}_d(d_t)}{\mathbb{E}[D]+\alpha}. \tag{6.14}$$

Remark 6.8. For uncorrelated networks, $f_{\mathrm{SRW}}(d_1, d_2) = f_{\mathrm{SRW}}(d_1) f_{\mathrm{SRW}}(d_2)$, $f_{\mathrm{PR}}(d_1, d_2) = f_{\mathrm{PR}}(d_1) f_{\mathrm{PR}}(d_2)$ and $f_{\mathrm{RWJ}}(d_1, d_2) = f_{\mathrm{RWJ}}(d_1) f_{\mathrm{RWJ}}(d_2)$.

6.3.3 Extremal index for bivariate Pareto degree correlation

As explained in the Introduction section, EI is an important parameter in characterizing dependence and extremal properties in a stationary sequence. We assume that we have waited sufficiently long that the underlying Markov chain of the three different graph sampling algorithms are in stationary regime now. Here we derive EI of SRW and RWJ for the model with degree correlation among neighbours as bivariate Pareto (6.7).

The two mixing conditions $D(u_n)$ and $D''(u_n)$ introduced in Section 6.2 are needed for our EI analysis. Condition $D(u_n)$ is satisfied as explained in Section 6.2.1. An empirical evaluation of $D''(u_n)$ is provided in Section 6.5.3.1.

6.3.3.1 Extremal index for random walk sampling

We use the expression for EI given in Theorem 6.2. As $f_{\text{SRW}}(x,y)$ is same as $f(x,y)$, we have,

$$\begin{aligned}
\widehat{C}(u,u) &= \mathbb{P}(D_1 > \bar{F}^{-1}(u), D_2 > \bar{F}^{-1}(u)) \\
&= \big(1 + 2(u^{-1/\gamma} - 1)\big)^{-\gamma} \\
\mathbf{1} \cdot \nabla\, \widehat{C}(u,u) &= 2(2 - u^{1/\gamma})^{-(\gamma+1)}.
\end{aligned}$$

Thus $\theta = 1 - \mathbf{1} \cdot \nabla\, \widehat{C}(0,0) = 1 - 1/2^{\gamma}$. For $\gamma = 1$ we get $\theta = 1/2$. In this case, we can also use expression obtained in (6.5).

6.3.3.2 Extremal index for random walk with jumps sampling

Although it is possible to derive EI as in SRW case above, we provide an alternative way to avoid the calculation of tail distribution of degrees and inverse of RWJ marginal (with respect to the bivariate Pareto degree correlation). We assume the existence of EI in the following proposition.

Proposition 6.9. *When the bivariate joint degree distributions of neighboring nodes are Pareto distributed as given by* (6.7)*, and random walk with jumps is employed for sampling, the EI is given by*

$$\theta = 1 - \frac{\mathbb{E}[D]}{\mathbb{E}[D] + \alpha} 2^{-\gamma}, \tag{6.15}$$

where $\mathbb{E}[D]$ is the expected degree, α is the parameter of the random walk with jumps, and γ is the tail index of the bivariate Pareto distribution.

Proof. Under the assumption of D'',

$$\theta = \lim_{n\to\infty} \frac{\mathbb{P}(D_2 \leqslant u_n, D_1 > u_n)}{\mathbb{P}(D_1 > u_n)} = \lim_{n\to\infty} \frac{\mathbb{P}(D_1 \geqslant u_n) - \mathbb{P}(D_2 \geqslant u_n, D_1 \geqslant u_n)}{\mathbb{P}(D_1 > u_n)} \tag{6.16}$$

Now using the condition (6.1) on the marginal and joint tail distribution of RWJ (6.14), we can write

$$\frac{\mathbb{P}(D_1 \geqslant u_n) - \mathbb{P}(D_2 \geqslant u_n, D_1 \geqslant u_n)}{\mathbb{P}(D_1 > u_n)}$$
$$= \frac{\tau/n + o(1/n) - \frac{\mathbb{E}[D]}{\mathbb{E}[D]+\alpha}\mathbb{P}_{\mathrm{SRW}}(D_2 \geqslant u_n, D_1 \geqslant u_n) - \frac{\alpha}{\mathbb{E}[D]+\alpha}\mathcal{O}(\tau/n)\mathcal{O}(\tau/n)}{\tau/n + o(1/n)}$$

The asymptotics in the last term of the numerator is due to the following:

$$\overline{F}_{\mathrm{RWJ}}(u_n) = \frac{\mathbb{E}[D]}{\mathbb{E}[D]+\alpha}\overline{F}(u_n) + \frac{\alpha}{\mathbb{E}[D]+\alpha}\overline{F}_d(u_n) = \tau/n + o(1/n),$$

and hence $\overline{F}_d(u_n) = \mathcal{O}(\tau/n)$. Therefore (6.16) becomes

$$\theta = 1 - \frac{\mathbb{E}[D]}{\mathbb{E}[D]+\alpha} \lim_{n\to\infty} \mathbb{P}_{\mathrm{SRW}}(D_2 \geqslant u_n, D_1 \geqslant u_n)\, n/\tau.$$

Then in the case of the bivariate Pareto distribution (6.7), we obtain (6.15). □

6.3.4 Lower bound of extremal index of the PageRank

We obtain the following lower bound for EI in the PageRank processes.

Proposition 6.10. *For the stationary PageRank process on degree state space (6.10) with EI θ, irrespective of the degree correlation structure in the underlying graph, the EI is bounded by*

$$\theta \geqslant (1-c),$$

where c is the damping factor in the PageRank algorithm.

Proof. From [O'Brien 1987], with another mixing condition $AIM(u_n)$ which is satisfied for functions of stationary Markov samples (e.g., degree samples) the following representation of EI holds,

$$\lim_{n\to\infty} \mathbb{P}\{M_{1,p_n} \leqslant u_n | D_1 > u_n\} \leqslant \theta, \tag{6.17}$$

where $\{p_n\}$ is an increasing sequence of positive integers, $p_n = o(n)$ as $n \to \infty$ and $M_{1,p_n} = \max\{D_2, ..., D_{p_n}\}$. Let $\mathcal{A}$ be the event that the node corresponding to D_2 is selected uniformly among all the nodes, not following random walk from the node for D_1. Then $\mathbb{P}_{\mathrm{PR}}(\mathcal{A}) = 1 - c$. Now, with (6.11),

$$\begin{aligned}
\mathbb{P}_{\mathrm{PR}}(M_{1,p_n} \leqslant u_n | D_1 > u_n) &\geqslant \mathbb{P}_{\mathrm{PR}}(M_{1,p_n} \leqslant u_n, \mathcal{A} | D_1 > u_n) \\
&= \mathbb{P}_{\mathrm{PR}}(\mathcal{A}|D_1 > u_n)\mathbb{P}_{\mathrm{PR}}(M_{1,p_n} \leqslant u_n | \mathcal{A}, D_1 > u_n) \\
&\overset{(i)}{=} (1-c)\mathbb{P}_{\mathrm{PR}}(M_{1,p_n} \leqslant u_n), \\
&\overset{(ii)}{=} (1-c)\mathbb{P}_{\mathrm{PR}}^{(p_n-1)\theta}(D_1 \leqslant u_n) + o(1) \\
&\geqslant (1-c)\mathbb{P}_{\mathrm{PR}}^{(p_n-1)}(D_1 \leqslant u_n) + o(1) \\
&\overset{(iii)}{\sim} (1-c)(1-\tau/n)^{p_n-1},
\end{aligned} \tag{6.18}$$

where $\{p_n\}$ is the same sequence as in (6.17) and (i) follows mainly from the observation that conditioned on $\mathcal{A}$, $\{M_{1,p_n} \leqslant u_n\}$ is independent of $\{D_1 > u_n\}$, (ii) and (iii) result from the limits in (6.3) and (6.1) respectively.

Assuming $p_n - 1 = n^{1/2}$ and since $(1-\tau/n)^{p_n-1} \sim e^{-\tau/\sqrt{n}} \to 1$ as $n \to \infty$, from (6.17) and (6.18),

$$\theta \geqslant 1 - c.$$

The PageRank transition kernel (6.11) on the degree state space does not depend upon the random graph model in Section 6.3.1. Hence the derived lower bound of EI is useful for any degree correlation model. □

6.4 Applications of Extremal Index in Network Sampling Processes

This section provides several applications of EI in inferring the sampled sequence. This emphasizes that the analytical calculation and estimation of EI are practically relevant.

The limit of the point process of exceedances, $N_n(.)$, which counts the times, normalized by n, at which $\{X_i\}_{i=1}^n$ exceeds a threshold u_n provides many applications of EI. A cluster is considered to be formed by the exceedances in a block of size r_n ($r_n = o(n)$) in n with cluster size $\xi_n = \sum_{i=1}^{r_n} 1(X_i > u_n)$ when there is at least one exceedance within r_n. The point process N_n converges weakly to a compound poisson process (CP) with rate $\theta\tau$ and i.i.d. distribution as the limiting distribution of cluster size, under condition (6.1) and a mixing condition, and the points of exceedances in CP correspond to the clusters (see [Beirlant et al. 2006, Section 10.3] for details). We also call this kind of clusters as blocks of exceedances.

The applications below require a choice of the threshold sequence $\{u_n\}$ satisfying (6.1). For practical purposes, if a single threshold u is demanded for the sampling budget B, we can fix $u = \max\{u_1, \dots, u_B\}$.

The applications in this section are explained with the assumption that the sampled sequence is the sequence of node degrees. But the following techniques are very general and can be extended to any sampled sequence satisfying conditions $D(u_n)$ and $D''(u_n)$.

6.4.1 Order statistics of the sampled degrees

The order statistics $X_{n-k,n}$, $(n-k)$th maximum, is related to $N_n(.)$ and thus to θ by

$$\mathbb{P}(X_{n-k,n} \leqslant u_n) = \mathbb{P}(N_n((0,1]) \leqslant k),$$

where we apply the result of convergence of N_n to CP [Beirlant et al. 2006, Section 10.3.1].

6.4.1.1 Distribution of maximum

The distribution of the maximum of the sampled degree sequences can be derived as (6.3) when $n \to \infty$. Hence if the EI of the underlying process is known then from (6.3) one can approximate the $(1-\eta)$th quantile x_η of the maximuml degree M_n as

$$\mathbb{P}\{M_n \leqslant x_\eta\} = F^{n\theta}(x_\eta) = \mathbb{P}^{n\theta}\{X_1 \leqslant x_\eta\} = 1-\eta,$$

i.e.

$$x_\eta \approx F^{-1}\left((1-\eta)^{1/(n\theta)}\right). \tag{6.19}$$

In other words, quantiles can be used to find the maximum of the degree sequence with certain probability.

If the sampling procedures have same marginal distribution, with calculation of EI, it is possible to predict how much large values can be achieved. Lower EI indicates lower value for x_η and higher represents high x_η.

For the simple random walk example in Section 6.3.3.1 for the degree correlation model, with the use of (6.19), we get the $(1-\eta)$th quantile of the maximum M_n

$$x_\eta \approx \mu + \sigma\left(\left(1-(1-\eta)^{1/(n\theta)}\right)^{-1/\gamma} - 1\right).$$

The following example demonstrates the effect of neglecting correlations on the prediction of the largest degree node. The largest degree, with the assumption of Pareto distribution for the degree distribution, can be approximated as $KN^{1/\delta}$ with $K \approx 1$, $|V|$ as the number of nodes and γ as the tail index of complementary distribution function of degrees [Avrachenkov et al. 2014b]. For Twitter graph (recorded in 2012), $\delta = 1.124$ for outdegree distribution and $|V| = 537,523,432$ [Gabielkov et al. 2014]. This gives the largest degree prediction as $59,453,030$. But the actual largest out degree is $22,717,037$. This difference is because the analysis in [Avrachenkov et al. 2014b] assumes i.i.d. samples and does not take into account the degree correlation. With the knowledge of EI, correlation can be taken into account as in (6.3). In the following section, we derive an expression for such a case.

6.4.1.2 Estimation of largest degree when the marginals are Pareto distributed

It is known that many social networks have the degree approximately distributed as Pareto [Van Der Hofstad 2016]. We find that in these cases, the marginal distribution of degrees of the random walk based methods also follow Pareto distribution (though we have derived only for the model with degree correlations among neighbors, see Section 6.3).

Proposition 6.11. *For any stationary sequence with marginal distribution following Pareto distribution $\bar{F}(x) = Cx^{-\delta}$ the largest value, approximated as the median of the extreme value distribution, is given by*

$$M_n \approx (n\theta)^{1/\delta}\left(\frac{C}{\log 2}\right)^{1/\delta}.$$

Proof. From extreme value theory [Beirlant et al. 2006], it is known that when $\{X_i, i \geqslant 1\}$ are i.i.d.,

$$\lim_{n\to\infty} \mathbb{P}\left(\frac{M_n - b_n}{a_n} \leqslant x\right) = H_\gamma(x), \tag{6.20}$$

where $H_\gamma(x)$ is the extreme value distribution with index γ and $\{a_n\}$ and $\{b_n\}$ are appropriately chosen deterministic sequences. When $\{X_i, i \geqslant 1\}$ are stationary with EI θ, the limiting distribution becomes $H'_{\gamma'}(x)$ and it differs from $H_\gamma(x)$ only through parameters. $H_\gamma(x) = \exp(-t(x))$ with $t(x) = \left(1 + (\frac{x-\mu}{\sigma})\gamma\right)^{-1/\gamma}$. With the normalizing constants ($\mu = 0$ and $\sigma = 1$), $H'_{\gamma'}$ has the same shape as H_γ with parameters $\gamma' = \gamma$, $\sigma' = \theta^\gamma$ and $\mu' = (\theta^\gamma - 1)/\gamma$ [Beirlant et al. 2006, Section 10.2.3].

For Pareto case, $\overline{F}(x) = Cx^{-\delta}$, $\gamma = 1/\delta$, $a_n = \gamma C^\gamma n^\gamma$ and $b_n = C^\gamma n^\gamma$. From (6.20), for large n, M_n is stochastically equivalent to $a_n\chi + b_n$, where χ is a random variable with distribution $H'_{\gamma'}$. It is observed in [Avrachenkov et al. 2014b] that median of χ is an appropriate choice for the estimation of M_n. Median of $\chi = \mu' + \sigma'\left(\frac{(\log 2)^{-\gamma'} - 1}{\gamma'}\right) = (\theta^\gamma(\log 2)^{-\gamma} - 1)\gamma^{-1}$. Hence,

$$\begin{aligned} M_n &\approx a_n\left(\frac{\theta^\gamma(\log 2)^{-\gamma}}{\gamma} - 1\right) + b_n \\ &= (n\theta)^{1/\delta}\left(\frac{C}{\log 2}\right)^{1/\delta} \qquad \square \end{aligned}$$

6.4.2 Relation to first hitting time and interpretations

Extremal index also gives information about the first time $\{X_n\}$ hits (u_n, ∞). Let T_n be this time epoch. As N_n converges to compound poisson process, it can be observed that T_n/n is asymptotically an exponential random variable with rate $\theta\tau$, i.e., $\lim_{n\to\infty} \mathbb{P}(T_n/n > x) = \exp(-\theta\tau x)$. Therefore $\lim_{n\to\infty} \mathbb{E}(T_n/n) = 1/(\theta\tau)$. For e.g., consider the Pareto distribution for $\{X_i\}$'s, then $\mathbb{P}(X_i \geqslant u_n) = u_n^{-\alpha}$ with $u_n = (n)^{1/\alpha}$ for $\tau = 1$ (see (6.1)) and for some parameter $\alpha > 0$. Thus the smaller EI is, the longer it will take to hit the extreme levels as compared to independent sampling. This property is particularly useful to compare different sampling procedures. It can also be used in quick detection of high degree nodes [Avrachenkov et al. 2014b, Avrachenkov et al. 2014a].

6.4.3 Relation to mean cluster size

If Condition $D''(u_n)$ is satisfied along with $D(u_n)$, asymptotically, a run of the consecutive exceedances following an upcrossing is observed, i.e., $\{X_n\}$ crosses the threshold u_n at a time epoch and stays above u_n for some more time before crossing u_n downwards and stays below it for some time until next upcrossing of u_n happens. This is called cluster of exceedances and is more practically relevant than blocks of exceedances at the starting of this section and is shown in [Leadbetter & Nandagopalan 1989] that these two definitions clusters are asymptotically equivalent resulting in similar cluster size distribution. The expected value of cluster of exceedances converges to inverse of EI [Beirlant et al. 2006, p. 384], i.e.,

$$\theta^{-1} = \lim_{n\to\infty} \sum_{j\geqslant 1} j\pi_n(j),$$

where $\{\pi_n(j), j \geqslant 1\}$ is the distribution of size of cluster of exceedances with n samples. Asymptotical cluster size distribution and its mean are derived in [Markovich 2014].

6.5 Estimation of Extremal Index and Numerical Results

This section introduces two estimators for EI. Two types of networks are presented: synthetic correlated graph and real networks (Enron email network and DBLP network). For the synthetic graph, we compare the estimated EI to its theoretical value. For the real network, we calculate EI using the two estimators.

We take $\{X_i\}$ as the degree sequence and use SRW, PR and RWJ as the sampling techniques. The methods mentioned in the following are general and are not specific to degree sequence or random walk technique.

6.5.1 Empirical copula based estimator

We have tried different estimators for EI available in literature [Beirlant et al. 2006, Ferreira & Ferreira 2007] and found that the idea of estimating copula and then finding value of its derivative at $(1, 1)$ works without the need to choose and optimize several parameters found in other estimators. We assume that $\{X_i\}$ satisfies $D(u_n)$ and $D''(u_n)$ and we use (6.4) for calculation of EI. Copula $C(u, v)$ is estimated empirically by

$$C_n(u,v) = \frac{1}{n}\sum_{k=1}^{n} \mathbb{I}\left(\frac{R^X_{i_k}}{n+1} \leqslant u, \frac{R^Y_{i_k}}{n+1} \leqslant v\right),$$

with $R^X_{i_k}$ indicates rank of the element X_{i_k} in $\{X_{i_k}, 1 \leqslant k \leqslant n\}$ and $R^Y_{i_k}$ is defined respectively. The sequence $\{X_{i_k}\}$ is chosen from the original sequence $\{X_i\}$ in such a way that X_{i_k} and $X_{i_{k+1}}$ are sufficiently apart to make them independent to certain extent and $Y_{i_k} = X_{i_k+1}$. The large-sample distribution of $C_n(u, v)$ is normal and centered at copula $C(u, v)$. Now, to get θ, we use linear least squares error fitting to find slope at $(1, 1)$ or use cubic spline interpolation for better results.

6.5.2 Intervals estimator

This estimator does not assume any conditions on $\{X_i\}$, but has the parameter u to choose appropriately. Let $|V| = \sum_{i=1}^{n} 1(X_i > u)$ be number of exceedances of u at time epochs $1 \leqslant S_1 < \ldots < S_N \leqslant n$ and let the interexceedance times are $T_i = S_{i+1} - S_i$. Then intervals estimator is defined as [Beirlant et al. 2006, p. 391],

$$\hat{\theta}_n(u) = \begin{cases} \min(1, \hat{\theta}^1_n(u)), & \text{if } \max T_i : 1 \leqslant i \leqslant |V|-1 \leqslant 2, \\ \min(1, \hat{\theta}^2_n(u)), & \text{if } \max T_i : 1 \leqslant i \leqslant |V|-1 > 2, \end{cases}$$

where

$$\hat{\theta}_n^1(u) = \frac{2(\sum_{i=1}^{|V|-1} T_i)^2}{(|V|-1)\sum_{i=1}^{|V|-1} T_i^2},$$

and

$$\hat{\theta}_n^2(u) = \frac{2(\sum_{i=1}^{|V|-1}(T_i-1))^2}{(|V|-1)\sum_{i=1}^{|V|-1}(T_i-1)(T_i-2)}.$$

We choose u as δ percentage quantile thresholds, i.e., δ percentage of $\{X_i, 1 \leqslant i \leqslant n\}$ falls below u,

$$k_\delta = \min\left\{k : \sum_{i=1}^{n} \frac{\mathbb{I}\{X_i \leqslant X_k\}}{n} \geqslant \frac{\delta}{100}, 1 \leqslant k \leqslant n\right\}, \qquad u = X_{k_\delta}.$$

We plot θ_n vs δ for the Intervals Estimator in the following sections. The EI is usually selected as the value corresponding to the stability interval in this plot.

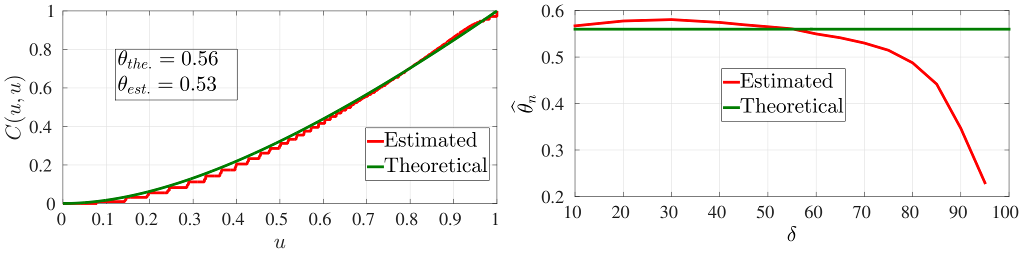
6.5.3 Synthetic graph

The simulations in the section follow the bivariate Pareto model and parameters introduced in (6.7). We use the same set of parameters as for Figure 6.1 and the graph is generated according to the Metropolis technique in Section 6.3.1.1.

For the SRW case, Figure 6.2a shows copula estimator, and theoretical copula based on the continuous distribution in (6.7), and is given by

$$C(u,u) = \left(1 + 2((1-u)^{-1/\gamma} - 1)\right)^{-\gamma} + 2u - 1.$$

Though we take quantized values for degree sequence, it is found that the copula estimated matches with theoretical copula. The value of EI is then obtained after cubic interpolation and numerical differentiation of copula estimator at point $(1,1)$. For the theoretical copula, EI is $1 - 1/2^\gamma$, where $\gamma = 1.2$. Figure 6.2b displays the comparison between the theoretical value of EI and Intervals estimate.



(a) Empirical and theoretical copulas (b) Intervals estimate and theoretical value

Figure 6.2: SRW sampling (synthetic graph)

For the RWJ algorithm, Figure 6.3 shows the Intervals estimate and theoretical value for different α. We used the expression (6.15) for theoretical calculation. The small difference in theory and simulation results is due to the assumption

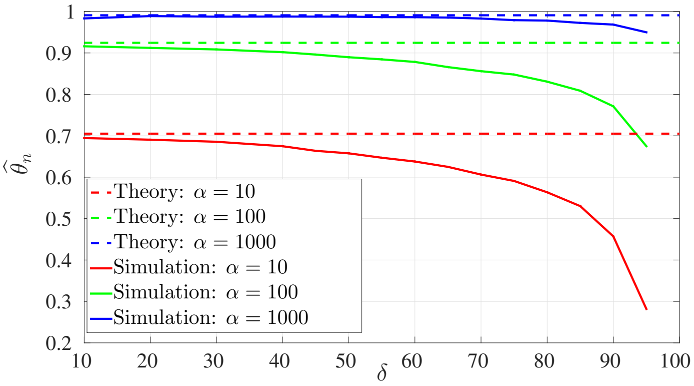


Figure 6.3: RWJ sampling (synthetic graph): Intervals estimate and theoretical value

of continuous degrees in the analysis, but the practical usage requires quantized version. Here $\alpha = 0$ case corresponds to SRW sampling.

Figure 6.4 displays the Intervals estimate of EI with PR sampling. It can be seen that the lower bound proposed in Proposition 6.10 gets tighter as c decreases. When $c = 1$, PR sampling becomes SRW sampling.

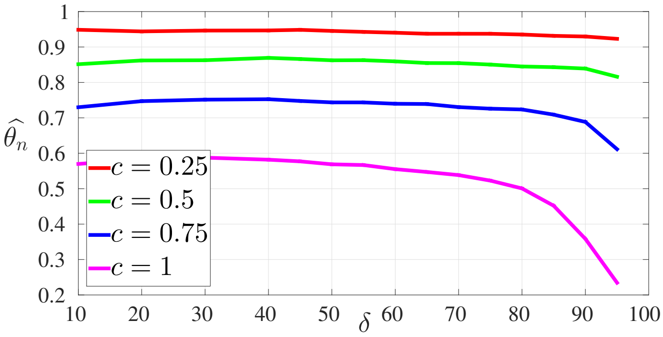


Figure 6.4: PR sampling (synthetic graph): Intervals estimate

6.5.3.1 Check of condition D''

The mixing conditions $D(u_n)$ and $D''(u_n)$ need to be satisfied for using the theory in Section 6.2. Though Intervals Estimator does not require them, these conditions will provide the representation by (6.4). Condition $D(u_n)$ works in this case as explained in previous sections and for $D''(u_n)$, we do the following empirical test. We collect samples for each of the techniques SRW, PR and RWJ with parameters given in respective figures. Intervals are taken of duration $5, 10, 15$ and 20 time samples. The ratio of number of upcrossings to number of exceedances r_{up} and ratio of number consecuitve exceedances to number of exceedances r_{cluster} are calculated in Table 6.1. These proportions are averaged over 2000 occurrences of each of these intervals and over all the different intervals. The statistics in the table indicates strong occurrence of Condition $D''(u_n)$. We have also observed that the changes in the parameters does not affect this inference.

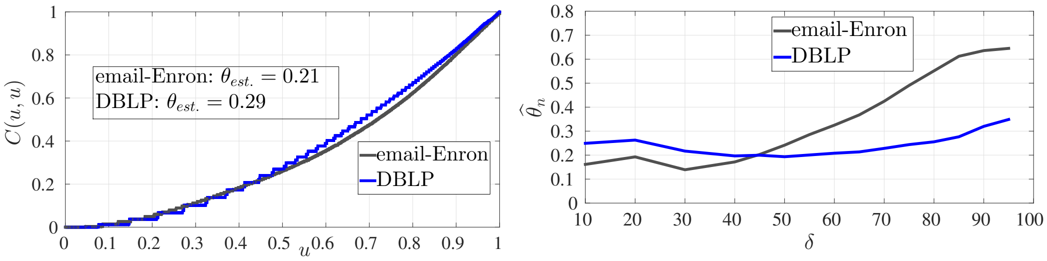
	$r_{\text{up}(\%)}$	$r_{\text{cluster}(\%)}$
SRW	4	89
PR	7	91
RWJ	5	86

Table 6.1: Test of Condition D'' in the synthetic graph

6.5.4 Real network

We consider two real world networks: Enron email network and DBLP network. The data is collected from [Leskovec & Krevl 2014]. Both the networks satisfy the check for Condition $D''(u_n)$ reasonably well.

For the SRW sampling, Figure 6.5a shows the empirical copula, and it also mentions corresponding EI. Intervals estimator is presented in Figure 6.5b. After observing plateaux in the plots, we took EI as 0.25 and 0.2 for DBLP and Enron email graphs, respectively.



(a) Empirical copulas

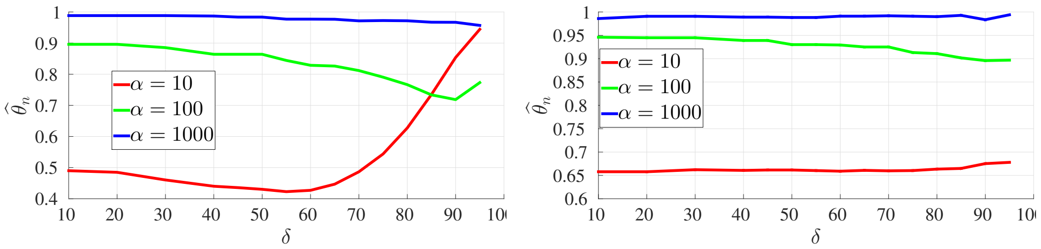
(b) Intervals estimate

Figure 6.5: SRW sampling (real networks)

In case of RWJ sampling, Figures 6.6a and 6.6b present Intervals estimator for email-Enron and DBLP graphs respectively.

6.6 Conclusions

In this work, we have associated extreme value theory of stationary sequences to sampling of large networks. We show that for any general stationary samples (function of node samples) meeting two mixing conditions, the knowledge of bivariate distribution or bivariate copula is sufficient to derive many of its extremal properties. The parameter extremal index (EI) encapsulates this relation. We relate EI to many relevant extremes in networks like order statistics, first hitting time,



(a) email-Enron graph

(b) DBLP graph

Figure 6.6: RWJ sampling (real networks)

mean cluster size, etc. In particular, we model dependence in degrees of adjacent nodes and examine random walk based degree sampling. Finally we have obtained estimates of EI for a synthetic graph with degree correlations and find a good match with the theory. We also calculate EI for two real-world networks.

Chapter 7

Conclusions and Future Research

We have considered the problem of sampling in networks. This thesis is aimed at developing distributed algorithms when the underlying graph is not known beforehand. The majority of the thesis focused on the design and performance analysis of sampling techniques based on random walks which turns out to be an efficient technique when only local information is available at each node (list of neighbors). We have also developed techniques for distributed computation of the eigen-spectrum of graph matrices.

First, we studied the distributed decomposition of the eigen-spectrum. The algorithms we introduced in this work produce a spectral-plot at each node and eigenvalues correspond to the frequencies of spectral peaks and respective eigenvector components are the amplitudes at those points. The algorithms are based on the idea of complex power iterations, a variation of power iterations. Several distributed approaches -diffusion algorithms, Monte Carlo techniques and random walk- are implemented. The proposed algorithms turned out to have a connection with quantum random walks. We have demonstrated the efficiency of the algorithms with simulations on many real-world networks.

We, then, dealt with a generalized unbiased sampling strategy for estimating aggregate edge functions ($\mu(G) = \sum_{(u,v)\in E} g(u,v)$). The edge functions can be easily extended to node and triangle functions. In this work we provided a partial crawling strategy using short dynamically adjustable random walk tours starting at a "virtual" super-node without invoking the notion of lumpability [Kemeny & Snell 1983]. A random walk tour is a random walk sample path that starts and ends at the same node on the graph. We used these tours to compute a frequentist unbiased estimator of $\mu(G)$ (including its variance) regardless of the number of nodes, $n > 0$, in the seed set (contracted into the super-node) unlike previous asymptotically unbiased methods. We also provide a Bayesian approximation of the posterior of $\mu(G)$ given the observed tours and it stands as a real-time assessment of estimation accuracy. In our experiments we noted that the posterior is remarkably accurate using a variety of networks, large and small. Furthermore, when the network is formed by randomly wiring connections while preserving degrees and attributes of the nodes in the observed network, we devised an estimation technique for the expected true value with partial knowledge of the original graph. It can also be noted that the proposed algorithm does not need to overcome the burn-in time barrier (mixing time) to start collecting samples.

Later, we extended the idea of super-node and elimination of the necessity of burn-in time barrier to average functions $\bar{\nu}(G) = |V|^{-1} \sum_{u \in V} g(u)$. We have developed an estimation technique based on reinforcement learning (RL) in this setting. We compared in a systematic manner some basic random walk based techniques like respondent driven sampling (RDS) and Metropolis Hastings sampling (MH). We demonstrate that with a good choice of cooling schedule, the performance of RL is comparable to that of RDS (which outperforms MH) but the trajectories of RL have less fluctuations than RDS.

Finally we analyzed some extremal events in the samples collected via random walks. We associated extreme value theory of stationary sequences to sampling of large complex networks and we studied the extremal and clustering properties of the sampling process due to dependencies. In order to facilitate a painless future study of correlations and clusters of samples in large networks, we proposed to abstract the extremal properties into a single and handy parameter called the extremal index (EI). For any general stationary samples meeting two mixing conditions, we find that knowledge of bivariate distribution or bivariate copula is sufficient to compute EI analytically and thereby deriving many extremal properties. Several useful applications of EI (first hitting time, order statistics and mean cluster size) to analyze large graphs, known only through sampled sequences, are proposed. Degree correlations are explained in detail using a random graph model with joint degree distribution between neighbor nodes. Three different basic random walk based algorithms that are widely discussed in literature are then revised for degree state space and EI is calculated when the joint degree distribution is bivariate Pareto. We established a general lower bound for EI in PageRank processes irrespective of the degree correlation model. Finally using two estimation techniques, EI is numerically computed for a synthetic graph with neighbour degrees correlated and for two real networks (Enron email network and DBLP network).

7.1 Future Research

The following problems can be further investigated.

Studies on super-node

Formation of super-node poses many questions. The best way to select a super-node is not yet known. It is also useful to explore how the choice of super-node affects the asymptotic variance. Another direction is to relax the global knowledge of disconnected components for the formation of super-node. The estimation procedure in Chapter 4 tackles the disconnected component issue, but requires at least one node to be known *a priori* from each of the components. How to perform this action with minimal cost?

Theoretical study of ratio with tours estimator

We provided several numerical comparisons in Chapter 4 for R-T estimator. Theoretical study of the this estimator is a possible work for future. Such a study may include finding the rate at which the difference $B - \sum_{k=1}^{m(B)} \xi_k$ in (4.10) goes to zero. The effect of dynamic super-node in the R-T estimator is also not explored in this thesis.

Concentration result for the regeneration based estimator

For the regeneration or tour based estimator introduced in Chapter 4, many extensions can be considered. One possible direction is in the lines of concentration result for ergodic theorem, Theorem 2.5. Such a result will provide the time and memory complexities for this estimator.

More theoretical results of random walk with jumps

We have used the random walk with jumps (RWJ) introduced in [Avrachenkov et al. 2010] in many places in this thesis. Though via simulations, it performs better than several other random walk estimators, theoretical study has not been done in proving its superior performance in terms of asymptotic variance and/or mixing time for general graphs. This could be a possible direction for future study.

Relation between extremal index and global graph coefficients

Since extremal index captures the correlation structure in the graph, it is useful to investigate its relation with the clustering coefficient, as well as with the assortativity coefficient.

Usefulness of extremal index in analyzing random walk based algorithms

In [Avrachenkov et al. 2014a], the authors analyzed an algorithm to detect high degree entities using extreme value theory with independent sampling. In a similar context where the random walks are used and generates stationary samples, the tools developed with extremal index in Chapter 6 will be helpful for theoretical studies, for instance, to analyze the algorithm in [Avrachenkov et al. 2014b].

Extension of spectral decomposition algorithms

We have proposed several algorithms for spectral decomposition in Chapter 3 for symmetric matrices. How those algorithms can be extended to non-symmetric matrices? Some of our observations in this direction are given below:

In the case of non-symmetric matrices, some of the eigenvalues will be complex with the exception of the main eigenvalue which will be still real. The right and left eigenvectors will be different, $\boldsymbol{A} = \sum_j \lambda_j \boldsymbol{u}_j \boldsymbol{v}_j^\intercal$, and consequently,

$$\begin{aligned}\int_{-\infty}^{\infty} e^{i\boldsymbol{A}t} e^{-vt^2/2-it\theta} dt &= 2\pi \sum_j \boldsymbol{u}_j \boldsymbol{v}_j^\intercal \exp\left(-\frac{(\Re(\lambda_j)-\theta)^2}{2v}\right) \times \exp\left(\frac{(\Im(\lambda_j)^2}{2v}\right) \\ &\quad \times \cos\left(\Im(\lambda_j(\Re(\lambda_j)-\theta))\right).\end{aligned}$$

This implies that the peaks on the eigenvalues have a value $\exp\left(\frac{(\Im(\lambda_j)^2}{2v}\right)$, which may be extremely large and too discrepant between eigenvalues. Nevertheless the complex diffusion and quantum random walks could be applied if the eigenvalues with non zero imaginary part are clearly separated from the pure real eigenvalues. This is the case when using the non-backtracking matrix for spectral clustering of [Bordenave et al. 2015] where it is proved to be more efficient than the classical spectral clustering counterpart, provided the graph satisfies certain properties. The above observations need to be verified with simulations on real data.

It will also be interesting to design an automatic procedure for the identification of dominant eigenvalues and eigenvectors

Random walks and spectral decomposition

As we saw in Sections 3.4.3 and 3.5.1 of the algorithms for distributed spectral decomposition, the Monte Carlo technique based on random walks surprisingly requires very few iterations to converge. In many cases it is even one iteration. This interesting behavior is a possible direction to explore later.

Study of asymptotic variance in random walk based sampling

There exist many works on mixing time of random walks relevant to graph sampling like $t_{\text{mix}} = \mathcal{O}(\log n)$ in preferential attachments [Mihail et al. 2003], but not many studies published on asymptotic variance of the random walks, especially in random graphs. We argued in Section 2.3 that asymptotic variance is a better choice of performance comparison of random walks for estimation in graphs. This can be included as a part the future work of this thesis.

Random walks in directed graphs

The extension of random walk theory to directed graphs requires the graph to be strongly connected. This is a strong assumption and many practical networks like Twitter do not satisfy this. Moreover, as far as we know, there is no easy closed expression for the stationary distribution on directed graphs and this makes unbiasing the random walk a difficult task. Some works consider the directed graph as undirected and use the existing RW techniques. But this fails to estimate some specific properties of directed graphs. PageRank, of course, avoids this problem by introducing a random jump with certain probability at each step, but at the cost of uniformly sampling the network.

7.1.0.1 Comparison with state of the art spectral decomposition algorithms

It would be interesting to see how the algorithms proposed in Chapter 3 perform in comparison with many of the recent spectral decomposition algorithms, for instance the Arnoldi methods in [Frahm & Shepelyansky 2010].

Appendix A

Présentation des Travaux de Thèse en Francais

A.1 Introduction

Ces dernières années ont connu un essor dans la science du traitement de données massives. Les grands ensembles de données, souvent appelés *Big data* dans l'industrie, ont déclenché le développement de nombreux logiciels et outils mathématiques. Cela a également aidé un domaine interdisciplinaire étroitement lié appelé «la Science des Réseaux», qui étudie les réseaux en général. En particulier, une classe spécifique appelée «Réseaux Complexes» est populaire dans la Science des Réseaux. Cette dernière est utilisée pour décrire, de manière informelle, les réseaux présentant les caractéristiques suivantes:

- Grande taille
- Topology creuse
- Petite distance moyenne (également appelée phénomène du petit monde)
- De nombreux triangles
- Distribution de degré à queue lourde (également appelé phénomène sans échelle)

L'étude des grands réseaux fait face à de nombreux problèmes: la collection de données à partir des réseaux sous-jacents prend du temps et d'énormes ressources. Même si l'ensemble du graphe est collecté, le traitement centralisé avec de nombreux algorithmes de matrice a de grandes exigences de mémoire et entraîne de longs délais pour observer des résultats utiles. Par conséquent, les chercheurs recourent à des algorithmes distribués. De plus, comme de nombreux algorithmes de réseau nécessitent un temps exponentiel pour terminer, les algorithmes randomisés et approximatif semblent prometteurs dans la science des réseaux.

Cette thèse traite des moyens efficaces pour recueillir pour échantillons représentatifs d'un grand réseau en utilisant des techniques probabilistes et fait une inférence statistique sur les propriétés du réseau avec ces échantillons. L'accent est mis sur les stratégies distribuées.

Les sections qui suivent fournissent une courte introduction aux problèmes abordés dans cette thèse.

A.1.1 Échantillonnage et estimation dans les réseaux

Considérons un grand réseau impossible à observer complètement, c'est-à-dire que le réseau est supposé inconnu, dans le sens que les matrices de graphe comme adjacence ou laplacien ne sont pas complètement connues à l'avance.

Comment répondre, au moins approximativement, aux questions sur les propriétés globales des réseaux? Par exemple: Quelle proportion parmi la population dans une ville soutient un certain parti politique? Quel est l'âge moyen des utilisateurs des réseaux sociaux en ligne comme Friendster, Myspace ou Facebook? Quelle est la fraction des connexions homme-femme par rapport à celle des connexions femme-femme dans un certain réseau social en ligne? Est-ce que l'OSN est assortative ou disassortative?

A.1.1.1 Contraintes d'information locale

Pour collecter des informations à partir d'un OSN, l'échantillonneur émet une requête Interface de programmation d'application (API) pour un utilisateur particulier qui renvoie son voisinage de saut et le contenu publié par l'utilisateur. Bien que certains tiers puissent obtenir la base de données complète des OSN (par exemple Twitter) avec des dépenses supplémentaires, nous nous concentrons ici sur le cas typique où le tiers peut obtenir des informations seulement sur les voisins d'un utilisateur particulier à travers des requêtes API.

Le problème qui nous intéresse peut être formulé comme suit. Soit $G = (V, E)$ un réseau non dirigé, où V est l'ensemble des sommets et $E \subseteq V \times V$ est l'ensemble des arêtes. Contrairement à la définition habituelle de E où chaque bord est une paire non ordonnée et ne se présente qu'une fois, pour simplifier notre notation on considère que si $(u, v) \in E$ then $(v, u) \in E$. Les bords et les nœuds peuvent avoir des étiquettes. $\{\lambda_k^{\boldsymbol{A}}\}_{k=1}^{|V|}$ et $\{\lambda_k^{\boldsymbol{L}}\}_{k=1}^{|V|}$ sont les valeurs propres des matrices adjacentes et laplaciennes et $\{\boldsymbol{u}_k^{\boldsymbol{A}}\}$ and $\{\boldsymbol{u}_k^{\boldsymbol{L}}\}$ soient leurs propres vecteurs.

Formulation du problème pour l'échantillonnage en réseau

- Estimer
$$\nu(G) = \sum_{u \in V} g(u), \quad \bar{\nu}(G) = \frac{1}{|V|} \sum_{u \in V} g(u), \tag{A.1}$$
où $g(u)$ est une fonction bornée du noeud u. Parfois nous nous concentrons également sur le problème d'estimation associé sur les arêtes, avec $g(u, v)$ désignant une fonction sur le bord (u, v):
$$\mu(G) = \sum_{(u,v) \in E} g(u, v), \quad \bar{\mu}(G) = \frac{1}{|E|} \sum_{(u,v) \in E} g(u, v). \tag{A.2}$$
- Le graphe G est inconnu.
- Seulement les informations locales sont connues: nous avons des informations uniquement sur les nœuds initiaux et leurs ID voisins, nous ne pouvons que

consulter ou visiter un voisin d'un noeud; Alors, nous avons également des informations sur les nœuds visités et leurs voisins.

Une solution possible, sous la contrainte que seulement l'information locale est disponible à chaque nœud est **échantillonnage en boule de neige**: ici, après avoir sélectionné un noeud initial, l'échantillonneur sonde tous ses voisins, puis pour chacun des voisins, le processus de sondage se répète. Le processus se poursuit ainsi. Un inconvénient principal de cette procédure est que la nombre de nœud échantillonné augmente exponentiellement et couvrira bientôt tout le réseau. Un autre problème est que dans le cas d'un très grand réseau, cet échantillonnage est asymptotiquement biaisé vers le vecteur propre principal de la matrice d'adjacence (appelée centralité du vecteur propre) [Newman 2010]. Un tel biais est difficile à compenser car la connaissance de la centralité du vecteur propre nécessite l'ensemble du réseau.

Une **marche aléatoire simple** sur un graphe sous la même contrainte locale fournit une solution viable. Dans une marche aléatoire simple, après avoir choisi un noeud initial de façon aléatoire ou déterministe, la marche aléatoire choisit un des voisins du noeud présent uniformément au hasard et se déplace vers le noeud sélectionné, et ce processus se poursuit. Asymptotiquement, le choix de l'échantillonneur de marche aléatoire est biaisé vers de grands degrés et comme un tel biais inclut seulement des informations locales, il peut être facilement inversé. Nous discuterons plus à propos de l'échantillonnage aléatoire dans cette thèse

La période de combustion (ou temps de mélange) des marches aléatoires est la période de temps après laquelle la marche aléatoire sera approximativement indépendante de la distribution initiale et produira des échantillons presque stationnaires. Dans la plupart des procédures d'estimation basées sur la marche aléatoire, les échantillons jusqu'à la période de brûlage sont jetés afin de fournir des garanties théoriques. Ceci pose de sérieuses limitations, en particulier avec des contraintes sévères sur le nombre d'échantillons imposés par les taux d'interrogation de l'API. En outre, si nous limitons le nombre d'échantillons autorisés, on ne sait pas à quel point la valeur estimée est précise. Une évaluation en temps réel de l'estimateur avec la distribution postérieure bayésienne sera utile dans une telle situation. Nous abordons ces problèmes dans l'échantillonnage dans cette thèse.

A.1.2 Décomposition spectrale: "échantillonnage dans le domaine spectral"

Les propriétés spectrales d'un graphe ou d'un réseau sont intéressantes pour des domaines divers en raison de sa forte influence dans de nombreux algorithmes pratiques. Comme expliqué plus tard dans cette section, de nombreuses propriétés des réseaux sont abstraites de manière concise dans quelques valeurs propres dominantes des matrices associées aux réseaux. Cependant, la complexité de calcul associée à l'estimation du spectre des valeurs propres et des vecteurs propres a été un problème exigeant pendant une longue période. Ainsi, dans le contexte de la science des réseaux, la conception des méthodes de décomposition spectrale distribuées est particulièrement importante.

Problème

Nous étudions des algorithmes efficaces pour la décomposition spectrale afin de trouver k valeurs propres dominantes (k les plus petites ou plus grandes) et ses vecteurs propres à haute résolution. Les algorithmes doivent permettre une implémentation distribuée dans le sens où chaque nœud effectue lui-même le traitement avec les données de son voisinage de houblon. En particulier, nous limitons notre attention aux matrices qui sont compatibles avec le graphe, c'est-à-dire que la composante de la matrice à (i, j) est 0 lorsqu'il n'y a pas de bord (i, j) dans le graphe. Cela facilite le développement des techniques distribuées.

Pour motiver le calcul du spectre propre des matrices de graphe, nous présentons maintenant quelques-unes de ses applications.

A.1.2.1 Pertinence de la décomposition spectrale

La littérature existante explore de nombreuses applications du spectre des matrices graphes. En particulier, les valeurs propres et les vecteurs propres peuvent être utilisés globalement ou localement. Les applications globales nécessitent une unité centrale pour collecter des valeurs propres et des composantes de vecteurs propres de tous les noeuds, puis transmettre ces informations globales aux algorithmes derrière l'application. Mais dans le cas d'applications locales, les algorithmes sous-jacents fonctionnent séparément à chaque nœud en utilisant la composante respective dans les vecteurs propres dominants, ainsi que la connaissance des valeurs propres.

Voici quelques applications des matrices adjacentes et laplaciennes liées aux graphes et aux réseaux. Par exemple:

- Le nombre d'arêtes $|E| = 1/2 \sum_{i=1}^{|V|} (\lambda_i^{\boldsymbol{A}})^2$.
- Le nombre total de triangles en G est donné par

$$T = \frac{1}{6} \sum_{i=1}^{|V|} (\lambda_i^{\boldsymbol{A}})^3. \tag{A.3}$$

- Le nombre total de marches de longueur k peut être approché par $a_1^2 (\lambda_1^{\boldsymbol{A}})^k$, lorsque G est connexe et non bipartite .
- Le nombre de triangles auxquels un nœud m a participé est $1/2 \sum_{i=1}^{|V|} \lambda_i^3(\boldsymbol{A}) \, (\boldsymbol{u_i^A}(m))^2$. Par conséquent, si l'on calcule localement les valeurs propres et les vecteurs propres top-k localement au noeud m, on peut approximer avec une bonne exactitude combien ses voisins sont connectés.

- Réduction dimensionnelle: Après le calcul des vecteurs propres top-k, chaque noeud est mappé à un point dans $\mathbb{R}^k$ espace avec les composantes du vecteur propre et la proximité dans le nouvel espace implique une affinité en termes de position dans le réseau. Par conséquent, de nouveaux liens peuvent être suggérés parmi les nœuds non connectés lorsque la distance entre eux dans $\mathbb{R}^k$ espace est petite [Kempe & McSherry 2008].

- Groupement spectral: Le problème de la recherche de grappes dans un réseau (en particulier dans les réseaux sociaux) est un problème ancien avec de nombreux développements ces dernières années. Parmi les techniques étudiées, le regroupement spectral est une solution éminente [Von Luxburg 2007]. Le principal goulet d'étranglement dans le regroupement spectral est le calcul des vecteurs propres dominants, que nous essayons d'accomplir ici avec beaucoup moins de complexité.

- Nombre d'arbres d'étalement: Le nombre d'arbres d'étalement de G est donné par une expression bien connue contenant les valeurs propres de la matrice laplacienne,
$$\frac{\lambda_1^{\boldsymbol{L}}\lambda_2^{\boldsymbol{L}}\ldots\lambda_{|V|-1}^{\boldsymbol{L}}}{|V|}.$$

Les détails sur les manières de calculer les valeurs propres et les vecteurs propres sont expliqués dans la thèse.

A.1.3 Théorie de la valeur extrême et ses applications

Après la collection d'échantillons aléatoires du réseau, une question se pose sur la façon de faire plus d'inférences avec eux (autres que l'estimation). À cette fin, notez que de nombreuses entités de réseau social sont corrélées, par exemple, si nous prenons un réseau de co-auteurs où les nœuds sont les auteurs et un lien est établi lorsque deux auteurs rédigent un document de recherche ensemble, les chercheurs ont tendance à établir des liens plus étroits entre eux. Comment extraire des informations sur la structure de corrélation du réseau avec quelques échantillons aléatoires? Dans de tels scénarios, les outils mathématiques de différents domaines comme la théorie des valeurs extrêmes (EVT) semblent être très utiles. Dans cette thèse, nous étudions également la relation entre la théorie des valeurs extrêmes et l'échantillonnage en réseau. En supposant qu'une séquence stationnaire d'échantillons aléatoires d'un réseau est disponible, nous étudions des propriétés extrêmes telles que la première fois pour frapper un nœud à grand degré, les grappes explorées pendant le processus d'échantillonnage, etc.

La théorie de la valeur extrême, en général, est l'étude des événements rares. En particulier, il traite de la convergence de $M_n := \max\{X_1, \ldots, X_n\}$ des échantillons aléatoires $\{X_i\}_{i \leqslant n}$. Dans la sous-section suivante, nous faisons une brève introduction à EVT.

Puisque nous nous intéressons aux randonnées aléatoires dans cette thèse, nous nous concentrons sur la théorie des valeurs extrêmes pour les séquences stationnaires

$\{X_i\}_{i\leqslant n}$. Ici, les techniques classiques étudient le M_n maximum de $\{X_i\}_{i\leqslant n}$ en utilisant le maximum $\widetilde{M_n}$ des échantillons iid associé $\{\widetilde{X_i}\}_{i\leqslant n}$. En particulier, dans des conditions de mixage correctes, $c_n^{-1}(\widetilde{M_n} - d_n) \to H$ et $c_n^{-1}(M_n - d_n) \to G$ sont liés comme $G = H^\theta$, où θ est appelé indice extrême. Il s'avère que l'indice extrême est lié à plusieurs événements extrêmes intéressants dans l'échantillonnage. Nous allons explorer l'index extrême et ses applications en détail dans cette thèse.

A.2 Contribution de la thèse

Nous faisons les contributions suivantes sur l'estimation de réseau marche aléatoire et sur la décomposition spectrale distribuée.

A.2.1 Méthodes distribuées d'échantillonnage dans le domaine spectral

Dans ce travail, nous abordons le problème de trouver des valeurs propres supérieures (k) dominantes (les plus petites ou les plus importantes) et les vecteurs propres correspondants de matrices de graphes symétriques dans des réseaux de manière distribué. Nous proposons une nouvelle idée appelée «complex Power iterations» pour décomposer les valeurs propres et les vecteurs propres au niveau du noeud, analogues à l'analyse temps-fréquence dans le traitement du signal. A chaque noeud, les valeurs propres correspondent aux fréquences des pics spectrales et les composantes des vecteurs propres respectifs sont les amplitudes à ces points. Basés sur des itérations de puissance complexes et motivés par des processus de diffusion fluide en réseaux, nous concevons des algorithmes distribués avec différents ordres d'approximation. Nous introduisons également une technique de Monte Carlo avec des bavardages qui réduit considérablement la surcharge de calcul. Un algorithme parallèle équivalent de marche aléatoire est également présenté. Nous validons les algorithmes avec des simulations sur des réseaux réels. Notre formulation de la décomposition spectrale peut être facilement adaptée à un algorithme simple basé sur des randonnées quantiques aléatoires. Avec l'avènement du calcul quantique, l'algorithme quantique proposé sera extrêmement utile.

Nous étendons ensuite les techniques distribuées susmentionnées pour détecter, avec une résolution plus élevée, des valeurs propres étroitement situées et des vecteurs propres correspondants de matrices de graphes symétriques. Nous modélisons le système de calcul spectral de graphes sous la forme de systèmes physiques à dynamique lagrangienne et hamiltonienne. Le spectre de la matrice laplacienne, en particulier, est encadré comme un système de ressort-masse classique avec une dynamique lagrangienne. Le spectre de toute matrice graphe symétrique générale se révèle avoir une connexion simple avec des systèmes quantiques et il peut donc être formulé comme solution à une équation différentielle de type Schrödinger. En tenant compte de l'exigence de résolution plus élevée dans le calcul du spectre et Les problèmes de stabilité liés à la solution numérique de l'équation différentielle sous-jacente, nous proposons l'application d'intégrateurs symplectiques au calcul de l'eigenspectrum.L'efficacité des techniques proposées est démontrée par des simulations numériques sur des réseaux réels de tailles et complexités différentes.

La publication relative à cette contribution est la suivante:

- [Avrachenkov et al. 2016b] Konstantin Avrachenkov, Philippe Jacquet and Jithin K. Sreedharan. *Distributed Spectral Decomposition in Networks by Complex Diffusion and Quantum Random Walk.* In Proc. IEEE International Conference on Computer Communications (INFOCOM), April 2016.

Travaux connexes : Nous fournissons une interprétation directe du spectre des valeurs propres et des vecteurs propres en termes de pics dans le domaine fréquentiel de l'exponentielle complexe de $\boldsymbol{A}$ et l'exploitons pour développer des algorithmes distribués. Au mieux de nos connaissances, le premier algorithme de décomposition spectrale du réseau distribué a été proposé dans [Kempe & McSherry 2008]. La partie la plus difficile de l'algorithme dans [Kempe & McSherry 2008] est l'orthonormalization distribuée à chaque étape de l'algorithme. C'est une opération difficile, que les auteurs résolvent en communiquant des informations par des randonnées aléatoires. De toute évidence, si le graphe a une faible conductance (un cas typique pour de nombreux grands graphes), cette opération prendra extrêmement longtemps à chaque étape de l'algorithme. Notre premier algorithme distribué basé sur la diffusion de fluide complexe à travers le réseau, une implémentation de *itérations d'énergie complexes*, ne nécessitent pas d'orthonormalization. Les auteurs utilisent des techniques de traitement du signal qui sont dans le même esprit de notre approche. Cependant, leur approche nécessite l'utilisation du voisinage à deux bonds ou quatre bonds pour chaque itération, alors que nos algorithmes fonctionnent avec un pas de temps et un voisinage à un seul bond. L'approche de [Sahai et al. 2012, Franceschelli et al. 2013] déforme les valeurs des vecteurs propres et des valeurs propres et la correction nécessaire n'est pas évidente. De plus, comme les méthodes de [Sahai et al. 2012, Franceschelli et al. 2013] sont basées sur des transformées de Fourier classiques, les valeurs propres peuvent ne pas être détectées à cause de pics parasites dans le spectre. Notre approche permet de surmonter ce problème en utilisant le lissage gaussien. Nos algorithmes peuvent également être mis en œuvre immédiatement à l'aide de légers bavardages et de randonnées aléatoires avec des récompenses complexes. Un algorithme de bavardage basé sur l'apprentissage par renforcement a récemment été introduit dans [Borkar et al. 2014], mais il ne calcule que le vecteur propre principal. A partir de l'analyse de notre technique de diffusion, nous observons que nos algorithmes sont évolutifs de l'ordre du degré maximum. Enfin, notre méthode a une relation très intéressante avec les randonnées quantiques aléatoires qui, avec l'avancement du calcul quantique, peuvent rendre nos approches très efficaces.

A.2.2 Échantillonnage en réseau avec des techniques de marche aléatoire

Échantillonnage non asymptotiquement impartial et inférence bayésienne

Les utilisateurs d'OSN-A sont-ils plus susceptibles de former des amitiés avec ceux qui ont des attributs similaires? Les utilisateurs d'OSN-B ont-ils un contenu X plus favorable qu'un autre contenu Y? Ces questions se posent fréquemment dans le contexte de Social Network Analysis (SNA), mais souvent l'exploration d'un réseau OSN via son API est le seul moyen de recueillir des données pour un tiers. À ce jour, la majorité des ensembles de données publics sont constitués d'analyses partielles de l'API et par conséquent manquent de garanties statistiques, ce qui limite considérablement les progrès de la recherche dans le SNA. En utilisant les propriétés régénératives des marches aléatoires, nous proposons des techniques d'estimation basées sur des traits courts qui ont des garanties statistiques éprouvées: non-asymptotique sans biais et évasion de «burn-in». De plus, nos tracés courts peuvent être implémentés dans des algorithmes massivement distribués. Nous fournissons également une chenille adaptative qui rend notre méthode sans paramètre, améliorant considérablement nos garanties statistiques. Nous dérivons alors une approximation de la postérieure bayésienne des estimations. De plus, nous obtenons un estimateur de la valeur attendue des statistiques de nœuds et de contours dans un modèle de configuration équivalent ou un modèle de graphe aléatoire de Chung-Lu du réseau donné (où les nœuds sont connectés aléatoirement) et l'utilisent comme base pour tester des hypothèses nulles. Les résultats théoriques sont supportés avec des simulations sur une variété de réseaux réels.

La publication relative à cette contribution est la suivante:

- [Avrachenkov et al. 2016c] Konstantin Avrachenkov, Bruno Ribeiro and Jithin K. Sreedharan. *Inference in OSNs via Lightweight Partial Crawls.* ACM SIGMETRICS Performance Evaluation Review, vol. 44, no. 1, pages 165 - 177, June 2016.

Travaux connexes : Les ouvrages de [Massoulié et al. 2006] et [Cooper et al. 2013] sont les plus proches de la nôtre. Les auteurs en [Massoulié et al. 2006] estime la taille d'un réseau en fonction des temps de retour des randonnées aléatoires. Le papier [Cooper et al. 2013] estime le nombre de triangles, la taille du réseau et le nombre de sous-graphes à partir de randonnées aléatoires pondérées en utilisant les résultats de [Aldous & Fill 2002, Chapter 2 and 3]. Les travaux antérieurs sur l'inférence par échantillons finis de statistiques de réseau à partir d'analyses de réseau incomplètes [Goel & Salganik 2009, Koskinen et al. 2010, Koskinen et al. 2013, Handcock & Gile 2010, Heckathorn 1997, Ligo et al. 2014, Thompson 2006] doivent adapter les données partielles observées à un modèle de graphe probabiliste tel que les ERGM (famille exponentielle de modèles de graphes aléatoires). Notre travail avance technique de pointe en estimant les statistiques de réseau à partir d'analyses partielles parce que: (A) nous estimons des statistiques des sommes de fonctions arbitraires sans hypothèses sur le modèle de graphe ou le graphe sous-jacent; (B) nous n'avons pas besoin de polariser la marche aléatoire avec des pondération comme dans Cooper et al.; ceci est particulièrement utile lors de l'estimation de statistiques multiples en réutilisant les mêmes observations; (C) on déduit des limites supérieure et inférieure sur la variance de l'estimateur, qui montrent toutes deux la connexion avec l'intervalle spectral; Et enfin (d) nous calculons a posteriori sur nos estimations pour donner aux praticiens un moyen d'accéder à la confiance dans les estimations sans s'appuyer sur des quantités inaccessibles comme «spectral gap» et sans prendre pour hypothèse un modèle de graphe probabiliste.

Dans notre travail, nous proposons une stratégie d'exploration partielle à l'aide de courtes randonnées aléatoires réglables de façon dynamique, à partir d'un supernœud «virtuel» sans invoquer la notion de «lumpability» [Kemeny & Snell 1983]. Une marche aléatoire est un moyen d'échantillonnage aléatoire qui commence et se termine au même noeud sur le graphe. Nous utilisons ces tours pour calculer un estimateur sans biais fréquencé de $\mu(G)$ (y compris sa variance) de manière independente du nombre de nœuds, $n > 0$, dans le ensemble initial et de manière independente de la valeur de $m > 0$, Contrairement aux méthodes précédentes asymptotiquement impartiales [Avrachenkov et al. 2010, Gjoka et al. 2011, Lee et al. 2012, Ribeiro & Towsley 2010, Ribeiro et al. 2012]. Nous fournissons également une approximation bayésienne de la postérieure de $\mu(G)$ $\mathbb{P}[\mu(G)|\mathcal{D}_m(I_n)]$, Qui se montre cohérente. Dans nos expériences nous notons que le postérieur est remarquablement précis en utilisant une variété de réseaux grands et petits. En outre, lorsque le réseau est formé par des connexions au câblage aléatoire tout en préservant les degrés et les attributs des noeuds dans le réseau observé, nous concevons une technique d'estimation de la valeur vraie attendue avec une connaissance partielle du graphe original.

Apprentissage par renforcement

L'apprentissage par renforcement (RL) permet d'approcher l'estimation de la moyenne des fonctions du réseau. Cette approche est également basée sur les propriétés régénératives des marches aléatoires et évite ainsi la barrière du temps de combustion. La technique RL est essentiellement une approximation stochastique formée à partir de l'équation de Poisson d'un processus associé semi-markovien. La performance de cette technique dépend des étapes associées à l'algorithme d'apprentissage. Cette technique peut être placée comme une technique intermédiaire entre l'itération pure Monte Carlo (MCMC) de la chaîne de Markov (stochastique) et l'itération de la valeur relative (déterministe). Les tailles de pas contrôlent la stabilité de la technique RL et ses trajectoires sont beaucoup plus stables que celles des procédures d'estimation basées sur la marche aléatoire standard. Ses performances sont également comparables à celles de l'échantillonnage piloté par les répondants qui a une faible variance asymptotique que bien d'autres estimateurs.

La publication relative à cette contribution est la suivante:

- [Avrachenkov et al. 2016a] Konstantin Avrachenkov, Vivek S. Borkar, Arun Kadavankandy and and Jithin K. Sreedharan. *Comparison of Random Walk based Techniques for Estimating Network Averages."* International Conference on Computational Social Networks (CSoNet), August 2016.

A.2.3 Théorie des valeurs extrêmes et processus d'échantillonnage en réseau

Ce travail explore la structure de dépendance dans la séquence échantillonnée d 'un réseau inconnu. Nous considérons des algorithmes aléatoires pour échantillonner les noeuds et étudier les propriétés extrêmes dans toute séquence stationnaire associée de caractéristiques d'intérêt telles que les degrés de noeud, le nombre d'adeptes ou le revenu des noeuds dans les réseaux sociaux en ligne, etc., qui satisfont à deux conditions de mélange. Plusieurs extrêmes utiles de la séquence échantillonnée comme k la plus grande valeur, des groupes de dépassements sur un seuil, un premier temps de frappe d'une grande valeur, etc., sont étudiés. Nous résumons la dépendance et les statistiques des extrêmes en un seul paramètre qui apparaît dans la théorie des valeurs extrêmes, appelée indice extrême (EI). Dans ce travail, nous dérivons ce paramètre de manière analytique et nous l'estimons empiriquement. Nous proposons l'utilisation de l'IE comme paramètre pour comparer différentes procédures d'échantillonnage. Comme exemple spécifique, les corrélations de degré entre les noeuds voisins sont étudiées en détail avec trois randonnées aléatoires proéminentes comme techniques d'échantillonnage.

La publication relative à cette contribution est la suivante:
Konstantin Avrachenkov, Natalia M. Markovich and Jithin K. Sreedharan. *Distribution and Dependence of Extremes in Network Sampling Processes*

- [Avrachenkov et al. 2015b] Computational Social Networks, Springer, 2015.
- Third International IEEE Workshop on Complex Networks and their Applications, November 2014.

A.3 Conclusions

Nous avons examiné le problème de l'échantillonnage dans les réseaux. Cette thèse vise à développer des algorithmes distribués lorsque le graphe sous-jacent n'est pas connu auparavant. La majorité de la thèse traite la conception et l'analyse des performances des techniques d'échantillonnage basées sur des randonnées aléatoires, ce qui s'avère être une technique efficace lorsque seule l'information locale est disponible à chaque noeud (liste des voisins). Nous avons également développé des techniques de calcul distribué pour estimer le spectre propre de matrices de graphe.

Tout d'abord, nous avons étudié la décomposition distribuée du spectre propre. Les algorithmes que nous avons introduits dans ce travail produisent un graphique spectrale à chaque noeud et les valeurs propres correspondent aux fréquences des pics spectraux et les composantes des vecteurs propres sont les amplitudes à ces points. Plusieurs approches distribuées - algorithmes de diffusion, techniques Monte Carlo et marche aléatoire - sont mises en œuvre. Les algorithmes proposés se sont avérés avoir une connexion avec des randonnées quantiques aléatoires.

Nous avons donc étudié une stratégie d'échantillonnage impartiale généralisée pour estimer la somme fonctions de noeuds ($\mu(G) = \sum_{(u,v)\in E} g(u,v)$). Dans ce travail, nous avons fourni une stratégie d'exploration partielle à l'aide de courtes randonnées aléatoires réglables dynamiquement à partir d'un super-noeud «virtuel». Nous avons utilisé ces tours pour calculer un estimateur sans biais fréquentiste de $\mu(G)$ (y compris sa variance) indépendamment du nombre de nœuds, $n > 0$, dans le ensemble initial (contracté dans le super-nœud) contrairement au précédent Asymptotiquement sans biais méthodes. Nous fournissons également une approximation bayésienne de la postérieure de $\mu(G)$ et il se présente comme une évaluation en temps réel de l'exactitude de l'estimation. Dans nos expériences, nous avons noté que le postérieur est remarquablement précis en utilisant une variété de réseaux, petits et grands.

Ensuite, nous avons étendu l'idée de super-noeud et l'élimination de la nécessité de la barrière du temps de brûlure à des fonctions moyennes. Nous avons développé une technique d'estimation basée sur l'apprentissage par renforcement (RL) dans ce contexte. Nous avons comparé de façon systématique certaines techniques basiques de base basées sur la marche comme l'échantillonnage piloté par les répondants (RDS) et Metropolis Hastings (MH). Nous démontrons qu'avec un bon choix de programme de refroidissement, la performance de RL est comparable à celle de RDS (qui surperforme MH) mais les trajectoires de RL ont moins de fluctuations que RDS.

Enfin, nous avons analysé certains événements extrêmes dans les échantillons recueillis par des randonnées aléatoires. Nous avons associé la théorie des valeurs extrêmes des séquences stationnaires à l'échantillonnage de grands réseaux complexes et nous avons étudié les propriétés extrêmes et de clustering du processus d' échantillonnage en raison des dépendances. Nous avons proposé d'abstraire les propriétés extrêmes dans un seul paramètre pratique appelé indice extrême (EI). Pour tout échantillon stationnaire général satisfaisant à deux conditions de mélange, nous constatons que la connaissance de la copule bivariée est suffisante pour calculer EI analytiquement et ainsi dériver de nombreuses propriétés extrêmes. Nous proposons plusieurs applications utiles de l'IE (premier temps de frappe, statistiques d'ordre et taille moyenne des grappes) pour l'analyse de grands graphes, connus uniquement par séquences échantillonnées.

Bibliography

[Abounadi et al. 2001] Jinane Abounadi, Dimitri Bertsekas and Vivek S. Borkar. *Learning Algorithms for Markov Decision Processes with Average Cost.* SIAM Journal on Control and Optimization, vol. 40, no. 3, pages 681–698, 2001. ↑86

[Aldous & Fill 2002] David Aldous and James Allen Fill. *Reversible Markov Chains and Random Walks on Graphs.* Unfinished monograph, recompiled 2014, available at `http://www.stat.berkeley.edu/~aldous/RWG/book.html`, 2002. ↑17, ↑56, ↑80, ↑128

[Avrachenkov et al. 2010] Konstantin Avrachenkov, Bruno Ribeiro and Don Towsley. *Improving Random Walk Estimation Accuracy with Uniform Restarts.* In Proc. Workshop on Algorithms and Models for the Web-Graph (WAW). Stanford, CA, USA, 2010. ↑23, ↑55, ↑56, ↑65, ↑79, ↑100, ↑103, ↑117, ↑129

[Avrachenkov et al. 2014a] Konstantin Avrachenkov, Nelly Litvak, Liudmila Ostroumova Prokhorenkova and Eugenia Suyargulova. *Quick Detection of High-Degree Entities in Large Directed Networks.* In Proc. IEEE International Conference on Data Mining (ICDM), Shenzhen, China, December 2014. ↑109, ↑117

[Avrachenkov et al. 2014b] Konstantin Avrachenkov, Nelly Litvak, Marina Sokol and Don Towsley. *Quick Detection of Nodes with Large Degrees.* Internet Mathematics, vol. 10, no. 1-2, pages 1–19, 2014. ↑59, ↑108, ↑109, ↑117

[Avrachenkov et al. 2015a] Konstantin Avrachenkov, Arun Kadavankandy, Luidmilla Ostroumova Prokhorenkova and Andrei Raigorodskii. *PageRank in Undirected Random Graphs.* In Proc. Workshop on Algorithms and Models for the Web-Graph (WAW), Eindhoven, The Netherlands, 2015. ↑64

[Avrachenkov et al. 2015b] Konstantin Avrachenkov, Natalia M. Markovich and Jithin K. Sreedharan. *Distribution and Dependence of Extremes in Network Sampling Processes.* Computational Social Networks, vol. 2, no. 1, 2015. ↑12, ↑130

[Avrachenkov et al. 2016a] Konstantin Avrachenkov, Vivek S. Borkar, Arun Kadavankandy and Jithin K. Sreedharan. *Comparison of Random Walk Based Techniques for Estimating Network Averages.* In Proc. International Conference in Computational Social Networks (CSoNet), Lecture Notes in Computer Science, pages 27–38, Ho Chi Minh City, Vietnam, August 2016. ↑12, ↑129

[Avrachenkov et al. 2016b] Konstantin Avrachenkov, Philippe Jacquet and Jithin K. Sreedharan. *Distributed Spectral Decomposition in Networks by Complex Diffusion and Quantum Random Walk.* In Proc. IEEE International Conference on Computer Communications (INFOCOM), San Francisco, CA, USA, April 2016. ↑12, ↑127

[Avrachenkov et al. 2016c] Konstantin Avrachenkov, Bruno Ribeiro and Jithin K. Sreedharan. *Inference in OSNs via Lightweight Partial Crawls.* ACM SIGMETRICS Performance Evaluation Review, vol. 44, no. 1, pages 165–177, June 2016. ↑12, ↑128

[Barrat et al. 2008] Alain Barrat, Marc Barthelemy and Alessandro Vespignani. Dynamical Processes on Complex Networks. Cambridge University Press, 2008. ↑93, ↑98, ↑99

[Beirlant et al. 2006] Jan Beirlant, Yuri Goegebeur, Johan Segers and Jozef Teugels. Statistics of Extremes: Theory and Applications. John Wiley & Sons, 2006. ↑94, ↑107, ↑108, ↑109, ↑110

[Benevenuto et al. 2009] Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha and Virgílio Almeida. *Characterizing User Behavior in Online Social Networks.* In Proc. ACM SIGCOMM Internet Measurement Conference (IMC), Chicago, Illinois, USA, November 2009. ↑55

[Bentkus & Götze 1996] Vidmantas Bentkus and Friedrich Götze. *The Berry-Esseen Bound for Student's Statistic.* The Annals of Probability, vol. 24, no. 1, pages 491–503, 1996. ↑63

[Billingsley 2008] Patrick Billingsley. Probability and Measure. John Wiley & Sons, 3 edition, 2008. ↑21

[Blanes et al. 2006] Sergio Blanes, Fernando Casas and Ander Murua. *Symplectic Splitting Operator Methods for the Time-Dependent Schrödinger Equation.* The Journal of chemical physics, vol. 124, no. 23, 2006. ↑48

[Blanes et al. 2008] Sergio Blanes, Fernando Casas and Ander Murua. *Splitting and Composition Methods in the Numerical Integration of Differential Equations.* arXiv preprint arXiv:0812.0377, 2008. ↑48

[Boguna et al. 2003] M. Boguna, R. Pastor-Satorras and A. Vespignani. *Epidemic Spreading in Complex Networks with Degree Correlations.* Statistical Mechanics of Complex Networks. Lecture Notes in Physica v.625, pages 127–147, 2003. ↑98

[Boldi et al. 2011] Paolo Boldi, Marco Rosa, Massimo Santini and Sebastiano Vigna. *Layered Label Propagation: A MultiResolution Coordinate-Free Ordering for Compressing Social Networks.* In Sadagopan Srinivasan, Krithi

Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino and Ravi Kumar, editors, Proceedings of the 20th International Conference on World Wide Web, pages 587–596. ACM Press, 2011. ↑55

[Bordenave et al. 2015] Charles Bordenave, Marc Lelarge and Laurent Massoulié. *Non-Backtracking Spectrum of Random Graphs: Community Detection and Non-Regular Ramanujan Graphs.* In Proc. IEEE Annual Symposium on Foundations of Computer Science (FOCS), Berkeley, CA, USA, October 2015. ↑118

[Borkar et al. 2014] Vivek S. Borkar, Rahul Makhijani and Rajesh Sundaresan. *Asynchronous Gossip for Averaging and Spectral Ranking.* IEEE J. Sel. Areas Commun., vol. 8, no. 4, pages 703–716, 2014. ↑26, ↑127

[Borkar 2008] Vivek S. Borkar. Stochastic Approximation. Cambridge University Press, 2008. ↑86

[Brémaud 1999] Pierre Brémaud. Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues. Springer, 1999. ↑16, ↑22, ↑82, ↑83

[Brin & Page 1998] Sergey Brin and Lawrence Page. *The Anatomy of a Large-Scale Hypertextual Web Search Engine.* Computer networks and ISDN systems, vol. 30, no. 1, pages 107–117, 1998. ↑100, ↑102

[Chernick et al. 1991] Michael R. Chernick, Tailen Hsing and William P. McCormick. *Calculating the Extremal Index for a Class of Stationary Sequences.* Advances in Applied Probability, vol. 23, no. 4, pages 835–850, 1991. ↑96

[Chung & Lu 2003] Fan R. K. Chung and Linyuan Lu. *The Average Distance in a Random Graph with Given Expected Degrees.* Internet Mathematics, vol. 1, no. 1, pages 91–113, 2003. ↑64

[Chung 1996] Fan R. K. Chung. Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92). American Mathematical Society, 1996. ↑58

[Cooper et al. 2013] Colin Cooper, Tomasz Radzik and Yiannis Siantos. *Fast Low-Cost Estimation of Network Properties Using Random Walks.* In Proc. Workshop on Algorithms and Models for the Web-Graph (WAW), Cambridge, MA, USA, 2013. ↑56, ↑64, ↑128

[Cramér 1946] Harald Cramér. Mathematical Methods of Statistics. (PMS-9). Princeton University Press, 1946. ↑70

[Dogster & friendships network dataset KONECT 2015] Dogster and Catster friendships network dataset KONECT. `http://konect.uni-koblenz.de/networks/petster-carnivore`, May 2015. ↑71

[Easley & Kleinberg 2010] David Easley and Jon Kleinberg. Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press, 2010. ↑7

[Embrechts et al. 2011] Paul Embrechts, Claudia Klp̈pelberg and Thomas Mikosch. Modelling Extremal Events: For Insurance and Finance . Springer, corrected edition, 10 2011. ↑9

[Ferreira & Ferreira 2007] Ana Ferreira and Helena Ferreira. *Extremal Functions, Extremal Index and Markov Chains*. Technical report, Notas e comunicações CEAUL, 12 2007. ↑97, ↑110

[Ferro & Segers 2003] Christopher A. T Ferro and Johan Segers. *Inference for Clusters of Extreme Values*. Journal of the Royal Statistical Society, Series B,, vol. 65, pages 545–556, 2003. ↑94

[Frahm & Shepelyansky 2010] Klaus M Frahm and Dima L Shepelyansky. *Ulam Method for the Chirikov Standard Map*. The European Physical Journal B, vol. 76, no. 1, pages 57–68, 2010. ↑119

[Franceschelli et al. 2013] Mauro Franceschelli, Andrea Gasparri, Alessandro Giua and Carla Seatzu. *Decentralized Estimation of Laplacian Eigenvalues in Multi-Agent Systems*. Automatica, vol. 49, no. 4, pages 1031–1036, 2013. ↑26, ↑127

[Gabielkov et al. 2014] Maksym Gabielkov, Ashwin Rao and Arnaud Legout. *Studying Social Networks at Scale: Macroscopic Anatomy of the Twitter Social Graph*. In Proc. ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, New York, NY, USA, June 2014. ↑108

[Gelman 2006] Andrew Gelman. *Prior Distributions for Variance Parameters in Hierarchical Models (comment on Article by Browne and Draper)*. Bayesian Analysis, vol. 1, no. 3, pages 515–534, 2006. ↑69

[Gjoka et al. 2011] Minas Gjoka, Maciej Kurant, Carter T. Butts and Athina Markopoulou. *Practical Recommendations on Crawling Online Social Networks*. IEEE J. Sel. Areas Commun., vol. 29, no. 9, pages 1872–1892, 2011. ↑55, ↑56, ↑129

[Gjoka et al. 2013] Minas Gjoka, Maciej Kurant and Athina Markopoulou. *2.5 K-Graphs: From Sampling to Generation*. In Proc. IEEE International Conference on Computer Communications (INFOCOM), pages 1968–1976, Turin, Italy, April 2013. ↑55

[Goel & Salganik 2009] Sharad Goel and Matthew J. Salganik. *Respondent-Driven Sampling as Markov Chain Monte Carlo*. Statistics in medicine, vol. 28, no. 17, pages 2202–2229, 2009. ↑56, ↑128

[Goltsev et al. 2008] Alexander V. Goltsev, Sergey N. Dorogovtsev and José F. Mendes. *Percolation on Correlated Networks.* Phys. Rev. E, vol. 78, page 051105, Nov 2008. ↑98

[Gray & Manolopoulos 1996] Stephen K. Gray and David E. Manolopoulos. *Symplectic Integrators Tailored to the Time-Dependent Schrödinger Equation.* The Journal of chemical physics, vol. 104, no. 18, pages 7099–7112, 1996. ↑49

[Gut 2012] Allan Gut. Probability: A Graduate Course (Springer Texts in Statistics). Springer, 2 edition, 10 2012. ↑64, ↑66, ↑80

[Handcock & Gile 2010] Mark S. Handcock and Krista J. Gile. *Modeling Social Networks from Sampled Data.* Annals of Applied Statistics, vol. 4, no. 1, pages 5–25, 2010. ↑56, ↑128

[Heckathorn 1997] Douglas D. Heckathorn. *Respondent-Driven Sampling: A New Approach to the Study of Hidden Populations.* Social Problems, vol. 44, no. 2, pages 174–199, 05 1997. ↑56, ↑128

[Horn & Johnson 2012] Roger A. Horn and Charles R. Johnson. Matrix Analysis. Cambridge University Press, 2 edition, 2012. ↑6

[Iserles 2008] Arieh Iserles. a First Course in the Numerical Analysis of Differential Equations. Cambridge University Press, 2 edition, 2008. ↑48

[Jackman 2009] Simon Jackman. Bayesian Analysis for the Social Sciences. Wiley, 2009. ↑70

[Jerrum & Sinclair 1989] Mark Jerrum and Alistair Sinclair. *Approximating the Permanent.* SIAM journal on computing, vol. 18, no. 6, pages 1149–1178, 1989. ↑17

[Kemeny & Snell 1983] John G. Kemeny and James L. Snell. Finite Markov Chains. Springer, 1983. ↑56, ↑59, ↑88, ↑115, ↑129

[Kempe & McSherry 2008] David Kempe and Frank McSherry. *A Decentralized Algorithm for Spectral Analysis.* Journal of Computer and System Sciences, vol. 74, no. 1, pages 70–83, 2008. ↑7, ↑26, ↑125, ↑127

[Koskinen et al. 2010] Johan H. Koskinen, Garry L. Robins and Philippa E. Pattison. *Analysing Exponential Random Graph (p-Star) Models with Missing Data Using Bayesian Data Augmentation.* Statistical Methodology, vol. 7, no. 3, pages 366–384, May 2010. ↑56, ↑128

[Koskinen et al. 2013] Johan H. Koskinen, Garry L. Robins, Peng Wang and Philippa E. Pattison. *Bayesian Analysis for Partially Observed Network Data, Missing Ties, Attributes and Actors.* Social Networks, vol. 35, no. 4, pages 514–527, October 2013. ↑56, ↑128

[Kunegis 2013] Jérôme Kunegis. *KONECT: The Koblenz Network Collection.* In Proc. International Conference on World Wide Web, WWW '13 Companion, Rio de Janeiro, Brazil, 2013. ↑55

[Leadbetter & Nandagopalan 1989] Malcolm R. Leadbetter and S. Nandagopalan. *On Exceedance Point Processes for Stationary Sequences Under Mild Oscillation Restrictions.* In Proc. Conference on Extreme Value Theory, pages 69–80, Oberwolfach, Germany, December 1989. Springer. ↑96, ↑109

[Leadbetter et al. 1983] Malcolm R. Leadbetter, Georg Lindgren and Holger Rootzén. Extremes and Related Properties of Random Sequences and Processes, volume 21. Springer-Verlag, 1983. ↑94, ↑99

[Lee et al. 2012] Chul-Ho Lee, Xin Xu and Do Young Eun. *Beyond Random Walk and Metropolis-Hastings Samplers: Why You Should Not Backtrack for Unbiased Graph Sampling.* In Proc. ACM SIGMETRICS/PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems, London, UK, 2012. ↑21, ↑56, ↑129

[Leimkuhler & Reich 2004] Benedict Leimkuhler and Sebastian Reich. Simulating Hamiltonian Dynamics. Cambridge University Press, 2004. ↑46, ↑48

[Leskovec & Krevl 2014] Jure Leskovec and Andrej Krevl. *SNAP Datasets: Stanford Large Network Dataset Collection.* `http://snap.stanford.edu/data`, June 2014. ↑39, ↑53, ↑54, ↑55, ↑71, ↑112

[Levin et al. 2008] David A. Levin, Yuval Peres and Elizabeth L. Wilmer. Markov Chains and Mixing Times. American Mathematical Society, 2008. ↑17, ↑18

[Ligo et al. 2014] Jonathan G. Ligo, George K. Atia and Venugopal V. Veeravalli. *A Controlled Sensing Approach to Graph Classification.* IEEE Trans. Signal Process., vol. 62, no. 24, pages 6468–6480, 2014. ↑56, ↑128

[Litvak et al. 2007] Nelly Litvak, Werner R. W. Scheinhardt and Yana Volkovich. *In-Degree and PageRank: Why Do They Follow Similar Power Laws?* Internet Mathematics, vol. 4, no. 2-3, pages 175–198, 2007. ↑102

[Lovász 1993] László Lovász. *Random Walks on Graphs: A Survey.* Combinatorics, Paul erdos is eighty, vol. 2, no. 1, pages 1–46, 1993. ↑100, ↑101

[Lovász 2007] László Lovász. Eigenvalues of graphs. unpublished, 2007. ↑39

[Markovich 2014] Natalia M. Markovich. *Modeling Clusters of Extreme Values.* Extremes, vol. 17, no. 1, pages 97–125, 2014. ↑94, ↑109

[Massoulié et al. 2006] Laurent Massoulié, Erwan Le Merrer, Anne-Marie Kermarrec and Ayalvadi Ganesh. *Peer Counting and Sampling in Overlay Networks: Random Walk Methods.* In Proc. ACM Annual Symposium on Principles of

Distributed Computing (PODC), Denver, Colorado, USA, 2006. ↑56, ↑64, ↑128

[Meyn & Tweedie 2012] Sean P. Meyn and Richard L. Tweedie. Markov Chains and Stochastic Stability. Springer Science & Business Media, 2012. ↑20, ↑80, ↑81

[Mihail et al. 2003] Milena Mihail, Christos Papadimitriou and Amin Saberi. *On Certain Connectivity Properties of the Internet Topology.* In Proc. IEEE Annual Symposium on Foundations of Computer Science (FOCS), pages 28–35, Orlando, FL, USA, March 2003. ↑18, ↑118

[Nelsen 2007] Roger B. Nelsen. an Introduction to Copulas. Springer, New York, 2nd edition, 2007. ↑95, ↑96

[Newman 2002] Mark E. J. Newman. *Assortative Mixing in Networks.* Physical review letters, vol. 89, no. 20, page 208701, 2002. ↑99

[Newman 2006] Mark E. J. Newman. *Finding Community Structure in Networks Using the Eigenvectors of Matrices.* Physical review E, vol. 74, page 036104, 2006. ↑52

[Newman 2010] Mark E. J. Newman. Networks: An introduction. Oxford University Press, 2010. ↑4, ↑123

[Nummelin 2002] Esa Nummelin. *MC's for MCMC'ists.* International Statistical Review, vol. 70, no. 2, pages 215–240, 2002. ↑16

[O'Brien 1987] George L. O'Brien. *Extreme Values for Stationary and Markov Sequences.* Annals of Probability, vol. 15, no. 1, pages 281–291, 1987. ↑97, ↑106

[Ottoni et al. 2013] Raphael Ottoni, João Paulo Pesce, Diego Las Casas, Geraldo Franciscani Jr., Wagner Meira Jr., Ponnurangam Kumaraguru and Virgilio Almeida. *Ladies First: Analyzing Gender Roles and Behaviors in Pinterest.* In Proc. International AAAI Conference on Web and Social Media, September 2013. ↑55

[Pastor-Satorras & Vespignani 2001] Romualdo Pastor-Satorras and Alessandro Vespignani. *Epidemic Spreading in Scale-Free Networks.* Physical review letters, vol. 86, no. 14, page 3200, 2001. ↑55

[Prakash et al. 2010] B. Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju and Christos Faloutsos. *Eigenspokes: Surprising Patterns and Scalable Community Chipping in Large Graphs.* In Proc. Advances in Knowledge Discovery and Data Mining (PAKDD), pages 435–448, June 2010. ↑7

[Ribeiro & Towsley 2010] Bruno Ribeiro and Don Towsley. *Estimating and Sampling Graphs with Multidimensional Random Walks*. In Proc. ACM SIGCOMM Internet Measurement Conference (IMC), Melbourne, Australia, November 2010. ↑55, ↑56, ↑129

[Ribeiro et al. 2012] Bruno Ribeiro, Pinghui Wang, Fabricio Murai and Don Towsley. *Sampling Directed Graphs with Random Walks*. In Proc. IEEE International Conference on Computer Communications (INFOCOM), March 2012. ↑56, ↑129

[Roberts & Rosenthal 2004] Gareth O. Roberts and Jeffrey S. Rosenthal. *General State Space Markov Chains and MCMC Algorithms*. Probability Surveys, vol. 1, pages 20–71, 2004. ↑16

[Ross 1992] Sheldon M. Ross. Applied Probability Models with Optimization Applications. Dover Publications, 1992. ↑85

[Sahai et al. 2012] Tuhin Sahai, Alberto Speranzon and Andrzej Banaszuk. *Hearing the Clusters of a Graph: A Distributed Algorithm*. Automatica, vol. 48, pages 15–24, 2012. ↑26, ↑127

[Shi & Malik 2000] Jianbo Shi and Jitendra Malik. *Normalized Cuts and Image Segmentation*. IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 8, pages 888–905, 2000. ↑8

[Shor 1997] Peter W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*. SIAM journal on computing, vol. 26, no. 5, pages 1484–1509, 1997. ↑37

[Tenenbaum & Pollard 1985] Morris Tenenbaum and Harry Pollard. Ordinary Differential Equations. Dover Publications, Oct 1985. ↑29

[Thompson 2006] Steven K. Thompson. *Targeted Random Walk Designs*. Survey Methodology, vol. 32, no. 1, page 11, 2006. ↑56, ↑128

[Tijms 2003] Henk C. Tijms. a First Course in Stochastic Models. Wiley, 2 edition, 2003. ↑81

[Travers & Milgram 1969] Jeffrey Travers and Stanley Milgram. *An Experimental Study of the Small World Problem*. Sociometry, pages 425–443, 1969. ↑1

[Van Der Hofstad 2016] Remco Van Der Hofstad. *Random Graphs and Complex Networks Vol. I*. Available on `http://www.win.tue.nl/rhofstad/NotesRGCN.pdf` [Accessed: 14 September 2016], 2016. ↑99, ↑108

[Van der Vaart 2000] Aad W. Van der Vaart. Asymptotic Statistics. Cambridge university press, 2000. ↑69

[Venegas-Andraca 2008] Salvador Elias Venegas-Andraca. *Quantum Walks for Computer Scientists.* Synthesis Lectures on Quantum Computing, vol. 1, no. 1, pages 1–119, 2008. ↑36

[Verlet 1967] Loup Verlet. *Computer "experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules.* Physical review, vol. 159, no. 1, page 98, 1967. ↑46

[Von Luxburg 2007] Ulrike Von Luxburg. *A Tutorial on Spectral Clustering.* Statistics and computing, vol. 17, no. 4, pages 395–416, 2007. ↑7, ↑125

[Weng & Zhang 2012] Chengguo Weng and Yi Zhang. *Characterization of Multivariate Heavy-Tailed Distribution Families via Copula.* Journal of Multivariate Analysis, vol. 106, no. 0, pages 178 – 186, 2012. ↑96, ↑97

[Zhukovskiy et al. 2012] Maxim Zhukovskiy, Dmitry Vinogradov, Yuri Pritykin, Liudmila Ostroumova, Evgeniy Grechnikov, Gleb Gusev, Pavel Serdyukov and Andrei Raigorodskii. *Empirical Validation of the Buckley-Osthus Model for the Web Host Graph: Degree and Edge Distributions.* In Proc. ACM International Conference on Information and Knowledge Management (CIKM), Maui, Hawaii, USA, 2012. ↑100