Detecting communities and performing statistical inferences on networks through renewal non-backtracking random walks

by

Behnaz Moradijamei

B.S., Amirkabir University (Tehran Polytechnic), 2007

M.S., Alzahra University, 2011

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Statistics
College of Art and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2019

Abstract

In many sciences—for example Sociology, Biology, and Computer Science—units under study often belong to communities, and units within the same community behave similarly. A way to identify communities may be through how units interact with each other; units within a community may be more likely to interact with each other than units across different communities. This is equivalent to viewing units under study as a network, where nodes are units and edges are drawn between two units if they interact with each other. Hence, a critical problem in these sciences is how to identify communities given a mathematical network.

Since members within the same community are more likely to interact with each other, it may follow that cycles may be more prevalent within communities than across communities. Thus, the detection of these communities may be aided through the use of measures of the local "richness" of cyclic structures. In this dissertation, we develop the *renewal non-backtracking random walk* (RNBRW)— a variant of a random walk in which the walk is prohibited from returning back to a node in exactly two steps and terminates and restarts once it completes a loop—as a way of quantifying this cyclic structure.

Specifically, we propose using the *retracing probability* of an edge—the likelihood that the edge completes a cycle in a RNBRW—as a way of quantifying cyclic structure. Intuitively, edges with larger retracing probabilities should be more important to the formation of cycles, and hence, to the detection of communities.

We show that retracing probabilities can be estimated efficiently through repeated iterations of RNBRW. Additionally, since RNBRW runs can be performed in parallel, accurate estimation can be obtained even when the network contains millions of nodes. We give simulation results

that suggest pre-weighting edges through RNBRW can improve the performance of popular community detection algorithms substantially. This improvement is most significant when the network is sparse. Moreover, the performance of community detection algorithms with this weighting are competitive to other scalable methods that do not allow for weighting of edges.

We also develop a goodness-of-fit test to help determine whether communities exist within a network. We begin with a network on n nodes that follows an unknown random graph model. We test the null hypothesis that the network is a realization of an Erdös-Renyi graph—a random graph in which each edge is equally likely to be formed, and hence, contains no inherent community structure. Rejecting this null implies that the network comes from a distribution with inherent community structure (for example, a planted partition model). To perform our test, we form an $n \times n$ matrix where entries are the retracing probabilities (estimated through RNBRW) of the corresponding edges of the network. We use as our test statistic a scaled version of the largest eigenvalue of this matrix. We perform a simulation study to compare the Type I Error probability and power of our method to that of other spectral approaches for network inference. We conclude by describing connections between RNBRW and the maximum expected cyclic overlap problem and giving theoretical results of RNBRW under the stochastic block model.

Detecting communities and performing statistical inferences on networks through renewal non-backtracking random walks

by

Behnaz Moradi

B.S., Amirkabir University (Tehran Polytechnic), 2007

M.S., Alzahra University, 2011

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Statistics
College of Art and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2019

Approved by:

Major Professor
Dr Michael Higgins

Copyright

Abstract

In many sciences—for example Sociology, Biology, and Computer Science—units under study often belong to communities, and units within the same community behave similarly. A way to identify communities may be through how units interact with each other; units within a community may be more likely to interact with each other than units across different communities. This is equivalent to viewing units under study as a network, where nodes are units and edges are drawn between two units if they interact with each other. Hence, a critical problem in these sciences is how to identify communities given a mathematical network.

Since members within the same community are more likely to interact with each other, it may follow that cycles may be more prevalent within communities than across communities. Thus, the detection of these communities may be aided through the use of measures of the local "richness" of cyclic structures. In this dissertation, we develop the *renewal non-backtracking random walk* (RNBRW)— a variant of a random walk in which the walk is prohibited from returning back to a node in exactly two steps and terminates and restarts once it completes a loop—as a way of quantifying this cyclic structure.

Specifically, we propose using the *retracing probability* of an edge—the likelihood that the edge completes a cycle in a RNBRW—as a way of quantifying cyclic structure. Intuitively, edges with larger retracing probabilities should be more important to the formation of cycles, and hence, to the detection of communities.

We show that retracing probabilities can be estimated efficiently through repeated iterations of RNBRW. Additionally, since RNBRW runs can be performed in parallel, accurate estimation can be obtained even when the network contains millions of nodes. We give simulation results

that suggest pre-weighting edges through RNBRW can improve the performance of popular community detection algorithms substantially. This improvement is most significant when the network is sparse. Moreover, the performance of community detection algorithms with this weighting are competitive to other scalable methods that do not allow for weighting of edges.

We also develop a goodness-of-fit test to help determine whether communities exist within a network. We begin with a network on n nodes that follows an unknown random graph model. We test the null hypothesis that the network is a realization of an Erdös-Renyi graph—a random graph in which each edge is equally likely to be formed, and hence, contains no inherent community structure. Rejecting this null implies that the network comes from a distribution with inherent community structure (for example, a planted partition model). To perform our test, we form an $n \times n$ matrix where entries are the retracing probabilities (estimated through RNBRW) of the corresponding edges of the network. We use as our test statistic a scaled version of the largest eigenvalue of this matrix. We perform a simulation study to compare the Type I Error probability and power of our method to that of other spectral approaches for network inference. We conclude by describing connections between RNBRW and the maximum expected cyclic overlap problem and giving theoretical results of RNBRW under the stochastic block model.

Table of Contents

List of Figures xi

List of Tables xiii

Acknowledgements xiii

Dedication xiv

1 Introduction 1

1.1 Network inference and community detection 1

1.2 Approaches to Community Detection 2

1.2.1 Preliminary and Notation 2

1.2.2 Models 3

1.2.3 Heuristics 8

1.2.4 Weighting methods 10

1.2.5 Scalable methods for community detection 12

1.2.6 Network inference 14

1.3 Methodology 14

1.4 Organization 16

2 Incorporating network cyclic structures to improve community detection 17

2.1 Introduction 17

2.1.1 Using edge weights to improve community detection 20

2.2 Loop Modulus 23

2.3 Edge participation in cyclic topologies 26

2.3.1 Minimum expected cycle overlap problem (MECO) 27
2.3.2 Renewal non-backtracking random walk (RNBRW) 28
2.3.3 Example 30
2.3.4 Algorithm for RNBRW 31
2.4 Results 32
2.4.1 Comparison of MECO and RNBRW 33
2.4.2 Performance of weighted algorithm compared to the other algorithms 33
2.4.3 Sufficient number of walkers 35
2.4.4 Comparsion of the weighting methods in sparse networks with small communities 36
2.4.5 Scalability of the proposed algorithm 39

3 Using cyclic structure to improve inference on networks 42
3.1 Introduction 42
3.2 Network inference problems 45
3.2.1 Erdős-Rényi Model 45
3.2.2 Stochastic block model(SBM) 46
3.3 Properties of RNBRW under Erdős-Rényi graph 47
3.3.1 Setting 47
3.3.2 The true mean of RNBRW 47
3.3.3 Emprirical variance of RNBRW under the null hypothesis 48
3.3.4 Bounds on the variance on RNBRW 49
3.3.5 Covariance and correlation of RNBRW 51
3.4 Goodness of fit statistic 51
3.4.1 RNBRW goodness-of-fit statistic 52
3.5 Attempt to approximate the asymptotic distribution of RNBRW under the null hypothesis 53
3.5.1 Ignoring dependency 54

3.6 Simulated data and results . . . 55
3.6.1 Empirical distribution of RNBRWGOF under the null hypothesis . . 55
3.6.2 Power analysis . . . 56
3.6.3 Type I error . . . 58
3.7 Conclusion . . . 58

4 Probabilistic properties of renewal non-backtracking random walks . . . 60
4.1 Introduction . . . 60
4.2 Random walk on network . . . 61
4.3 Non-backtracking random walk . . . 63
4.4 Renewal non-backtracking random walk . . . 66
4.4.1 Terminating conditions of NBRW and Importance of the last edge . . 67
4.5 Weighting probability of an edge to be retraced terminal . . . 67
4.5.1 Occupation probability . . . 68
4.5.2 Retracing probability . . . 68
4.5.3 Retracing Probability for SBM . . . 70
4.5.4 Computing the edge retracing weights . . . 75
4.6 Asymptotic properties and consistency . . . 76

5 Conclusions and future work . . . 79

Bibliography . . . 82

A RNBRW source code . . . 89

List of Figures

2.1 A schematic of a non-backtracking random walk. Dotted edge represents the retraced ordered edge. 30

2.2 House graph with weights w_1, w_2, where $w_1 = P(\text{edge} \in \gamma)$ and w_2 is normalized retracing values on the edges. 31

2.3 (a) An excerpt of an LFR graph. (b) Same graph weighted by RNBRW. (c) Normalized frequency of retracing of inter and intra community edges. X-axis is the number of times an edge is retraced and the plot is normalized to have area under the curve 1. 34

2.4 Performance analysis on LFR benchmark networks with $n = 500$, average degree 7, and community sizes ranging from 30 to 70. The mixing rate μ adjusts the ratio of within-communities links over all links. (a) The plot depicts the normalized mutual information for community memberships found by Clauset Newman Moore with the improved performances by weighting by Loop modulus and RNBRW. (b) The plot depicts the normalized mutual information for community memberships found by Louvain method with the improved performances by weighting by Loop modulus and RNBRW. 35

2.5 Performance of Louvain equipped by RNBRW weighting is compared with Infomap[1], Label propagation Raghavan et al.[2], Edge betweenness[3], Spinglass Reichardt and Bornholdt[4], and Walktrap[5] algorithms. The LFR benchmark networks have 500 nodes with average degree 7, and community sizes ranging from 30 to 70. The mixing rate μ, for adjusting ratio of intra-communities links over all links are 0.1, 0.2, and 0.3. 36

2.6 (a) Performance of community detection with increasing the number of walkers in both CNM and Louvain using RNBRW weighting for LFR networks with $n = 10,000$ and average degrees 9 and 27. (b) Weighting of Erdős-R'enyi graphs with n and $p = 2\log n/n$ using WERW-Kpath (cannot be paralleled) and RNBRW (with 2 and 5 cores). 37

2.7 Performance analysis of weighting methods on LFR benchmark networks with varying average degree. Comparing the performance of weighting RNBRW with the raw algorithm (no weighting) and weighting WERW-Kpath with the raw algorithm (no weighting). 38

2.8 Performance improvement for different weighting algorithms in sparse LFR benchmarks 39

2.9 Substantial performance boosting in both CNM and Louvain using RNBRW weighting with increasing graph size simultaneously with sparsity. (left) LFR networks with $n = 10,000$ nodes with average degree in x-axis. (right) LFR networks with $n = 100,000$ nodes with average degree in x-axis. 40

3.1 Comparing the distribution of our proposed test under the null hypothesis with true TW distribution with $\beta = 1$ in (a), and comparing Bickel et al.'s method with true TW distribution in (b), $n = 50$ and,$p = 0.15$ with 500 simulations and under Bartlett type correction. 56

4.1 The conditional probability (4.14) for NBW on a planted partition model with $n = 100$ on each partition and different average within block degree and compared to the empirical probability for different t 73

List of Tables

2.1 Performance of algorithms for sparse LFR networks with $\mu = 0.3$ and average degree $\approx \log n$. 40

3.1 Estimated power at different nominal significant levels for n=30 network size. considering three settings of p and q. 57

3.2 Estimated power at different nominal significant levels for n=50 network size. considering three settings of p and q. 57

3.3 Estimated power at different nominal significant levels for n=100 network size. considering three settings of p and q. 57

3.4 Estimated type one error at different nominal significant levels for n=30 network size. 58

3.5 Estimated type one error at different nominal significant levels for n=50 network size. 58

3.6 Estimated type one error at different nominal significant levels for n=100 network size. 59

Acknowledgments

Firstly, I want to thank my advisor, Prof. Michael Higgins for his advice and encouragement and in particular his patience in guiding me up to this point. Secondly, I would like to convey my appreciation to Prof. Pietro Poggi-Corradini for his assistance during this time.

My sincere gratitude also goes to Profs. James Neil, Chen Wu, Gary Gadbury, Vahl, Jager for their support and consideration during this time.

Special thanks to my friends, Yeng, Adam, Sahar, Hojjat, Hazhar, and Sina for being supportive through this journey. I would like to single out Yeng for her sympathetic ear and unconditional help.

I would like to thank my family. Parastoo for her wise counsel, Behnam for his practical suggestions, Shahram for being inspiring and believing in me, Ali for his support and help and my mom for always being encouraging, nurturing and, motivational through my life. Dad, I hope that I have made you proud. I would also like to extend my gratitude to my parents in law for their encouragement and support thought my studies. I want to thank Noushin and Peyman for their support.

I find it extremely hard to express my thankfulness to my dear love: Heman, because it is limitless. He has been with me, beside me, through the entire up and downs of this journey. His patience, love, , care and, encouragement is endless. Heman! Do you know how amazing you are? I am grateful of Liam (Kawa) for making me stronger than ever and for being such a bundle of joy and laughter. I love you to the moon and back.

Dedication

To Heman and Liam

Chapter 1

Introduction

1.1 Network inference and community detection

Communities can be formed by a variety of groups such as nations, religions, sport fans, coworkers and friendships. Additionally, the advancement of the internet has led to the creation virtual communities such as in social networks like Facebook and Instagram. Often, individuals within the same community will behave similarly. Hence, grouping individuals/users based on similar characteristics or behavior is very useful in many domains; some examples are: users with similar political views on social networks, proteins or metabolites with similar functions in biology, assignment of tasks to multiple processors and reducing communication between them in Parallel computing, and more.

Consequently, correctly identifying communities is an important problem and has been studied in different disciplines. One way to identify communities may be through how units interact with each other since units within a community are more likely to socialize than units across different communities. Community detection methods often use the framework of mathematical networks; Units are nodes (vertices) in the graph, and edges are drawn between two nodes if the corresponding units interact with each other. Under this framework, the problem of community detection becomes a graph partitioning problem, where units within each block of the partition are "optimally" connected under some objective.

Commonly, communities are identified through edge prevalence since edges are observed more frequently within communities compared to between them. There are two general approaches to community detection problems: combinatorial optimization such as maximizing the graph modularity and model-based approaches that fit a network model to the data and maximize the likelihood of the observed graph.

1.2 Approaches to Community Detection

1.2.1 Preliminary and Notation

In this section, we introduce the basic notations and definitions used throughout the entire thesis. $G = (V, E)$ represents a binary and symmetric graph G with node set V and edge set E. E refers to the undirected edge set whereas $\overrightarrow{E}$ is a directed edge set. We typically use the notation uv and ij for an unordered edge and $\overrightarrow{ij}$, $\overrightarrow{uv}$ as an ordered edge. We denote $n = |V|$ as the number of nodes and $m = |E|$ as the number of edges.

The adjacency matrix of G is represented by an $n \times n$ matrix A, where element $A_{ij} = 1$ if there is an edge between nodes i and j, and $A_{ij} = 0$ otherwise. We denote d_i as the *degree* of node i, i.e., the number of edges connected to i in unweighted graphs. We define a weighted version of d_i for weighted graphs as

$$d_i \equiv \sum_{j=1}^{n} w_{\vec{ij}}$$

We define a walk by the edges it traverses; a *walk* $c = (\overrightarrow{v_0v_1}, \overrightarrow{v_1v_2}, \ldots, \overrightarrow{v_{k-1}v_k})$ is a vector of edges $\overrightarrow{v_{\ell-1}v_\ell} \in E$, $\ell = 1, 2, \ldots, k$ connecting (possibly non-distinct) nodes $v_0, v_1, v_2, \ldots, v_k \in V$. A *random walk* $c^* = (E_1, E_2, \ldots, E_n)$ is a walk such that:

$$\begin{aligned} P\left(E_{\ell+1} = \overrightarrow{v_\ell v_{\ell+1}} | E_\ell = \overrightarrow{v_{\ell-1}v_\ell}, \ldots, E_1 = \overrightarrow{v_0v_1}\right) &= P\left(E_{\ell+1} = \overrightarrow{v_\ell v_{\ell+1}} | E_\ell = \overrightarrow{v_{\ell-1}v_\ell}\right) \\ &= \frac{w_{\overrightarrow{v_\ell v_{\ell+1}}}}{d_\ell} \end{aligned} \tag{1.1}$$

A walk c is a *simple cycle* if $v_0 = v_k$ and $v_0, v_1, v_2, \ldots, v_{k-1}$ are distinct. Define $C = C_G$ as the set of all simple cycles in G.

The nodes V are partitioned into q communities, numbered 1 through q. Let $g_v \in \{1, \ldots, q\}$ denote the community to which v belongs. In problems of community detection, we wish to uncover the true label g_v for each node $v \in V$. Let $\mathbf{g} = (g_1, g_2, \ldots, g_n)$.

1.2.2 Models

There are two general approaches to community detection problems; maximizing a criteria that represents a graph modularity criteria and model based approaches that fit a network model (e.g. stochastic block model) to the data by maximizing the likelihood of realizing the observed graph. Model-based approaches for community detection have been offering an efficient approach for finding groups of nodes within-interconnected networks. Both approaches are based on the paradigm that within group nodes are connected with a different probability than nodes in different groups.

Modularity maximization

A common objective function for community detection is the graph modularity Newman[6]:

$$M(\mathbf{g}; A) = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta_{g_i g_j}. \tag{1.2}$$

where δ is delta Kronecker function. Intuitively, for a given community label $\mathbf{g}$, the modularity function measures how much within-group and across-group edge formation deviates from independence. In fact, first term

$$\frac{1}{2m} \sum_{ij} A_{ij} \delta_{g_i g_j} = \frac{\sum_{ij} A_{ij} \delta_{g_i g_j}}{\sum_{ij} A_{ij}}$$

measures the ratio of edges that fall in the clusters over all edges. Originally, the modularity function was introduced as a property of the network and it measures the strength of a

partition of a network. A good partition is expected to have a larger value but, when the function reaches its highest value of 1, we face this issue that the entire network is considered as one community. To overcome this issue, the quantity is subtracted from its expected value when edges of the network are rewired and formed randomly. This is known as configuration model. In the configuration model where expected number of all edges between i and j is $\frac{\text{all edges between i and j}}{\text{total possible rewiring}}$. If we rewire all the edges between i and j, the total number of possible rewiring is $2m$. Therefore, the expected number of edges between them is $\frac{d_i d_j}{2m}$, where d_i and d_j are the corresponding degrees of i and j.

In practice $M \geq 0.3$ means that a network has a significant community structure[7]. Better choices of communities $\mathbf{g}$ correspond to larger values of $M(\mathbf{g})$. Choosing communities $\mathbf{g}$ that maximize the modularity—or almost any other graph partitioning objective—is an $\mathcal{NP}$-hard problem[8]; hence, this approach often focuses on heuristic or approximately optimal solutions–see Section 1.2.3 for a review on the heuristics.

Stochastic block model

The stochastic block model (SBM), assumes q blocks labeled by $g_i \in \{1, \cdots, q\}$ with the chance ω_{g_i} of forming an edge between each pair[9]

These partitions are joined together through a multibipartite random graph that for each pair of nodes $i, j \in V$ have $\omega_{g_i g_j}$ probability of connection. Thus we can write symmetric matrix Ω as:

$$\Omega = \begin{pmatrix} \omega_1 & \omega_{12} & \cdots & \omega_{1q} \\ \omega_{21} & \omega_2 & \cdots & \omega_{2q} \\ \vdots & \cdots & \ddots & \vdots \\ \omega_{q1} & \cdots & \cdots & \omega_q \end{pmatrix}_{q \times q}$$

where the diagonal elements of Ω are larger than off-diagonal ones.

Instead of setting $\omega_{g_i g_j}$ as the probability of having an edge between a pair of nodes pair of nodes in groups g_i and g_j, a more general approach assumes as the mean number of edges

with a Poisson distribution. Then self loop is allowed for a node in group g_i with mean number $\frac{1}{2}\omega_{g_i}$. Both approaches lead to similar results for large and sparse graphs.

Assuming an observed network A, to uncover the community components, we should find the parameters that best fits the data to the SBM. In other words, we need to determine Ω and $\mathbf{g}$ that maximize the following likelihood function:

$$\begin{aligned}\mathcal{L}(\Omega, \mathbf{g}|A) &= \Pr(A|\Omega, \mathbf{g}) \\ &= \prod_i \frac{\left(\frac{1}{2}\omega_{g_i}\omega_{g_i}\right)^{A_{ii}/2}}{\left(\frac{1}{2}A_{ii}\right)!} e^{-\omega_{g_i g_i}/2} \prod_{i<j} \frac{\omega_{g_i}\omega_{g_j}^{A_{ij}}}{A_{ij}!} e^{-\omega_{g_i g_i}}\end{aligned} \tag{1.3}$$

Therefore, $\Omega^*, \mathbf{g}^* = \arg\max \mathcal{L}(\Omega, \mathbf{g}|A)$ gives the parameters that most likely generate the observed A. We are interested in $\mathbf{g}^*$ or node memberships.

Instead of finding the maximum of (1.3), it is more convenient to work with the log-likelihood function,

$$\begin{aligned}\ell(\Omega, \mathbf{g}|A) &= \log \Pr(A|\Omega, \mathbf{g}) \\ &= \sum_i \left[\frac{1}{2}A_{ii}\log\frac{1}{2} + \frac{1}{2}A_{ii}\log\omega_{g_i g_i} - \frac{1}{2}\omega_{g_i g_i} - \log\left(\frac{1}{2}A_{ii}!\right)\right] \\ &+ \sum_{i<j}\left(A_{ij}\log\omega_{g_i g_j} - \omega_{g_i g_j} - \log A_{ij}!\right)\end{aligned} \tag{1.4}$$

We can simplify (1.4) by neglecting the terms independent of the parameters:

$$\begin{aligned}\ell(\Omega, \mathbf{g}|A) &= \log \Pr(A|\Omega, \mathbf{g}) \\ &= \frac{1}{2}\sum_{i,j}\left(A_{ij}\log\omega_{g_i g_j} - \omega_{g_i g_j}\right)\end{aligned} \tag{1.5}$$

Degree-corrected block model (DCBM)

Although the above model can handle networks with Poisson or Bernoulli degree distributions, it performs (fits) poorly for network with different degree distribution. To accommodate real world networks with more general distributions, we use another version of SBM, also known as degree-corrected block model (DCBM)[10] that preserves the degrees of nodes.

To start, we know the expected number of edges between two nodes i and j with degrees d_i and d_j is $\frac{d_i d_j}{2m}\omega_{ij}$, where $\frac{d_i d_j}{2m}$ is the probability of having an edge between i and j in the configuration model. Therefore $\omega_{g_i g_j}$ quantifies the probability of having an edge with respect to the configuration model.

Rewriting (1.5) for DCBM:

$$\begin{aligned} \ell\left(\Omega, \mathbf{g}|A\right) &= \log \Pr\left(A|\Omega, \mathbf{g}\right) \\ &= \frac{1}{2}\sum_{i,j}\left(A_{ij}\log\omega_{g_i g_j} - \frac{d_i d_j}{2m}\omega_{g_i g_j}\right) \end{aligned} \tag{1.6}$$

Community detection can be done by finding the best configuration $\mathbf{g}^*$ exhaustively for small network or using heuristics (approximated solution) for larger ones.

Planted partition model

A simplified (and less flexible) variant of SBM is planted partition model where components of Ωfollows:

$$\omega_{g_i g_j} = \begin{cases} \omega_{\text{in}} & \text{if } g_i = g_j \\ \omega_{\text{out}} & \text{otherwise} \end{cases}$$

Using the following relations:

$$\omega_{ij} = \left(\log\omega_{\text{in}} - \log\omega_{\text{out}}\right)\delta_{ij} + \omega_{\text{out}}$$

and

$$\log \omega_{ij} = (\omega_{\text{in}} - \omega_{\text{out}})\, \delta_{ij} + \log \omega_{\text{out}}$$

Using $E[A_{ij}] = d_i d_j \omega_{ij}$, we can rewrite the log-likelihood of the DCBM version of the planted partition:

$$\begin{aligned} \ell(\Omega, \mathbf{g}|A) &= \log \Pr(A|\Omega, \mathbf{g}) \\ &= \frac{1}{2} \sum_{i,j} A_{ij} \left[\delta_{g_i g_j} \log \frac{\omega_{\text{in}}}{\omega_{\text{out}}} + \log \omega_{\text{out}} \right] \\ &\quad - \frac{1}{2} \sum_{ij} \frac{d_i d_j}{2m} \left[(\omega_{\text{in}} - \omega_{\text{out}})\, \delta_{g_i g_j} + \omega_{\text{out}} \right] \end{aligned} \tag{1.7}$$

We know $\sum_{ij} A_{ij} = \sum_i d_i = 2m$, rewriting (1.7):

$$\begin{aligned} &\ell(\Omega, \mathbf{g}|A) = \log \Pr(A|\Omega, \mathbf{g}) \\ &= \frac{1}{2} \log \frac{\omega_{\text{in}}}{\omega_{\text{out}}} \sum_{i,j} \left[A_{ij} - \frac{\omega_{\text{in}} - \omega_{\text{out}}}{\log \omega_{\text{in}} - \log \omega_{\text{out}}} \frac{d_i d_j}{2m} \right] \delta_{g_i g_j} \\ &+ m(\omega_{\text{out}} + \log \omega_{\text{out}}) \end{aligned} \tag{1.8}$$

Assuming we know ω_{in} and ω_{out}

$$mB = \log \frac{\omega_{\text{in}}}{\omega_{\text{out}}}$$

$$\gamma = \frac{\omega_{\text{in}} - \omega_{\text{out}}}{\log \omega_{\text{in}} - \log \omega_{\text{out}}}$$

$$\theta = m(\omega_{\text{out}} + \log \omega_{\text{out}})$$

Then,

$$\begin{aligned}\ell\left(\Omega, \mathbf{g}|A\right) &= \log \Pr\left(A|\Omega, \mathbf{g}\right) \\ &= B\frac{1}{2m}\sum_{ij}\left(A_{ij} - \gamma\frac{k_i k_j}{2m}\right)\delta_{g_i g_j} + \theta\end{aligned} \tag{1.9}$$

Maximizing (1.9) with respect to Ω and g improves the fitness of the model and so (1.9) acts as a measure of goodness of fit. Note that this function is almost the same as (1.2) where originally Newman and Girwan did not include γ but due to deficiencies such as resolution limit[11], later γ proposed as a tuning parameter to avoid this problem[12].

Both modularity and planted partition assume constant ω_{in} or, in other words, that communities are statistically similar. This can cause trouble in networks with heterogeneous size and type of communities.

Regardless of the method, optimal detection of communities is often $\mathcal{NP}$-hard because as the number of units grows the number of possible partitions exponentially increases. Hence, we need a scalable heuristic approach to overcome this problem. Due to equivalence of these methods providing a heuristic for one leads to a solution for the other. Heuristics for modularity maximization can be substituted for alternative MLE methods and there exists well-studied methods that provide acceptable solutions. In the next section, we briefly discuss two popular heuristic algorithms for modularity maximization.

1.2.3 Heuristics

Choosing communities $\mathbf{g}$ that maximize the modularity—or almost any other graph partitioning objective—is an $\mathcal{NP}$-hard problem; hence, this approach often focuses on finding heuristic or approximately optimal solutions[8]. We now detail two such approaches, the CNM method[7] and the Louvain method[13].

CNM method

The CNM method is a greedy algorithm in which communities are iteratively merged so that each merger maximizes the change in modularity. More specifically, each node begins in its own singleton community. The change of modularity obtained from merging communities i and j into a single community, denoted by ΔQ_{ij}, is initialized for all possible pairs of nodes:

$$\Delta Q_{ij} = \begin{cases} 1/m - 2d_i d_j/(2m)^2 & \text{if } A_{ij} \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{1.10}$$

Once the initializations are complete, the algorithm repeatedly selects the best merger, then updates the ΔQ values, until no further mergers are possible. CNM then selects communities according to the largest found modularity. The computational complexity of the CNM algorithm is $O(n^2 \log n)$.

Louvain method

The Louvain method is divided into two phases. The first phase initializes each node into its own community and considers increasing modularity locally by changing a node's community label to that of a neighboring community. For example, the modularity change obtained by moving singular node i from its community to a neighboring community is calculated by:

$$\Delta Q = \left[\frac{\Sigma_{in} + d_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + d_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{d_i}{2m}\right)^2\right] \tag{1.11}$$

where Σ_{in} is sum of internal edge weights within the neighboring community, Σ_{tot} is the sum of all edge weights incident to nodes in the neighboring community, and $d_{i,in}$ is the sum of the weights of edges between node i and nodes in the neighboring community, and m is the sum of the edge weights of the entire network. For each node i, this change in modularity is computed for each community neighboring i, and i is then assigned the community label that corresponds with the largest change. If no positive change in modularity is possible,

then i does not change its community label.

In the next phase, the algorithm generates a new network. Each community is condensed into a single node, and edges between two (possibly non-distinct) communities are condensed into a single edge with weight equal to the sum of the corresponding between-community edge weights. In particular, self-loops in the new network may be formed if the corresponding community in the previous network has more than one node.

These two phases are repeated iteratively until no local improvement in the modularity is possible. The Louvain algorithm examines fewer mergers (only adjacent ones) compared to CNM, and hence, the time complexity is reduced–it runs in linear time for sparse graphs[13].

1.2.4 Weighting methods

The heuristics discussed in Section 1.2.3 try to maximize the graph modularity. A number of studies have proposed steps to enhance the performance of existing community detection algorithms. They showed that including additional aspects of graph structure (rather than graph modularity) can improve community detection, e.g., incorporating information about specific graph structures through edge weights as a preprocessing step.

Identifying communities not only depend on the network connections, but also on the strength of these connections. Granovetter[14] argued that edge weights play different roles in real-world networks, in that strong edges represent the communities and weak edges guarantee the integrity of the entire network. However, a new and important problem of community detection is enriching the edge weights based on further collected information on network topology.

Edge centrality measures are used to re-weight the network before using a desired community detection algorithm. These weights help the algorithm to better distinguish internal and external connection among communities. Ciglan et al.[15] proposed a centrality measure based on the edge assortitavity is the tendency to connect similar nodes based on the sum of edge weights. After finding the assortitivity measure they use a function that weights edges between highly disassortative nodes (nodes with different degrees). The main purpose is to

decrease the importance of disassortative nodes that are connected.

Girvan and Newman[3] propose an algorithm based on edge betweenness, which counts counts the number of times each edge appears in the shortest paths between different pairs of nodes. Meligy et al.[16] cosider weighting the edge based on the edge betweenness. First, they compute the node betweenness centrality and edge betweennes centrality, and then they calculate the Euclidean distance between each pair of connected nodes. Finally, they weight the edge based on the distance measure. Fortunato et al.[17] suggest the information centrality as an alternative measure of edge centrality. The information centrality of a given edge can be defined by the variation efficiency of the graph and a number edges are removed based on decreasing values of information centrality. The complexity of algorithm is $O(m^3n)$ or $O(n^4)$ on a sparse graph. Khadivi et al.[18] propose a weighting method which is a combination of common neighborhood ratio an edge betweenness centrality measure. Zhang et al.[19] also develop a weighting scheme based on SimRank centrality measure[20]. Jeh and Widom[20] introduced SimRank as a recursive similarity score betweenpair of nodes nodes which is the average similarity between in-neighbors of those nodes.

Cyclic structures are important for functional modules in biological networks[21;22]. Vragović and Louis[23] introduce a measure of vertex centrality based on cycles and define loop coefficient of a node as the inverse of average length of shortest path between all pairs of neighbors of i, if i is removed. Moreover, most of the works on analysis of community structure focus on cycles that have lengths of order 3 to 5, Newman[6] and only a few of them discuss the role of higher order cycles in order to detect clusters[24]. Radicchi et al.[25] propose a local algorithm based on the number of short cycles and can outperform the edge betweenness algorithm of Newman and Girvan[26] speedwise, but it may fail in the case of sparse network and in the presence of few cycles. Zhang et al.[27], extend the edge clustering coefficient to bipartite graph with even number of cycles. Recent work by Shakeri et al.[28] develop a method for analyzing cyclic structure in networks called *loop modulus* (LM). LM finds a probability distribution on all cycles in a graph so that the expected overlap between two cycles sampled from this distribution is minimized; from this distribution, a measure of the overall "richness" or density of cycles can then be obtained. Shakeri et al.[28] go on to show

that weighting edges by its expected usage under the optimal distribution of cycles may be helpful in community detection algorithms. Although this method shows great promise, the amount of computation required for LM prohibits its use in very large networks.

On the other hand, random walks have been proposed as a computationally efficient tool for uncovering the structure of networks. In particular, random walk paths may be helpful for identifying communities. Namely, since the density of edges is higher within communities than across communities, the walks will spend a majority of their time traveling within communities because of high density and multiple paths[29;30]. Lai et al.[30] propose a random walk based algorithm to pre-process the network based on observations that random walkers experience similar walks if they start from the same communities. They use random walk to explore local structures of communities. An edge uv gets a higher weight if the nodes u and v are more cosine similar. They suggest that with the length of random walk as small as $\log(n)$ could empirically capture enough information on the local structure.

De Meo et al.[31] introduces a method called weighted edge random walk-K path (WERW-Kpath) that runs a k-hop random walk and weights edges on the walk based on their ability in passing a message. WERW-Kpath works as follows: Given a graph $G = (V, E)$, WERW-Kpath initially assigns to each edge $e \in E$ a weight $w_e = 1$. It then runs many random walks from a randomly chosen source node based on the node degree. The walk length is less than or equal to k and algorithm weights the edges based on their appearance on the walks, which are biased toward edges that already have been seen in the previous walks. They suggest $n \log n$ number of iteration for the algorithm. We will discuss this method more in results section of Chapter 2.

1.2.5 Scalable methods for community detection

In this section we mainly discuss scalable methods of community detection.

Rosvall and Bergstrom[1] introduce the the infomap algorithm based on information flow in networks. It encodes the network into modules such that it maximizes the information of the original network. Then it sends the signal by a channel to a decoder. The decoder has

a limited capacity, decodes the message and constructs a set of possible candidates for the original graph. If the amount of transfered information is large, we have a smaller number of candidates. The complexity of the algorithm is $O(m)$.

Label propagation algorithm is introduced by Raghavan et al.[2], where it initially assumes that each vertex in the network is assigned to the same community as the majority of its neighbors. This method first initiates a group for each vertex and all vertices in entire network randomly are listed in a sequential order. Then, in the ordered vertex set, each vertex takes the label of the majority of its adjacent vertices. The steps will repeat until each vertex takes the same label as the majority of its neighbors. This algorithm runs in $O(m)$.

Girvan and Newman[3] use an edge betweenness algorithm, with the assumption that edges between communities are expected to have higher edge betweenness centrality compared to within community edges. Therefore, removal of more edges with high edge betweenness reveals an underlying group structure of the network. This algorithm runs in $O(nm)$. Reichardt and Bornholdt[4] propose a spin glass algorithm based on the so-called Potts model in statistical physics. In this algorithm, each node can be in one of c spin states, and the connection between pairs of nodes specifies preference of pairs to stay either in the same or different spin state. Then it simulates the spin states of nodes for a given number of steps. This method is limited in that it only identify up to 200 communities. The algorithm is not particularly fast and not deterministic, and the computational complexity of this algorithm for a sparse graph is approximately $O(n^{2+\theta})$ where θ is approximately 1.2 . Edges should connect nodes of the same spin state (community, in the current context), whereas nodes of different states (belonging to different communities) should be disconnected.

Pons and Latapy[5] introduce a hierarchical partitioning method called Walktrap algorithm. The basic idea of this method is that short random walks stay within the same community. First, it calculates the distances between all adjacent nodes. Then, two neighbor communities are merged into a new one. The complexity of this method for a sparse graph is $\mathrm{O}(n^2 \log n)$.

Blondel et al.[13] propose a heuristic hierarchical approach based on modularity maximization which is applicable for a general weighted graph. The first step of this method, assigns a distinct label to each node of the network, which is called the first level partition. In the next step, the node moves to its neighbor's community that has maximum positive modularity value. The algorithm is repeated until the modularity value cannot be increased. The computational complexity of the algorithm is $O(m)$. The method is more limited by storage demands than computational complexity.

1.2.6 Network inference

In statistics literature, several hypothesis testing procedures has been developed to determine the significance of a community detection result. These methods traditionally consider a null model where the entire network is assumed to have no community structure, i.e., testing the Erdős Rényi model with one community. Lei et al.[32] propose a goodness-of-fit test to obtain the community structure using the largest eigenvalue of the normalized adjacency matrix. They also obtain the asymptotic null distribution when under the assumption of $k = o(n^{16})$ holds. Bickel and Sarkar[33] develop a goodness-of-fit test using the largest eigenvalue of centered and re-scaled adjacency matrix and they also derived the asymptotic null distribution under the Erdős Rényi model. Karwa et al.[34] propose a finite sample Monte Carlo goodness of fit test for the stochastic block model which is computationally expensive. Zhao et al.[35] propose a community extraction framework through a series of hypothesis testing and they also show the consistency of their result analytically. Golub and Jackson[36] introduce a method that uses the second largest eigenvalue of the matrix of expected fractions of the links between communities of a multi-type random network.

1.3 Methodology

It has been observed that incorporating cyclic structures into partitioning algorithms can improve the results[6;23–25;27]. In this thesis, we are particularly interested in edge weights

derived from the cyclic structure of graphs. The intuition is that, if edges are more prevalent within a community than across communities, cycles (especially small cycles) may be more prevalent within communities as well. Hence, edge weights that incorporate information about the density of cycles within a graph may be useful in detecting communities within that graph.

An important type of random walk is the non-backtracking random walk (NBRW) such that the random walk is avoided from returning back to a node in exactly two steps[37;38]. While work on NBRW currently focuses on its fast convergence rate, we exploit another useful property of NBRW—its ability to identify cyclic structure. NBRW terminates if the walker reaches a node with degree one. NBRW is particularly interesting due to its faster convergence rate than simple random walk[37]. NBRW has been extensively studied by Fitzner and van der Hofstad[38]. However, for the first time in this dissertation, we will illustrate its utility in improving community detection algorithms. Particularly, we focus only on a special case of NBRW that terminates and restart once the walker visits one of its previously visited nodes and record the last edge that complete the cycle as a retracing edge, denoted by renewal-NBRW or shorten RNBRW.

In this dissertation, first, we extend NBRW to better capture the cyclic structure of networks. We perform an iterative process where, for each iteration, we independently perform an NBRW from a random node until it forms a cycle, record the *retracing edge*—the edge that completes the cycle, and terminate the NBRW. This amounts to performing a renewal process on NBRWs—hence the name *renewal non-backtracking random walk* (RNBRW).

We show that, when weighting edges proportionally to the empirical frequency of terminal edges, RNBRW obtains a measure of cyclic density. Furthermore, the RNBRW method is very computationally efficient and can be used for networks on the order of millions of nodes. We also give an intuition as to why the RNBRW method heuristically approximates loop modulus. Then, we show through simulations that weighting edges by both RNBRW and LM improves the ability of Louvain and CNM methods to detect communities. Moreover, after the weighting step the performance of these algorithms improve significantly.

Additionally, we propose a new goodness-of-fit test to obtain the community structure

using the largest singular value of the normalized and wighted adjacency matrix by RNBRW. Empirically, we investigate the asymptotic behavior of null distribution of the centered and scaled RNBRW weighted adjacency matrix as well as power and type one analysis. Finally, we investigate the analytical and asymptotic for the distribution of RNBRW in the special case of stochastic block models and plant partition model with two groups which is called plant bisection model.

1.4 Organization

We discuss our new weighting method to incorporate the loop structure of the graph through RNBRW and empirically compare our method to other scalable methods of community detection using the runtime and the complexity of the algorithms in Chapter 2. In the next chapter we investigate the use of RNBRW in the problem of network inference by proposing a goodness-of-fit test base on the spectral properties of retracing probability matrix. Moreover, we obtained the analytical result of retracing edges by RNBRW for plant bisection model and discuss the asymptotic results in Chapter 4. Finally, we discuss the conclusions and summary of contributions.

Chapter 2

Incorporating network cyclic structures to improve community detection

2.1 Introduction

A distinguishing property of communities in networks is that cycles are more prevalent within communities than across communities. Thus, the detection of these communities may be aided through the incorporation of measures of the local "richness" of the cyclic structure. In this chapter, we introduce renewal non-backtracking random walks (RNBRW) as a way of quantifying this structure. RNBRW gives a weight to each edge equal to the probability that a non-backtracking random walk completes a cycle with that edge. Hence, edges with larger weights may be thought of as more important to the formation of cycles. Of note, since separate random walks can be performed in parallel, RNBRW weights can be estimated very quickly, even for large graphs. We give simulation results showing that pre-weighting edges through RNBRW may substantially improve the performance of common community detection algorithms. Our results suggest that RNBRW is especially efficient for the challenging case of detecting communities in sparse graphs.

In many sciences—for example Sociology[39], Biology[40], and Computer Science[41]—units under study often belong to communities, and units within the same community behave similarly. Hence, understanding the community structure of units is critical in these sciences, and quite a bit of work has been devoted to the development of methods to detect these communities.

Community detection methods often use the framework of mathematical networks: units are nodes (vertices) in the graph, and edges are drawn between two nodes if the corresponding units interact with each other. Under this framework, the problem of community detection becomes a graph partitioning problem where units within each block of the partition are "optimally" connected under some objective. Commonly, communities are identified through edge prevalence; edges are more frequently observed between nodes in the same community than between nodes in different communities.

Optimal detection of communities is often $\mathcal{NP}$-hard because of an exponentially increasing number of possible partitions as the number of units n grows. Numerous efforts and heuristics have been offered that can provide solutions under large n settings, for example, Blondel et al.[13] and Clauset et al.[7]–referred as Louvain and CNM respectively. In this paper, we are particularly interested in methods that can be categorized into one of two general approaches: first, combinatorial optimization of an object, such as maximizing the graph modularity or finding the sparsest graph cut; second, model based approaches that fit a network model, such as the stochastic block model, to the data and discovers communities by maximizing, for example, the likelihood of realizing the observed graph. For an in depth review on community detection algorithms, see Fortunato[42].

The ability of these algorithms to accurately identify communities may be improved through incorporating additional graph structure—typically integrated into these algorithms through the use of edge weights[18;31;43]. In this paper, we are particularly interested in edge weights derived from the cyclic structure of graphs. Previous works has shown cyclic structure to be useful in detecting communities Newman[6]; Vragović and Louis[23]; Kim and Kim[24]; Radicchi et al.[25]; Zhang et al.[27]. The intuition is that, if edges are more prevalent within a community than across communities, cycles (especially small cycles) may be more

prevalent within communities as well. Hence, edge weights that incorporate information about the density of cycles within a graph may be useful in detecting communities within that graph.

Recent work by Shakeri et al.[28] developed a method for analyzing cyclic structure in networks called *loop modulus* (LM). LM finds a probability distribution on all cycles in a graph so that the expected overlap between two cycles sampled from this distribution is minimized; from this distribution, a measure of the overall "richness" or density of cycles can then be obtained. Shakeri et al.[28] go on to show that weighting edges by its expected usage under the optimal distribution of cycles may be helpful in community detection algorithms. Although this method shows great promise, the amount of computation required for LM prohibits its use in very large networks.

On the other hand, random walks have been proposed as a computationally efficient tool for uncovering the structure of networks. In particular, random walk paths may be helpful for identifying communities. Namely, since the density of edges is higher within communities than across communities, the walks will spend a majority of their time traveling within communities[29;30]. An important variant of random walk is the non-backtracking random walk (NBRW), in which the random walk is prohibited from returning back to a node in exactly two steps[37;38]. While work on NBRW currently focuses on its fast convergence rate, we exploit another useful property of NBRW—its ability to identify cyclic structure.

In this paper we extend NBRW to better capture the cyclic structure of networks. We perform an iterative process where, for each iteration, we independently perform an NBRW from a random node until it forms a cycle, record the *terminal edge*—the edge that completes the cycle, and terminate the NBRW. This amounts to performing a renewal process on NBRWs—hence the name *renewal non-backtracking random walk* (RNBRW).

We show that, when weighting edges proportionally to the empirical frequency of terminal edges, RNBRW obtains a measure of cyclic density. Moreover, the RNBRW method is very computationally efficient, and can be used for networks on the order of millions of nodes. We give intuition as to why the RNBRW method heuristically approximates loop modulus. We show through simulation that weighting edges through both RNBRW and LM improves

the ability of Louvain and CNM methods to detect communities, and community detection using Louvain and CNM using these weightings offers comparable performance in detecting communities to state-of-the-art methods.

In this dissertation, we deviate from conventional literature by defining a walk by the edges it traverses; a *walk* $c = (\overrightarrow{v_0v_1}, \overrightarrow{v_1v_2}, \ldots, \overrightarrow{v_{k-1}v_k})$ is a vector of edges $\overrightarrow{v_{\ell-1}v_\ell} \in E$, $\ell = 1, 2, \ldots, k$ connecting (possibly non-distinct) nodes $v_0, v_1, v_2, \ldots, v_k \in V$. A *random walk* $c^* = (E_1, E_2, \ldots, E_n)$ is a walk such that:

$$\begin{aligned} &P\left(E_{\ell+1} = \overrightarrow{v_\ell v_{\ell+1}} | E_\ell = \overrightarrow{v_{\ell-1}v_\ell}, \ldots, E_1 = \overrightarrow{v_0v_1}\right) \\ &= P\left(E_{\ell+1} = \overrightarrow{v_\ell v_{\ell+1}} | E_\ell = \overrightarrow{v_{\ell-1}v_\ell}\right) = \frac{w_{\overrightarrow{v_\ell v_{\ell+1}}}}{d_\ell} \end{aligned} \tag{2.1}$$

A walk c is a *simple cycle* if $v_0 = v_k$ and $v_0, v_1, v_2, \ldots, v_{k-1}$ are distinct. Define $C = C_G$ as the set of all simple cycles in G.

2.1.1 Using edge weights to improve community detection

The quality of community detection may be improved by adding a pre-process step to weight the graph. We briefly go over some proposed weighting schemes in the literature.

Inspired by message propagation, De Meo et al.[31] introduces a method called weighted edge random walk-K path (WERW-Kpath) that runs a k-hop random walk and weights edges on the walk based on their ability in passing a message. WERW-Kpath works as follows: Given a graph $G = (V, E)$, WERW-Kpath initially assigns to each edge $e \in E$ a weight $w_e = 1$. runs many random walks from a randomly chosen source node based on the node degree. They suggested $n \log n$ number of iteration for the algorithm. WERW-Kpath runs many biased random walks from a randomly chosen source node based on the node degree. The walk length is $\leq k$ and algorithm weights the edges based on their appearance on the walks, which are biased toward edges that already have been seen in the previous walks. We will discuss this method more in Results section. Moreover, Lai et al.[30] suggests using random walk to explore local structures of communities. An edge $\{u, v\}$ gets a higher

weight if the nodes u and v are more cosine similar. They suggested that with the length of random walk as small as $\log(n)$ could empirically capture enough information on the local structure.

Preprocessing the graph by analyzing cyclic topologies

The idea of using cyclic structure to identify communities relays on the known fact that flows tend to stay intra-communities. For example, Klymko et al.[44] focused on using the smallest cycles–triangles–to improve community detection in directed networks by weighting the edges based on *3-cycle cut ratio*. Furthermore, Radicchi et al.[25] incorporate the importance of triangles using edge clustering coefficient. This method is similar to Newman and Girvan[26] in which at each iteration the edge with the smallest clustering coefficient is removed. The complexity of this algorithm is in $O(m^4/n^2)$ and $O(n^2)$ on a sparse graph. This algorithm performs poorly in graphs with few short cycles.

Castellano et al.[45] modified the edge clustering coefficient for the weighted networks, in which number of cycles is multiplied by the edge weight. Also, Zhang et al.[27] extended this to bipartite graphs with even number of cycles. Moreover, Vragović and Louis[23] introduced node loop centrality measure such that communities are build around nodes with high centrality in graph cycles. This algorithm has time complexity of $O(nm)$.

Therefore, we want to take into account the cycles as a measure for group quality and thus find communities by maximizing their richness of cycles. In next sections, we explain a notion called modulus that enables us to analytically quantify richness of loops and edge participation in them. Then motivated by these concepts, we introduce a stochastic approach that is able to handle large graphs efficiently.

The weighting step is readily applied before the community detection, for example the update step of CNM in (1.10) is changed as following for weighted adjacency matrix A^ρ

$$\Delta Q^\rho = \begin{cases} \frac{1}{\rho^T \mathbf{1}} - \frac{2 d_i^\rho d_j^\rho}{(2\rho^T \mathbf{1})^2} & \text{if } A_{ij}^\rho \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

where $k_i^\rho = \sum_j \rho_{ij}$ is the new weighted degree.

In Sections 2.1.1-**??**, we review four of the existing weighting methods in the literature and later compare them with our proposed method in Section 2.4.

Weighting algorithms based on local and global network measures

The use of global measures for determining edge weights requires the knowledge of entire networks. This leads to expensive computation while sometimes lacking the necessary specificity to the community structures. Therefore, incorporating local measures are also encouraged to have trade-off between these two types of information. In the following, we summarized the use of global and local measures in the literature.

Newman and Girvan[26] propose to use of edge-betweeness centrality (EBC) as a way to evaluate the community structure of networks. EBC is introduced by Freeman[46] and for each edge e measures the fraction of geodesic paths between all pairs of nodes that passes through e. Khadivi et al.[18] proposed to use common neighbor ratio in addition to EBC to balance the local and global measurements and weight the graph edges and thus have smaller intercluster weights and also facilitate greedy algorithms in identifying the hierarchical structure of the clusters. Doing this improves the resolution limit of modularity maximization methods and thus improving their performance on finding the communities.
Moreover, the common neighbor ratio for a pair of nodes is the fraction of their neighbors that they share together. Authors propose to use a combination (with tunable parameters) of the two measures as a way to capture the local and global structure of the underlying modular graph.

The runtime complexity of this weighting strategy is dominated by computing the edge-betweenness centrality which is in $\mathcal{O}(nm)$[47] that makes its use prohibitive for large networks. Furthermore, tuning the hyperparameters requires design of efficient heuristics to improve the performance of the weighting algorithm. For our comparisons in this paper, we use the authors suggestions for the parameter values.

Zhang et al.[19] focused on the problem of local and global weighting of the edges to

improve the separation of communities. They proposed to measure the similarity of each pair of nodes based on their similarity to other nodes through an iterative function called SimRank that converges after some iterations.

2.2 Loop Modulus

As discussed previously, cycles are more prevalent within communities than across communities. Hence, community detection algorithms may be improved by finding edge weights that quantify the local "richness" of the cyclic structure. Shakeri et al.[28] provide a novel method for finding these weights called *loop modulus* (LM). Here, we briefly recall the basic concepts.

Let $\mathcal{P}(C_G)$ denote the family of all probability mass functions (pmfs) on C_G. Loop modulus minimizes the expected overlap of two random cycles over the set of all possible pmf's $\mu \in \mathcal{P}(C_G)$.

More precisely, we construct a $(|C_G| \times m)$ matrix $\mathcal{N}$ where entry $N(c, e)$ indicates whether cycle $c \in C_G$ contains $e \in E$:

$$\mathcal{N}(c, e) := \begin{cases} 1, & e \in c, \\ 0, & \text{otherwise.} \end{cases} \tag{2.3}$$

Given two cycles c_1 and c_2, the overlap is the number of shared edges between them. We write this in matrix notation using the $(|C_G| \times |C_G|)$ *overlap matrix* $\mathcal{C}$ with entries:

$$\mathcal{C}(c_1, c_2) := \sum_{e \in E} \mathcal{N}(c_1, e)\mathcal{N}(c_2, e) = |c_1 \cap c_2|. \tag{2.4}$$

With this notation, given a pmf $\mu \in \mathcal{P}(C)$, the expected overlap of two independent random cycles with distribution μ is given by

$$\mathbb{E}_\mu |\underline{c_1} \cap \underline{c_2}| = \sum_{e \in E} \sum_{c_1, c_2 \in C} \mu(c_1)\mu(c_2)\mathcal{N}(c_1, e)\mathcal{N}(c_2, e) = \mu^T \mathcal{N}\mathcal{N}^T \mu = \mu^T \mathcal{C}\mu,$$

where the underlined $\underline{c}$ denotes a random variable.

The *minimum expected cycle overlap* (MECO) problem is the following optimization:

$$\begin{aligned} &\text{minimize} \quad \mathbb{E}_\mu|\underline{c_1} \cap \underline{c_2}| \\ &\text{subject to} \quad \mu \in \mathcal{P}(C). \end{aligned} \tag{2.5}$$

Although an optimal pmf μ^* for (2.9) is always guaranteed, it is in general not unique and computationally not easy to obtain precisely. However, Albin and Poggi-Corradini [48] have shown that (2.9) can be reformulated as a modulus problem and thus approximately optimal pmfs can be generated using a simple algorithm that we will describe below.

First, note that given a pmf $\mu \in \mathcal{P}(C)$, every edge $e \in E$ acquires an *edge-usage probability*

$$\eta(e) := \mathbb{P}_\mu(e \in \underline{c}) = \sum_{c \in C} \mu(c)\mathcal{N}(c, e) = (\mathcal{N}^T\mu)(e),$$

namely, the probability that an edge e belongs to a random cycle with law μ. In particular, the expected overlap given μ can be rewritten as an *energy* (2-norm) of the vector η:

$$\mathbb{E}_\mu|\underline{c_1} \cap \underline{c_2}| = \mu^T\mathcal{N}\mathcal{N}^T\mu = \eta^T\eta = \|\eta\|_2^2.$$

Then (2.9) can be rewritten as

$$\begin{aligned} &\text{minimize} \quad \eta^T\eta \\ &\text{subject to} \quad \eta = \mathcal{N}^T\mu,\ \mu \in \mathcal{P}(C), \end{aligned} \tag{2.6}$$

and this is shown in Albin and Poggi-Corradini [48] to be related to the dual problem of the

following convex optimization problem known as *Loop Modulus*.

$$\begin{aligned} &\text{minimize} \quad \rho^T \rho \\ &\text{subject to} \quad \rho \geq 0,\ \mathcal{N}\rho \geq \mathbf{1}. \end{aligned} \tag{2.7}$$

Here, $\rho \in \mathbb{R}^E$ is thought as a cost function for using an edge and $\mathcal{N}\rho \geq \mathbf{1}$ means that every cycle $c \in C$ gives rise to a constraint on ρ, namely, the total cost paid by c

$$\ell_\rho(c) := \sum_{e \in E} \mathcal{N}(c,e)\rho(e)$$

should be at least 1 dollar.

We write $\mathrm{Mod}_2(C)$ to denote the optimal value in (2.7) and recall that, since this is a quadratic minimization, there always is a unique optimal density ρ^*. The theory of convex duality gives rise to a relation between ρ^* for (2.7) and the unique optimal vector of edge-probabilities η^* for (2.10). Namely,

$$\eta^*(e) = \frac{\rho^*(e)}{\mathrm{Mod}_2(C)} = \mathbb{P}_{\mu^*}(e \in \underline{c}), \tag{2.8}$$

for any optimal pmf μ^* for (2.9).

In the pre-processing step using loop modulus, see Figure 2.2(b), We use $\eta*$ to weight the edges in the graph. The idea is that $\eta(e)$ is larger if edge e is likely to belong to many cycles in the support of optimal measures.

Remark 2.2.1. The optimal measures μ^* are normalized optimal Lagrange multipliers for (2.7). In particular, by complementary slackness, if a loop γ is in the support of some optimal measure μ^*, then the ρ^*-length, $\ell_{\rho^*}(\gamma)$, is 1.

We now describe a simple modulus algorithm. Modulus has nice monotonicity properties.

Algorithm 1 Basic algorithm for approximating densities for $\mathrm{Mod}_2(C)$ with tolerance $0 < \epsilon_{\mathrm{tol}} < 1$.

$\rho' \leftarrow 0$; $\rho_0 \leftarrow \mathbf{1}$
$C' \leftarrow \mathit{ShortestLoop}(\rho_0)$
while $\exists c \in C$ such that $\ell_{\rho'}(c) \leq 1 - \epsilon_{\mathrm{tol}}$ **do**
    $C' \leftarrow C' \cup \{c\}$
    $\rho' \leftarrow \mathrm{argmin}\{\rho^T \rho : \mathcal{N}\rho \geq \mathbf{1}\}$
end while

In particular, given two families of loops C_1, C_2,

$$C_1 \subset C_2 \Longrightarrow \mathrm{Mod}_2(C_1) \leq \mathrm{Mod}_2(C_2).$$

With this in mind, one way to compute modulus is to start with an empty family of loops C' and a density $\rho' \equiv 0$. At every pass, we then find a loop $c \in C$ that violates the constraint $\ell_{\rho'}(c) \geq 1$ and add it to the growing family C'. Finally, we compute optimal ρ^*, η^*,and μ^* for problems (2.7), (2.10), and (2.9) (respectively), for the smaller family C'. This algorithm was shown to converge in Albin et al.[49] and in practice when the algorithm stops the family C' is of size $O(|E|)$. Algorithm 1 provides a version of this basic algorithm where at each iteration one looks for the most violated constraint, i.e., one looks for the lightest cycle given the edge-weights ρ'. See Shakeri et al.[28] for details.

2.3 Edge participation in cyclic topologies

Quantifying the role of each edge in the family of all cycles, guides us to learn about channels of information flows that tend to stay in the communities of the graph. In this section, we discuss how to approach this question and propose a scalable method to approximate this quantity with highly desirable computational properties that allows deploying it on large networks.

2.3.1 Minimum expected cycle overlap problem (MECO)

Let $\mathcal{P}(C)$ denote the family of all probability mass functions (pmfs) on C. We want to minimizes the expected overlap of two randomly chosen cycles over the set of all possible pmf's $\mu \in \mathcal{P}(C_G)$, this is known as the *minimum expected cycle overlap* (MECO) and can be formulated as the following optimization problem:

$$\begin{aligned} &\text{minimize} \quad \mathbb{E}_\mu |\underline{c_1} \cap \underline{c_2}| \\ &\text{subject to} \quad \mu \in \mathcal{P}(C). \end{aligned} \tag{2.9}$$

where the underlined $\underline{c}$ denotes a random variable.

Although an optimal pmf μ^* for (2.9) is always guaranteed, it is in general not unique and computationally not easy to obtain precisely.

Given a pmf $\mu \in \mathcal{P}(C)$, the expected overlap of two independent random cycles with distribution μ is given by

$$\mathbb{E}_\mu |\underline{c_1} \cap \underline{c_2}| = \sum_{e \in E} \sum_{c_1, c_2 \in C} \mu(c_1)\mu(c_2)\mathcal{N}(c_1, e)\mathcal{N}(c_2, e) = \mu^T \mathcal{N}\mathcal{N}^T \mu.$$

Therefore,, every edge $e \in E$ acquires an *edge-usage probability*

$$\eta(e) := \mathbb{P}_\mu(e \in \underline{c}) = \sum_{c \in C} \mu(c)\mathcal{N}(c, e) = (\mathcal{N}^T \mu)(e),$$

namely, the probability that an edge e belongs to a random cycle with law μ. In particular, the expected overlap given μ can be rewritten as an *energy* (2-norm) of the vector η:

$$\mathbb{E}_\mu |\underline{c_1} \cap \underline{c_2}| = \mu^T \mathcal{N}\mathcal{N}^T \mu = \eta^T \eta = \|\eta\|_2^2.$$

Then (2.9) can be rewritten as

$$\begin{aligned} &\text{minimize} \quad \eta^T \eta \\ &\text{subject to} \quad \eta = \mathcal{N}^T \mu,\ \mu \in \mathcal{P}(C), \end{aligned} \tag{2.10}$$

and this is shown in Albin and Poggi-Corradini[48] to be related to the dual problem of the Loop Modulus in Section **??**.

Greedy algorithm to solve MECO

Clemens[50] proposed a a greedy algorithm known as 'Plus-one'. Starting with all weights identically zero, the algorithm chooses a cycle (uniformly at random) and adds 1 to weights of the edges that form the chosen cycle. In the next iterations of the algorithm finds the shortest cycle with the new set of weights and again adds 1 to the edge weights of the shortest cycle until the weights stops changing with a chosen tolerance.

2.3.2 Renewal non-backtracking random walk (RNBRW)

In Section 2.3.1, we discussed a method to find the usages of edges in the (important) cycles and the fact that it is not yet able to be scaled up to moderate size networks. The main reason behind this problem is in both (2.7) or (2.10), we are trying to solve an optimization problem and the computational costs of each iterations is dominated by finding the shortest cycle or girth of the weighted graph. Finding girth is an expensive operation and for large networks becomes prohibitive.

Furthermore, we focus on analyzing a subset of cycles, e.g., active constraints for (2.7), and one consequence of that analysis was obtaining the role of each edge in these subset. In this section, we want to approximate the role of each edge in all cycles,and thus we change the approach by focusing specific edges that shape the cycles. Therefore, we propose to use a variant of random walk called *non-backtracking random walk* where we obtain information

that closely follows a similar concept as MECO.

A non-backtracking random walk (NBRW) starting from a node v_0 with length d is a sequence of vertices that uniformly drawn from the following set

$$W = \{(v_0, v_1, \cdots, v_d) : \ v_i \in V, \ v_i, v_{i+1} \in E, \ v_{i-1} \neq v_{i+1}\} \tag{2.11}$$

There are numerous advantages for NBRW; Alon et al.[37] showed that NBRW explores the graph more efficiently and reaches the stationary distribution faster, than simple random walk with the cost of losing Markov property on the steps. However, by replacing each edge uv in E with two directed edges $\vec{uv}$ and $\vec{vu}$, we can translate NBRW on nodes to directed edges while having Markov chain properties–we denote the set of directed edges with $\vec{E}$. Therefore, we can derive the transition probability[51]

$$\Pr(\vec{uv}, \vec{ij}) = \begin{cases} 1/(d_i - 1) & \text{if } v = i \text{ and } j \neq u \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

where $d_i - 1$ is the out-degree of node i.

In previous works such as De Meo et al.[31], the notion of simple random walk has been used to quantify the idea of information passing and to identify nodes or edges that are more capable of passing information. Both random walk and NBRW behave similarly in the long run and their stationary distribution depends only on the degree of nodes[52]. To reduce this dependability, we rather keep the walk length k small by stopping the NBRW.

Definition 2.3.1 (Retracing edge)**.** *When a NBRW meets one of the previously visited nodes v_k, the last hopped edge $\overrightarrow{v_{k-1}v_k}$ is the retracing edge.*

Therefore, we stop the walk once it meets the previously traversed nodes; this leads to produce a simple loop. For example, in Figure 2.1, we illustrate a NBRW that retraced itself and the *retraced edge* (shown by dotted edge) is particularly important for our subsequent analysis. Note that, if the NBRW reaches a node with degree 1, we discard the walk.

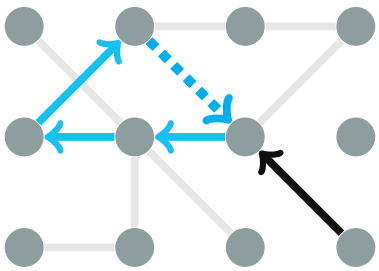


Figure 2.1: A schematic of a non-backtracking random walk. Dotted edge represents the retraced ordered edge.

Definition 2.3.2 (Renewal non-backtracking random walk (RNBRW))**.** *A renewal non-backtracking random walk (RNBRW) is a NBRW that terminates and renew the walk by visiting a retracing edge.*

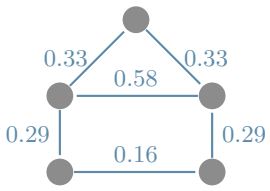
A motivation for renewing the walk is to avoid strong dependency on node degrees of long walks and intuition behind our approach is to find communities of networks based on the capability of community on circulating the message within itself. Nevertheless, the dependency on the node degree still presents when the walkers start their trip from each directed edge uniformly at random.

Highly retraced edges are parts of loops that overlap more, this makes them essential elements in their communities. Therefore, to quantify the importance of edges in this regard, we look at the number of appearances of each edge as a retracing edge for a NBRW that serves as an alternative for the $\eta(e)$ or $\mathbb{P}_\mu(e \in \underline{c})$. We illustrate this approximation in the following example.

2.3.3 Example

For the house graph presented in Figure 2.2, given an edge, we first compute the probability of its participation in important loops using the modulus algorithm and (2.8). Subsequently, we also compute the proportion of appearances of the same edge as a retracing edge in the NBRW process.

As we can see, the values are closely related but not identical. The differences arise from the fact that modulus only considers loops that are active constraints in (2.7). In fact,



(a)

Figure 2.2: House graph with weights w_1, w_2, where $w_1 = P(\text{edge} \in \gamma)$ and w_2 is normalized retracing values on the edges.

both $[c, d, e, c]$ and $[a, b, c, d, a]$ in this example have ρ-length equal to 1 (for the optimal ρ), while the longer loop $[a, b, c, e, d, a]$ has ρ-length strictly greater than 1, hence is never included in the support of optimal measures μ for (2.9) (see Remark 2.2.1). However, a fairly large number of NBRW's may traverse any given loop. Furthermore, different edge retracing values of $\{b, c\}, \{a, d\}$ with $\{a, b\}$ are because more NBRW, e.g., one starting from either (a, d) or (e, c), can be retraced in (a, d) and (b, c).

Therefore, modulus and RNBRW are conceptually close with the difference that modulus is trying to minimize the variance of the edge probabilities arising from families of random cycles, while the RNBRW values arise from a specific family of random cycles.

2.3.4 Algorithm for RNBRW

We use Monte carlo method that builds on repeated random sampling to find the probability of retracing of edges. Each run of the algorithm runs a NBRW and returns the retracing edge (if any) as the sample of retracing edge. The number of times each edge has been retraced by a NBRW corresponds to its probability of retracing with some constant coefficient. The algorithm only requires the graph data and each run is independent of the other. Therefore, we can collect samples in parallel leading to fast convergence. Each run of the algorithm constitutes the following steps:

1 Choosing a random edge $v_0\vec{v}_1$ in $\vec{E}$.

2 Form the walk $w = [v_0, v_1]$.

3 (For $k = \{1, \cdots\}$) The walker continues her walk from v_k to a neighboring node

Algorithm 2 Algorithm for RNBRW from a random directed edge $\vec{uv}$.

```
walk ←empty set
add u, v to walk
while True do
    nexts←neighbors of v
    remove u from nexts
    if nexts is empty break
    next ←choose a node from nexts randomly
    if next ∈walk then
        return (v,next) as the retracing edge
    end if
    add next to walk
    u ← v
    v ←next
end while
```

$v_{k+1} \neq v_{k-1}$.

4 If v_{k+1} is already in w, return $v_k\vec{v_{k+1}}$ as the retracing edge. Otherwise add v_{k+1} to w and go to Step 3 with incrementing $k = k + 1$.

In the Algorithm 2, we present the pseudo-code for NBRW and the retracing edge (if any). By employing a swarm of walkers and picking up their obtained retracing edges, we count the number edge appearances as its retracing value.

We can utilize Algorithm ?? by sending a swarm of walkers independetly from each other and starting from a randomly chosen edge and collect the returned retraced edges at the end. Therefore, one can execure this process as array jobs on cluster of computers efficiently.

2.4 Results

We primarily investigate the performance of two community detection methods Louvain and CNM with and without the proposed weighting methods. Moreover, we compare the performance of Louvain and CNM community detection algorithms, both preprocessed and unweighted, with other popular techniques that follow different paradigms such as Info map Rosvall and Bergstrom[1], Label propagation[2], Edge betweenness[3], Spin glass[4], and Walk

trap[5].

To show the effectiveness of the algorithms, we apply them on LFR benchmarks Lancichinetti et al.[53] and measure their similarity to the ground truth data using normalized mutual information (NMI)[54].

Figure 2.3(a) shows a subgraph of an LFR network with four communities and Figure 2.3(b) shows the preprocessed graph weighted by RNBRW algorithm.

2.4.1 Comparison of MECO and RNBRW

Both MECO (and its equivalent problem modulus) and RNBRW are giving us a measure of edge participation in the cyclic topologies in the network. However, MECO does that by determining a pmf that minimizing the overlap between two randomly drawn loops, in doing so, only a subset of loops are used known as important loops or loops that act as active constraint in the corresponding optimization problem. RNBRW attempts to approximate edge participation in all loops. Therefore, we expect fairly similar improvement on community detection methods using both methods. We illustrate this in Figure 2.4 where we show the improvements on community detection for standard LFR benchmark networks. We plot the mutual information for both the derived membership from the algorithm on each network and the weighted version, compared to the ground truth from LFR. One of our surprising result is that weighted CNM is as good as weighted Louvain.

2.4.2 Performance of weighted algorithm compared to the other algorithms

For the rest of the paper, we focus on weighting by RNBRW, because it scales better with graph size compared to modulus and they show similar performance behavior. For example, we compare five popular algorithms with the Louvain method Blondel et al.[13] equipped with the pre-processing RNBRW step. These algorithms are listed as Infomap, Label propagation, Edge betweenness, Spinglass, and Walktrap. In Figure 2.5(b)-(f), the same benchmarks are used as in Figure 2.4. As shown, the weighted Louvain method performs better community

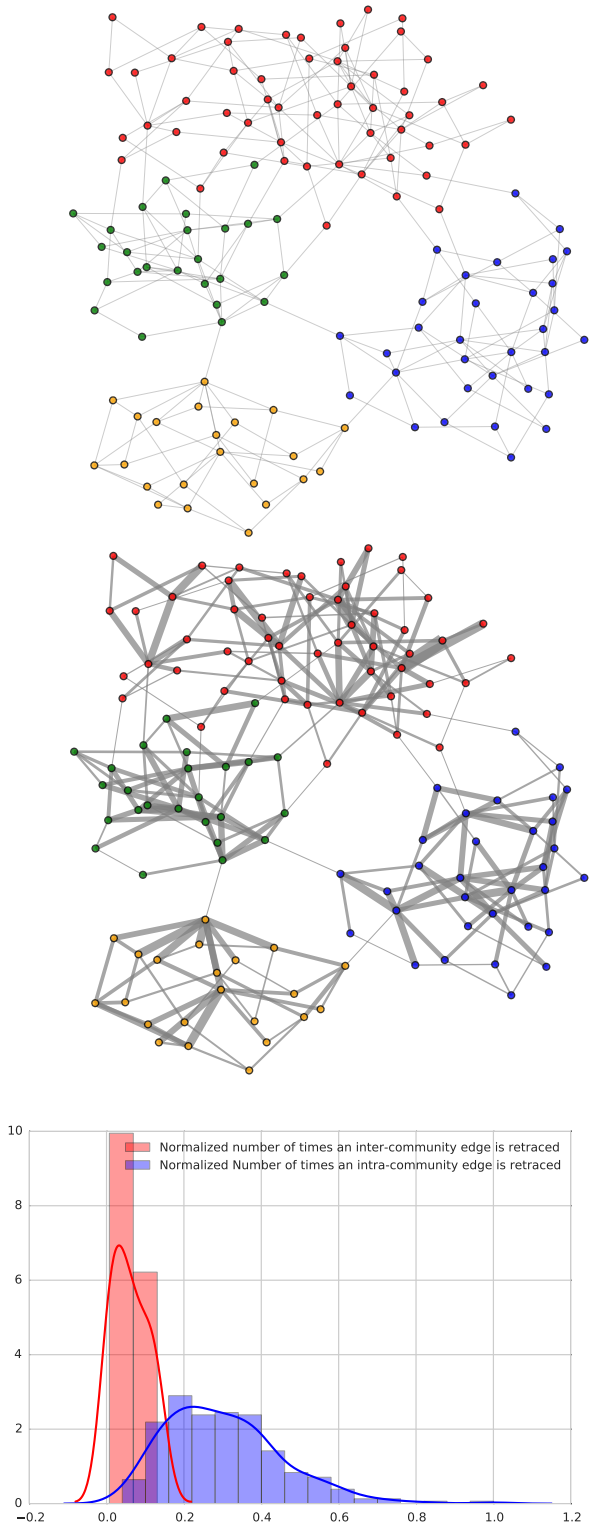
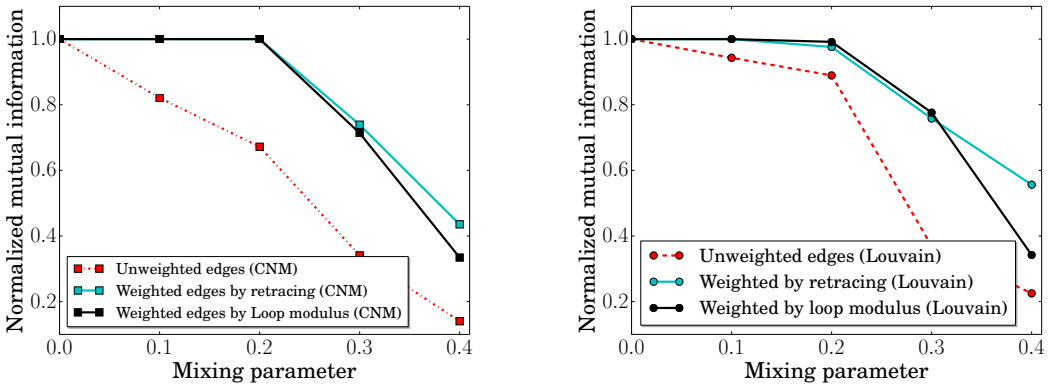


Figure 2.3: (a) An excerpt of an LFR graph. (b) Same graph weighted by RNBRW. (c) Normalized frequency of retracing of inter and intra community edges. X-axis is the number of times an edge is retraced and the plot is normalized to have area under the curve 1.



(a) (b)

Figure 2.4: Performance analysis on LFR benchmark networks with $n = 500$, average degree 7, and community sizes ranging from 30 to 70. The mixing rate μ adjusts the ratio of within-communities links over all links. (a) The plot depicts the normalized mutual information for community memberships found by Clauset Newman Moore with the improved performances by weighting by Loop modulus and RNBRW. (b) The plot depicts the normalized mutual information for community memberships found by Louvain method with the improved performances by weighting by Loop modulus and RNBRW.

detection compared to other algorithms.

2.4.3 Sufficient number of walkers

Since each iteration of RNBRW is independent of others, this facilitates efficient parallel implementations. For instance, an advantage of RNBRW is that its computation is significantly less expensive than WEW-Kpath (see Section 2.1.1). Since we can use Algorithm 2 in a simple parallel fashion mapping each random walk to cores in a computer cluster, we can easily scale the algorithm for large networks.

In Figure 2.6(a), we illustrate the time spent to weight Erdős-R'enyi random networks with n and $p = 2\log n/n$ with WERW-Kpath and RNBRW. We first employ two cores for RNBRW and then 5 cores to show the improvements of performance with the increase of resources[1].

Furthermore, Figure 2.6(b) illustrates another interesting aspect about the efficiency of

[1]Our implementation is in Python 2.7 running on Linux Redhat with AMD processors each with 2799.9 MHz cpu and 2Mb cash.

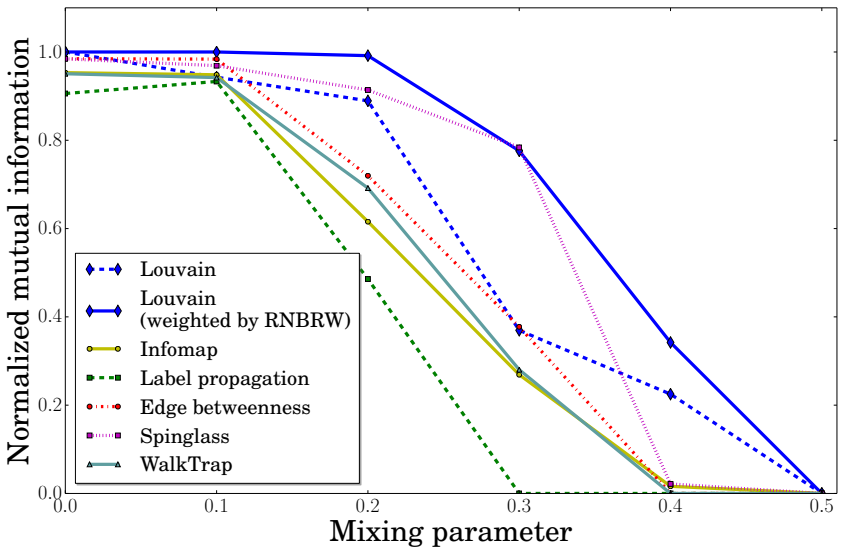


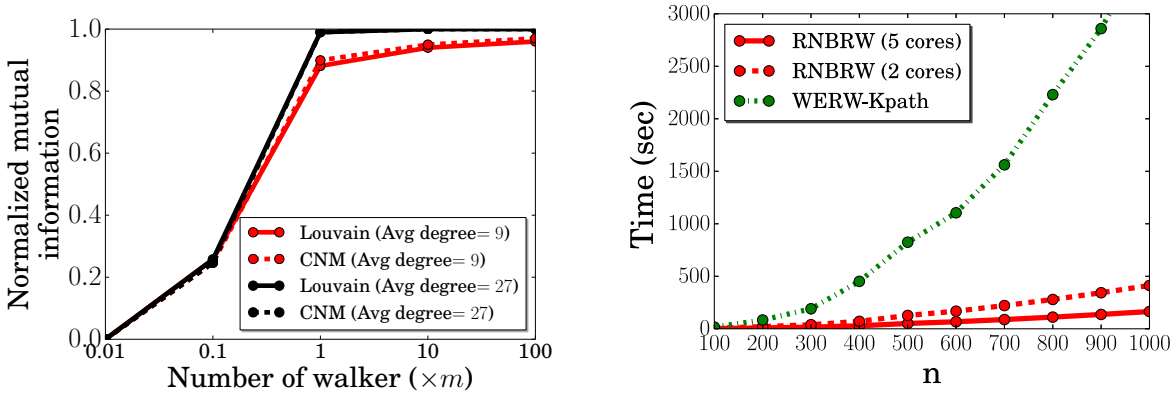
Figure 2.5: Performance of Louvain equipped by RNBRW weighting is compared with Infomap[1], Label propagation Raghavan et al.[2], Edge betweenness[3], Spinglass Reichardt and Bornholdt[4], and Walktrap[5] algorithms. The LFR benchmark networks have 500 nodes with average degree 7, and community sizes ranging from 30 to 70. The mixing rate μ, for adjusting ratio of intra-communities links over all links are 0.1, 0.2, and 0.3.

our algorithm. Namely, we obtain a satisfactory detection quality after only a number m (size of the edge-set) of walkers.

2.4.4 Comparsion of the weighting methods in sparse networks with small communities

Zhao et al.[55] showed consistent detection of communities if the average degree grows at least logarithmically with network size. We test the performance of CNM and Louvain algorithms with and without preprocessing and evaluate how they perform for networks with low average degree.

Another problem that arises with modularity maximization methods is their resolution limit in detecting small communities. This is a critical shortcoming since small size communities are common in large social networks and their size are not necessarily growing with graph size. Fortunato and Barthélemy[11] addresses this issue and explain that communities



(a) (b)

Figure 2.6: (a) Performance of community detection with increasing the number of walkers in both CNM and Louvain using RNBRW weighting for LFR networks with $n = 10,000$ and average degrees 9 and 27. (b) Weighting of Erdős-R'enyi graphs with n and $p = 2\log n/n$ using WERW-Kpath (cannot be paralleled) and RNBRW (with 2 and 5 cores).

with number of internal edges less than $\sqrt{2|E|}$ are most likely mis-detected. This leads to, a resolution limit that holds back heuristics for maximizing the likelihood for stochastic block models and also modularity maximizations Fortunato and Barthélemy [11]. Analyzing the network with the proposed stopped random walk allows to reduce the effect of this problem significantly.

In the following, we test the methods and see how they withstand the challenges of low average degree and existence of small communities for LFR benchmarks, with power law distribution of community size distributions and degree. The community sizes are varying between an upper and a lower bound that falls below the resolution limit and the average degree is changing for different size graphs.

We compare the improvements of CNM algorithm with both preprocessing methods: RNBRW and WERW-Kpath in Figure 2.7. We observe that with varying average degree of the network, WERW-Kpath fails to improve the detection consistently. Also, with increasing μ to a high mixing ratio of 0.4, WERW-Kpath is unable to improve CNM anymore while RNBRW consistently improves the performance of CNM in all range of μ and average degrees.

Although, increasing the mixing rate makes the detection of underlying communities more

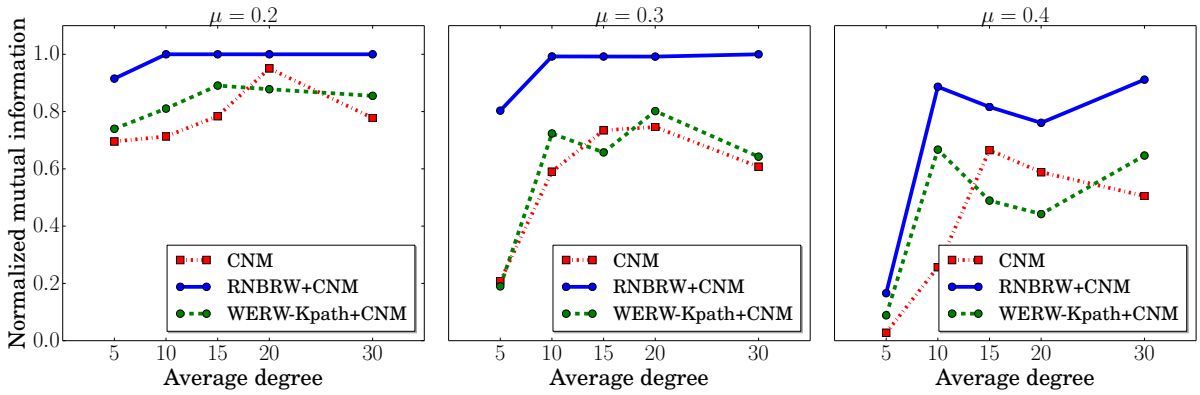


Figure 2.7: Performance analysis of weighting methods on LFR benchmark networks with varying average degree. Comparing the performance of weighting RNBRW with the raw algorithm (no weighting) and weighting WERW-Kpath with the raw algorithm (no weighting).

difficult, the existing algorithms generally performs well as long as the difference between the edge density between and within clusters are distinguishable and the cluster sizes are large compared to the entire network. Likewise, the proposed algorithm has more walker starting from dense communities and thus more retraced edges in the respective community. However, for sparse communities the situation is different.

According to Figure 2.7, for sparse networks that have small average degrees, the performance of most algorithms, except ours, deteriorates significantly. Therefore, from now on we will illustrate the performance of our proposed method in the sparse networks.

Average community sizes n' can vary between the resolution limit which happens when the number of edges in a community falls bellow $\sqrt{2|E|}$, and an arbitrary upper bound. By using the fact that for graph with average degree k, the number of edges are $\frac{nk}{2}$, then $n'k \leq 2\sqrt{nk}$. Hence, the number of nodes in a community with size n' can be adjusted to be less than $2\sqrt{\frac{n}{k}}$ with mixing rate $\mu = 0.4$. In Figure 2.8, we consider LFR benchmark networks with average degree as small as $\log(n)$ and the average community size below the resolution limit. We run our simulation by keeping both k and n' constant and by increasing n to show the robustness of our Community detection weighting method and compared it to other weighting methods (WERW-Kpath, Khadivi, SimRank). This is a difficult condition as most partitioning methods require the degree to increase at least linearly with the network

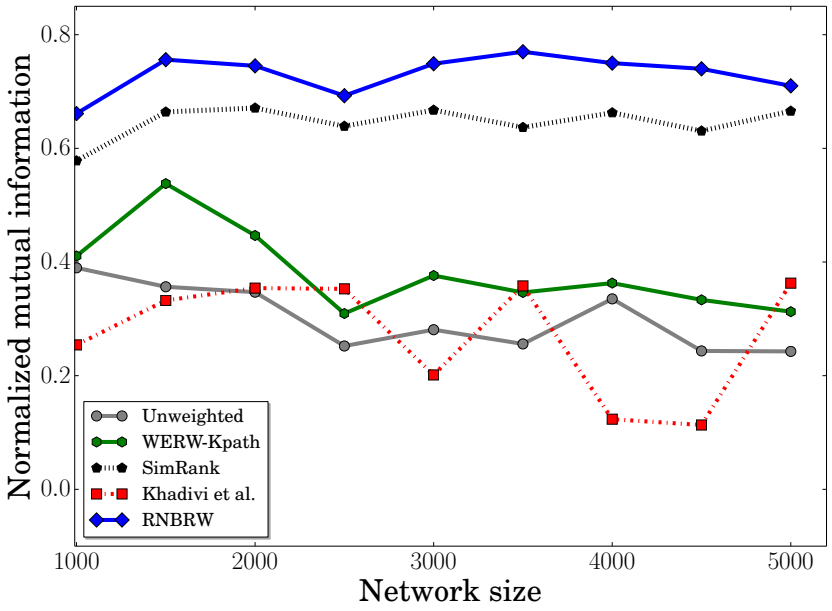


Figure 2.8: Performance improvement for different weighting algorithms in sparse LFR benchmarks

size (see for example Rohe et al.[56]).

2.4.5 Scalability of the proposed algorithm

We repeat our experiments for larger benchmarks[2] and test the same question for a constant mixing rate $\mu = 0.3$ and different average degree in terms of $\log n$ with network sizes 500, $10,000$, $100,000$ and $1,000,000$ nodes and the same improvements in Figure 2.9 confirm the efficacy of our method. We summarize the simulations for sparse LFR benchmarks in Table 2.1 with different sizes and illustrate the utility of our proposed preprocessing methods compared to other algorithms (Edge betweenness algorithm is not included due to its cubic time complexity which is prohibitive for large networks in our Python implementations). The time complexity of LP, WT, and CNM algorithms compared to Louvain is larger by a $\geq n$ factor–Infomap $\mathcal{O}(n^2)$, Label propagation $\mathcal{O}(n^2)$, Spinglass $\mathcal{O}(n^{3.2})$ for sparse networks,

[2]We use Beocat for the large networks. Beocat is a computer cluster located in Kansas State University https://support.beocat.ksu.edu/BeocatDocs/index.php/Main_Page.

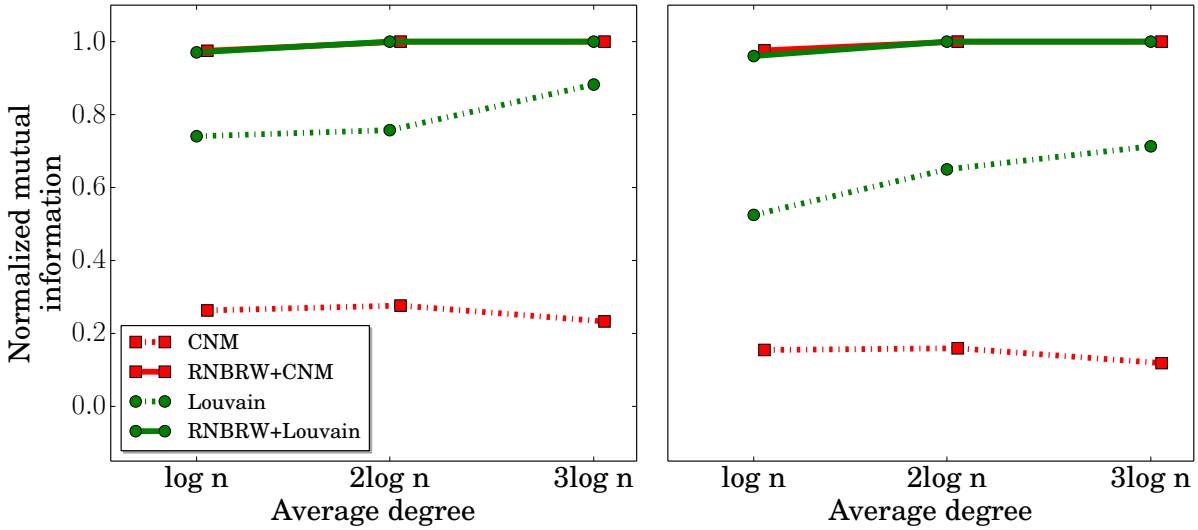


Figure 2.9: Substantial performance boosting in both CNM and Louvain using RNBRW weighting with increasing graph size simultaneously with sparsity. (left) LFR networks with $n = 10,000$ nodes with average degree in x-axis. (right) LFR networks with $n = 100,000$ nodes with average degree in x-axis.

Table 2.1: Performance of algorithms for sparse LFR networks with $\mu = 0.3$ and average degree $\approx \log n$.

network size	average degree	Infomap	LP	WT	CNM	Louvain	RNBRW+CNM	RNBRW+Louvain
$10,000$	$\log n$	0.912	**0.992**	0.763	0.263	0.740	0.974	0.970
	$2\log n$	**1**	0.995	**1**	0.276	0.757	**1**	**1**
	$3\log n$	**1**	**1**	**1**	0.232	0.882	**1**	**1**
$100,000$	$\log n$	0.73	**0.988**	0.644	0.154	0.524	0.975	0.960
	$2\log n$	**1**	**1**	**1**	0.159	0.649	**1**	**1**
	$3\log n$	**1**	**1**	**1**	0.118	0.713	**1**	**1**
$1,000,000$	$\log n$	0.994	0.998	–	0.050	0.192	**0.989**	**0.969**

and walktrap $\mathcal{O}(n^2 \log n)$. Therefore adding a very fast preprocessing step by RNBRW to Louvain seems very reasonable while the performance is often as good as best algorithms out there.

Chapter 3

Using cyclic structure to improve inference on networks

Statistical models such as the stochastic block model have proven to be successful in explaining the structure of communities in real-world network data. In this chapter, we develop a goodness-of-fit test to examine the existence of communities by using a distinguishing property in networks: cyclic structures are more prevalent within communities than across them. We utilize these structures through the use of our novel method, renewal non-backtracking random walk (RNBRW) to the existing goodness-of-fit test. As introduced in Chapter 2, RNBRW is an important variant of random walk in which the walk is prohibited from returning back to a node in exactly two steps and terminates and restarts once it completes a loop. We investigate the use of RNBRW to improve the performance of existing goodness-of-fit tests for community detection algorithms that is based on the spectral properties of the adjacency matrix.

3.1 Introduction

Communities can be formed by a variety of units such as nations, religions, sport fans, coworkers, Facebook friendships. Grouping individuals based on similar characteristics or

behavior is critical in many different disciplines, for example; users with similar political views on social networks, proteins or metabolites with similar functions in biology, and more. Consequently, correctly identifying communities is an important problem and has been studied in different disciplines. Community detection is one of the fundamental problems in network science where often clusters are identified as groups of nodes that are densely connected, i.e., more edge prevalence inside the clusters. Nodes are individuals and edges represent the interaction between them. There are two general approaches to community detection problems: maximizing the graph modularity criteria or maximizing the likelihood of realizing the observed graph from a fitted network model, e.g. stochastic block model (SBM). In statistics literature, stochastic block model has been used broadly as a canonical model to reveal the network structure. However both of these methods are considered as a NP hard problem and approximation algorithms have been proposed to obtain unit's membership.

Recently, several hypothesis testing procedures has been developed to determine the significance of a community detection result. These methods traditionally consider a null model where the entire network is assumed to have no community structure, i.e., testing the Erdős Rényi model with one community. Lei et al.[32] propose a goodness-of-fit test to obtain the community structure using the largest eigenvalue of the normalized adjacency matrix. They also obtain the asymptotic null distribution when under the assumption of $k = o(n^{16})$ holds. Bickel and Sarkar[33] develop a goodness-of-fit test using the largest eigenvalue of centered and re-scaled adjacency matrix and they also derived the asymptotic null distribution under the Erdős Rényi model. Karwa et al.[34] propose a finite sample Monte Carlo goodness of fit test for the stochastic block model which is computationally expensive. Zhao et al.[35] propose a community extraction framework through a series of hypothesis testing and they also show the consistency of their result analytically. Golub and Jackson[36] introduce a method that uses the second largest eigenvalue of the matrix of expected fractions of the links between communities of a multi-type random network.

A common way to identify communities is through how units interact with each other; units within a community are more likely to interact with each other than with units across

different communities. This is equivalent to viewing units of under study as a network, where nodes are units and edges are drawn between two units if they interact with each other. Therefore, it may follow that cycles may be more prevalent within communities than across. De Meo et al.[31], Lai et al.[30], Klymko et al.[44], Radicchi et al.[25], Newman and Girvan[26], and Zhang et al.[27] utilize this idea of network cyclic structure to develop heuristics that can improve the quality of community detection. It has also been observed that the quality of community detection may be further improved by weighing the graph as a pre-process step before feeding it to the desired community detection algorithm.

Recent work[57] develops a novel and scalable method; Renewal non-backtracking random walk(RNBRW), that incorporates both of these concepts by weighing the edges based on their probability of participation in cycles before applying heuristics for community detection. Our simulation results show a substantial boost in performance in comparison to popular and scalable existing methods of community detection. However, in this chapter we introduce a new goodness-of-fit test to obtain the community structure using the largest singular value of the normalized and wighted adjacency matrix by RNBRW. Empirically, we show that the asymptotic null distribution of the centered and scaled RNBRW weighted adjacency matrix converges to a Tracy-Widom distribution. Although simulation results does not show a power improvement in comparison to previous work by[33] for sparse networks, where the network is not weighted by RNBRW, we are able to detect community structure. Additionally, we observe that our method is as powerful as previous method for larger network sizes. Moreover, our simulation results suggests a desirable behavior for type one error across nominal levels of $\alpha = 0.01$,$\alpha = 0.05$ and, $\alpha = 0.1$.

This chapter is organized as follow; first, we briefly discuss common problems of network inference in Section 3.2. In Section **??** we give an overview to RNBRW and its properties. Then in Section 3.4, we propose our goodness-of-fit test. In Section 3.5, we discuss two different approach to approximate Tracy-Widom distribution. Finally, in Section 3.6 we discuss our simulation result.

3.2 Network inference problems

The inference of different types of networks is an active research area in many fields, e.g., biology systems, social sciences. In this work, we consider the assessment of community structure in the network. A class of network inference problems are addressing whether community structure exists in a network. Thus several statistical methods using hypothesis testing has been developed. These methods generally introduces a goodness-of-fit test assuming a null hypothesis of no community structure against an alternative hypothesis of having more than one community. The null hypothesis assumes homogeneous structure where the entire network is considered as one cluster i.e., the probability of having a edge between and within blocks are equal to p and the network is realization of an Erdős-Rényi random graph model with probability of formation of edges p.

3.2.1 Erdős-Rényi Model

Let $G(n,p) = (\Omega, \mathcal{F}, \mathcal{P})$ is a random graph with n nodes and p is the existence probability of each edge. Let Ω be the sample space including all of these graphs. The probability measure of each graph in sample space. We denote the node set V and edge set E, where $n = |V|$ and $m = |E|$. Adjacency matrix of G is represented by $n \times n$ matrix A where element $A_{ij} = 1$ if there is an edge between nodes i and j, and $A_{ij} = 0$ otherwise[58]. The total number of all possible graphs with n nodes in sample space of G is $|\Omega| = 2^M = 2^{\binom{n}{2}}$. Further,the expected number of edges in an Erdős-Rényi model with (n,p) is $p\binom{n}{2}$ and the probability of graph G with k edges is $P(G) = p^k(1-p)^{M-k}$. Moreover, in the case of fixing n and k, there exist $\binom{M}{k}$ possible graphs with n labeled nodes that have k edges, if the probability of drawing edges are equal where $G(n,k) = (\Omega, \mathcal{F}, \mathcal{P})$, in this case: $|\Omega| = \binom{M}{k}$, and $P(G) = 1/\binom{M}{k}$. Hence, the distribution of number of edges of Erdős-Rényi graph is binomial with: $P(x = k) = \binom{M}{k} p^k (1-p)^{M-k}$.

Definition 3.2.1. *Wigner matrix :*

A wigner matrix is a symmetric (Hermitian) matrix scaled by $\sqrt{n}$ *with iid upper diagonal entries* X_{ij} *random variable with zero mean* $E(x_{ij} = 0)$ *and variance* 1, *i.e* $E(x_{ij}^2 = 0)$

entries above the diagonal and diagonal entries are i.i.d random variables with mean zero and finite variance.

Definition 3.2.2. *Gaussian orthogonal ensemble (GOE):*
A wigner matrix is when random variables of both diagonal and off diagonal entries are real Gaussian such that $X_{ij} \sim N(0,1)$ *and* $X_{ii} \sim N(0,2)$*, i.e.,* $E|x_{ij}^2| = 2$.[59]

Definition 3.2.3. *Edge universality:*
Under the distribution of the largest singular value of a Wigner matrix condition on having the forth moment finite, converges weakly to a Tracy-Widom distribution F_β*: i.e.* $\lim_{n\to\infty} P(n^{2/3}(\lambda_n - 2) \leq s) = F_\beta(s)$ *where* λ_n *is the largest eigenvalue of a wigner matrix and* β *can take values* $1, 2, 4$ *depends on the type of Gaussian ensemble of the random variable of entries*[60].

Let A' be a standardized adjacency matrix of Erdős-Rényi graph, $A' = \frac{A_{ij} - Pij}{\sqrt{nP_{ij}(1-P_{ij})}}$. By definition 3.2.1, A' is a general Wigner matrix. In previous work by Bickel and Sarkar [33] under the Erdős-Rényi used estimated $\hat{p}$ and substitute it in A' and obtained $A^\star = \frac{A_{ij} - \hat{P}ij}{\sqrt{(n-1)\hat{p}_{ij}(1-\hat{p}_{ij})}}$. They showed that the distribution of largest eigenvalue of $A^\star$ converges weakly to Tracy Widom distribution with parameter $\beta = 1$:

$$\lim_{n\to\infty} P(n^{2/3}(\lambda_n(A^\star) - 2) \leq s) = F_\beta(s) \tag{3.1}$$

However, their simulation results for finite samples dose not provide a great approximation to TW, therefore they used a parametric bootstrap method named Bartlett type correction to match the first and second moment of Tracy Widom distribution. They further used $\theta = n^{2/3}(\lambda_n(A^\star) - 2)$ as a goodness of fit test in order to test the null hypothesis of the network structure with one community and following a Erdős-Rényi model against the alternative hypothesis of more than one community.

3.2.2 Stochastic block model(SBM)

The network inference literature for testing community structure considers stochastic block model that allows the probability of edge occurrences between pairs of nodes to be different

when nodes are within clusters than between clusters. SBM, assumes a graph $G(n_{g_i}, \omega_{g_i})$ with labels $g_i \in \{1, \cdots, k\}$ and probability ω_{g_i} of having an edge between nodes cluster g_i. These cluster are joined together through a multi bipartite random graph such that for each pair of nodes $i, j \in V$ may have a different probability of connection $\omega_{g_i g_j}$ i.e.,v$P(e_{ij} \in E) = \omega_{g_i g_j}$. The generated network model forms a SBM that has community structure characterized by symmetric matrix $\emptyset xmega$ where diagonal entries are larger than off-diagonals.

3.3 Properties of RNBRW under Erdős-Rényi graph

3.3.1 Setting

We define Y_k as a random vector consisting of random variables of Y_{ij}'s as following:
$Y_{ij}=$ proportion of edge e_{ij} being a retracing edge of a RNBRW. $\mu = \mathbb{E}[Y_{ij}]$ be the true mean of proportion of each edge being a retracing edge and μ be the mean vector of RNBRW weights. Further we define Y as an $n \times n$ symmetric matrix with $Y_{ij} = Y_{ji}$ on off-diagonal elements and $Y_{ii} = 0$.

3.3.2 The true mean of RNBRW

In following lemma we show for the case of Erdős-Rényi network where the probability of edge presence is p, the network is homogeneous. Hence, each edge has $\frac{1}{\binom{n}{2}}$ chance of being a retracing edge (across all Erdős-Rényi graphs).

Lemma 3.3.1. *Under the setting of Erdős-Rényi graph, the expected value of proportion of an edge being retraced by a NBRW is* $2/n(n-1)$ *i.e.* $\mathbb{E}[Y_{ij}] = 1/\binom{n}{2}$.

Proof. Given a graph G, the total number of possible edges is $\binom{n}{2}$. To proceed with proof it is enough to show $\sum_{e_{ij}} \mathbb{E}[Y_{ij}] = 1$. By the law of total expectation, we have $\mathbb{E}[\sum_{e_{ij}} Y_{ij}] = \mathbb{E}[\mathbb{E}[\sum_{e_{ij}} Y_{ij}|G]]$ and since, Y_{ij}'s are random variables of proportions, using the property for law of total probability we have: $\mathbb{E}[\sum_{e_{ij}} Y_{ij}|G] = 1$, substituting in the previous equation,

we have: $\mathbb{E}[\sum_{e_{ij}} Y_{ij}] = \mathbb{E}[1]$ and hence:

$$\mathbb{E}[\sum_{e_{ij}} Y_{ij}] = 1.$$

Now, by using the linearity of expectation: $\sum_{e_{ij}} \mathbb{E}[Y_{ij}] = 1$.

□

3.3.3 Emprirical variance of RNBRW under the null hypothesis

In Table 3.3.3, we summarize our simulation results of the empirical variance of RNBRW under the null hypothesis of no community structure. Our result shows that as the network size increases, the empirical variance of RNBRW weights decreases dramatically. Further, we observed the variance of RNBRW weight is not sensitive with sparsity. We only observe a very slight increase in variance as network get denser. In Table 3.3.3, we run similar simulations to investigate the covariance structure of RNBRW edge weights.

Variance	$n = 25$	$n = 50$	$n = 100$	$n = 200$
$p = 0.5$	1.6e-06	1.41e-06	9e-08	5e-09
$p = 0.4$	2.3e-05	1.409e-06	9e-08	1e-08
$p = 0.35$	2.9e-05	1.7e-06	1.05e-07	6.05e-09
$p = 0.3$	3.6e-05	2.11e-06	1.29e-07	7.95e-09
$p = 0.25$	4e-05	2.6e-06	1.7e-07	1.007e-08
$p = 0.2$	6.5e-05	3.5e-06	2.1e-07	1.3e-08
$p = 0.15$	0.0001	5.1e-06	3e-07	1.8 e-08

Covariance	$n = 25$	$n = 50$	$n = 100$	$n = 200$
$p = 0.5$	-2.8e-08	-4.1e-10	-2.2e-11	-3.3e-13
$p = 0.4$	-4.1e-08	-5.8e-10	-8.8e-12	-1.3e-13
$p = 0.35$	-5.e-08	-7.1e-10	-1.06e-11	-1.63e-13
$p = 0.3$	-6.1e-08	-8.8e-10	-1.325e-11	-2.02e-13
$p = 0.25$	-8.1e-08	-1.1e-09	-1.68e-11	2.55e-13
$p = 0.2$	-1.09e-07	-1.45e-09	-2.19e-11	-3.36e-13

Correlation	$n = 25$	$n = 50$	$n = 100$	$n = 200$
$p = 0.5$	-1.75e-03	-2.9e-04	-2.4e-04	-6.6e-05
$p = 0.4$	-1.78e-03	-4.11e-04	-9.7e-05	-1.3e-05
$p = 0.35$	-1.73e-03	-4.11e-04	-1.01e-04	-2.7e-05
$p = 0.3$	-1.69e-03	-4.17e-04	-1.02e-04	-2.54e-05
$p = 0.25$	-2.02e-03	-3.8e-04	-0.994e-04	-2.53e-05
$p = 0.2$	-1.68e-03	-2.86e-04	-1.04e-04	-2.58e-05

3.3.4 Bounds on the variance on RNBRW

Lemma 3.3.2. *The variance of proportion of a retracing edge of a RNBRW under the setting of Erdős-Rényi graph has the upper bound of* $1/4 - (2/n(n-1))^2$.

Proof. $Var(Y_{ij}) = \mathbb{E}[Y_{ij} - \mathbb{E}[Y_{ij}]]^2 = \mathbb{E}[Y_{ij}^2] - \mathbb{E}[Y_{ij}]^2 = \mathbb{E}[\mathbb{E}[Y_{ij}^2|G]] - \mathbb{E}[\mathbb{E}[Y_{ij}|G]]^2$.
Noting that Y_ij as a proportion $\mathbb{E}[Y_{ij}^2|G] \leq 1/4$ and the result of lemma 3.3.1. Hence:

$$Var(Y_{ij}) = \mathbb{E}[\mathbb{E}[Y_{ij}^2|G]] - \mathbb{E}[\mathbb{E}[Y_{ij}|G]]^2 \leq 1/4 - (2/n(n-1))^2 \tag{3.2}$$

□

Lemma 3.3.3. *The Variance of proportion of a retracing edge of a RNBRW under the setting of Erdős-Rényi graph has the lower bound of* $1/m^2 - (2/n(n-1))^2$.

Proof. Assuming X is a vectors of one's of dimension m and $Y = (Y_1, .., Y_m)$ then:

$$Y.X = \sum_{e_{ij}} Y_{ij} \tag{3.3}$$

$$X.X = \sum_{e_{ij}} 1 = m.$$

$$Y.Y = \sum_{e_{ij}=1}^{m} Y_{ij}^2$$

Taking conditional expectation of both side of 3.3: $\mathbb{E}[Y.X|G] = \mathbb{E}[\sum_{e_{ij}} Y_{ij}|G]$. from lemma 3.3.1, $\mathbb{E}[\sum_{e_{ij}} Y_{ij}|G] = 1$, therefore:

$$\mathbb{E}[Y.X|G] = \mathbb{E}[\sum_{e_{ij}} Y_{ij}|G] = 1 \tag{3.4}$$

Further by using conditional Cauchy-Schwarz inequality

$$(\mathbb{E}[Y.X|G])^2 \leq \mathbb{E}[Y.Y|G]\mathbb{E}[X.X|G]$$

Then:

$$1 \leq \mathbb{E}[Y_{ij}^2|G]\mathbb{E}[X_{ij}^2|G]$$

$$1 \leq \mathbb{E}[\sum_{e_{ij}=1}^{m} Y_{e_{ij}}^2|G]\mathbb{E}[m]$$

$$\frac{1}{m} \leq \mathbb{E}[\sum_{e_{ij}=1}^{m} Y_{ij}^2|G]$$

$$\frac{1}{m} \leq \sum_{e_{ij}=1}^{m} \mathbb{E}[Y_{ij}^2|G]$$

$$\frac{1}{m^2} \leq \mathbb{E}[Y_{ij}^2|G]$$

$$\mathbb{E}[\frac{1}{m^2}] \leq \mathbb{E}[\mathbb{E}[Y_{ij}^2|G]]$$

$$\frac{1}{m^2} \leq \mathbb{E}[Y_{ij}^2]$$

By:

$$Var[Y_{ij}] = \mathbb{E}[Y_{ij}^2] - \mathbb{E}[Y_{ij}]^2 = \mathbb{E}[Y_{ij}^2] - (2/n(n-1))^2$$

we will have:

$$1/m^2 - (2/n(n-1))^2 \leq Var(Y_{ij})$$

□

3.3.5 Covariance and correlation of RNBRW

In this section we discuss a conjecture on the correlation of RNBRW random variables and we further discuss the empirical result of covariance between RNBRW edge weights.

Conjecture 3.3.4. *The correlation between each pair of RNBRW edge weight $Y_{e_{ij}}$ is obtained by:*

$$\rho(Y_{e_{ij}}, Y_{e'_{ij}}) = \frac{2}{n(n-1)} \tag{3.5}$$

In table below we investigate the covariance between RNBRW edge weight. We perform our simulation under different settings of p, probability of formation of the edge of an Erdős-Rényi model and the network size. We varied probability of formation of the edge from 0.1 to 0.9 with 0.1 increment and under fixed network size of $n = 50$. We further investigate this covariance under different network sizes.

3.4 Goodness of fit statistic

We perform Monte Carlo simulations that provide the empirical distribution of the test statistic under the null and alternative hypothesis, and obtain the number of sampled net-

works that their value of the statistic is at least that of the observed network. Our statistical test on community structure is based on the probability distribution of eigenvalues of the normalized edge-weighted matrix is derived by RNBRW.

We attempt to make the best use of asymptotic results on such a distribution when there is no community structure i.e. asymptotic distribution under the null hypothesis. In this section, we also provide a theoretical foundation for our statistic.

3.4.1 RNBRW goodness-of-fit statistic

Traditionally, cyclic topology has been used as a significant criteria of observed graph to quantify its underlying community structure. In the other hand, it has been studied that using random walk, in the network involved algorithms is significantly efficient. In our recent work, we utilized the idea of cycle prevalence within community, through NBRW. Coupling the accuracy and scalability makes RNBRW, an efficient algorithm that without sacrificing any accuracy can highly perform in the cases of scalable networks as well as sparse network without facing any resolution limit problem.

In this section, we utilize RNBRW and its unique properties by constructing the hypothesis of no community structure based on the computation of the largest eigenvalue of weighted adjacency by RNBRW. After proper weighting and shifting of the largest singular value of RBNRW weights, we obtain the empirical distribution of the obtained goodness of fit test. We mimic a similar probabilistic approach to Bickel and Sarkar[33]'s methodology that is based on the largest eigenvalue of centered and scaled adjacency matrix. Our contribution to their method is by incorporating weights derived by RNBRW through using retraing probability matrix instead of adjacency matrix. In following section, we discuss our attempts to approximate TW distribution through the distribution of our proposed test under the null hypothesis.

Each run of our proposed goodness-of-fit statistic which incorporates the graph cyclic structure through RNBRW edge weights is as follow:

- Given a graph G with n vertices.

1 $\hat{p}$ =m/$\binom{n}{2}$

2 Simulate $i = 1 \cdots \rho$ Erdős-Rényi graph $Gs(n, \hat{p})$

3-a Run RNBRW $Gs(n, \hat{p})$ and get the weight matrix.

3-b Estimate σ of RNBRW weight matrix.

4 Run RNBRW on graph G, and get the weight matrix Y.

5 Standardize the Y matrix, get $Y^\star$.

6 $\theta^\star = n^{2/3}(\lambda_n(Y^\star) - 2)$

7 Bartlett-type bootstrap correction on $\theta^\star$ to match first and second moments of TW.

8 Compute the p-value using TW quantiles.

Let Y' be a standardized weighted adjacency matrix of Erdős-Rényi graph, where the edge weights has been obtained by $RNBRW$ algorithm and $\boldsymbol{\mu}$ is the mean of Y_{ij}, which is obtained following lemma 3.3.1.

$$Y' = \frac{Y_{ij} - \mu_{rnbrw}}{\sqrt{n}\sigma_{rnbrw}} \tag{3.6}$$

,

3.5 Attempt to approximate the asymptotic distribution of RNBRW under the null hypothesis

In this section, we attempt to approximate Tracy Widom distribution through the distribution of normalized largest eigenvalue of standardized Y, i.e. retracing probability matrix. To proceed with this we mimic the similar approach as Bickel and Sarkar[33]. They used the result of edge universality lemma for approximating Tracy Widom distribution and this requires the retracing probability matrix satisfy the condition of a Wigner matrix. According to definition 3.2.1, all elements of Y needed to be independent of each other and centered

around zero and the variance on diagonal and off diagonal random elements must be one and finite accordingly. We can obtain the mean and variance condition normalizing Y matrix. However, our simulation result as depicted in 3.3.3 shows that RNBRW edge weights are weakly correlated and our current random variables of retracing probability matrix suffer from lack of independence. Thus we violate *iid* condition of Wigner matrix.

We pursue the following approaches to satisfy the conditions of edge universality lemma considering the existing correlation among Y_{ij}'s and that to be considered as a Wigner matrix. In the first approach, since the correlation is very small and negligible, we continue our discussion with this assumption that edge weights are independent. Although the observed correlations are very weak and ignorable but we attempt to decorrelate the retracing probability matrix using the existing decorrelation transformation methods such as whitening or ZCA mahalonobis in our futute work. Once RNBRW weight matrix satisfies the Wigner matrix, we investigate approximate the asymptotic behavior of its distribution under the null hypothesis. Then, we will address the question of investigating underlying network community structure, by setting the null hypothesis of under no community structure and computing type one errors using the true TW-quantiles and further we investigate the power of our test under parsimonious SBM, planted partition model.

3.5.1 Ignoring dependency

One natural way to approximate TW distribution is assuming the independence between retracing probability matrix and disregarding the existing correlation. We consider the following algorithm for this setting:

We can approximate Tracy Widom under the null, but our GOF test does not approximate TW_1 perfectly and as n grows, our observed quantile will be slightly different than the true quantiles. However, by proper shifting and scaling, as done in previous study by[33] and then using the Bartlett type correction, we attempt to matching the first and second moment of Tracy Widom.

Algorithm 3 RNBRW GOF

for $j = 1$ to 1000 **do**
  $A \leftarrow G(n, p)$
  $m_G \leftarrow$number of edges in G
  $\mu \leftarrow n(n-1)/2$
  $po = m_G/2 * n(n-1)$
  for $i = 1$ to 1000 **do**
    $AA \leftarrow Gs(n, po)$
    $YA \leftarrow RNBRW(Gs)$
    $\hat{\sigma} \leftarrow Var(YA)$
  end for
  $Y \leftarrow RNBRW(G)$
  $Y' \leftarrow \frac{Y_{ij} - \mu_{rnbrw}}{\sqrt{(n-1)}\hat{\sigma}_{rnbrw}}$
  $\theta^\star \leftarrow n^{2/3}(\lambda)(Y') - 2)$
  $((\theta^\star - \hat{\mu}_{\theta^\star})/\hat{\sigma}_{\theta^\star})\sigma_{TW}) + \mu_{TW}$
end for

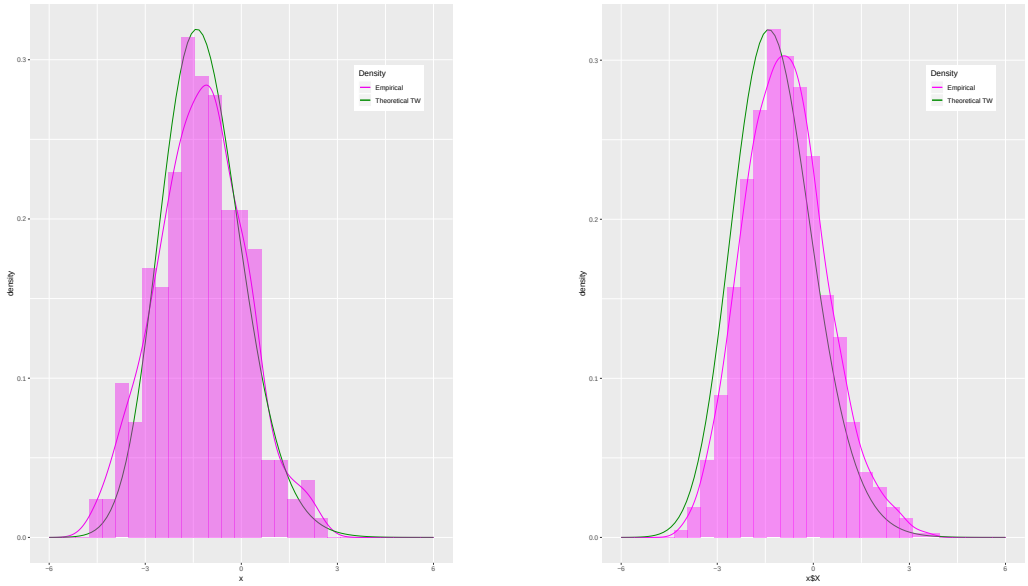
3.6 Simulated data and results

In this section, we first investigate the convergence of empirical distribution of our proposed test. Moreover, we demonstrate the simulation results for probability of making type one and two errors.

3.6.1 Empirical distribution of RNBRWGOF under the null hypothesis

Under the null hypothesis setting where there is no community structure and the network structure follows the form of Erdős-Rényi graph model, we first examine how well the empirical distribution of the RNBRWGOF statistic approximates the Tracy Widom limiting distribution.

Under the setting of negligible correlation, we take a random sample of 1000 replications then obtain distribution of singular value of centered and scaled weighted adjacency by RNBRW. We illustrate the plot of the distributions of our GOF under both settings against TW_1 for different networks of sizes 100 and 500. We observe that Tracy Widom distribution with $\beta = 1$ perfectly approximates the distribution of RNBRWGOF test under the decorrelation



(a) (b)

Figure 3.1: Comparing the distribution of our proposed test under the null hypothesis with true TW distribution with $\beta = 1$ in (a), and comparing Bickel et al.'s method with true TW distribution in (b), $n = 50$ and,$p = 0.15$ with 500 simulations and under Bartlett type correction.

setting.

3.6.2 Power analysis

Considering for the hypothesis test: $H_0 : k = k_0 = 1$, when the the alternative is SBM with more than one community where the true k is greater than 1, here we considered $k = 2$. In this section, we perform the analysis for the power of RNBRWGOF statistic and investigate if our proposed GOF test will reject the null hypotheses of homogeneous graph with one community. We varied the p and q the edge probability of between and within clusters when the true SBM has more than one community. We considered weakly and strongly connected networks with very close p and q, where uncovering community structure is the most challenging and the existing studies fail to recognize them. In the table below, we compute the proportions of correctly rejections at different nominal levels of $\alpha = 0.1, 0.05, 0.01$. We further compared the power of our proposed test with previous work by Bickel and Sarkar[33]. We observed that our proposed GOF test is as powerful previous

Table 3.1: Estimated power at different nominal significant levels for n=30 network size. considering three settings of p and q.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.3$, $q = 0.1$	0.52	0.42	0.23
	$p = 0.6$, $q = 0.1$	1	1	1
	$p = 0.9$, $q = 0.1$	1	1	1
RNBRW	$p = 0.3$, $q = 0.1$	0.127	0.4	0.1
	$p = 0.6$, $q = 0.1$	1	1	0.998
	$p = 0.9$, $q = 0.1$	1	1	1

Table 3.2: Estimated power at different nominal significant levels for n=50 network size. considering three settings of p and q.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.3$, $q = 0.1$	0.94	0.91	0.814
	$p = 0.6$, $q = 0.1$	1	1	1
	$p = 0.9$, $q = 0.1$	1	1	1
RNBRW	$p = 0.3$, $q = 0.1$	0.538	0.404	0.143
	$p = 0.6$, $q = 0.1$	1	1	1
	$p = 0.9$, $q = 0.1$	1	1	1

method for larger network size and larger p.

Table 3.3: Estimated power at different nominal significant levels for n=100 network size. considering three settings of p and q.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.3$, $q = 0.1$	1	1	0.1
	$p = 0.6$, $q = 0.1$	1	1	1
	$p = 0.9$, $q = 0.1$	1	1	1
RNBRW	$p = 0.3$, $q = 0.1$	1	0.998	0.998
	$p = 0.6$, $q = 0.1$	1	1	1
	$p = 0.9$, $q = 0.1$	1	1	1

Table 3.4: Estimated type one error at different nominal significant levels for n=30 network size.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.2$	0.128	0.078	0.018
	$p = 0.35$	0.14	0.07	0.03
	$p = 0.5$	0.144	0.066	0.04
RNBRW	$p = 0.2$	0.042	0.02	0.008
	$p = 0.35$	0.08	0.03	0.01
	$p = 0.5$	0.086	0.038	0.009

Table 3.5: Estimated type one error at different nominal significant levels for n=50 network size.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.2$	0.076	0.052	0.01
	$p = 0.35$	0.052	0.02	0.0
	$p = 0.5$	0.116	0.074	0.026
RNBRW	$p = 0.2$	0.11	0.056	0.01
	$p = 0.35$	0.064	0.026	0.002
	$p = 0.5$	0.114	0.088	0.024

3.6.3 Type I error

In the below table we demonstrate the simulation result of the proportion of rejections by RNBRWGOF under the Erdős-Rényi model compare it to the probability of making type one error of the previous study where is is using a non cyclic-GOF test. The simulation results is done for different settings of p and network size. We observe that our proposed GOF has more desirable type one error and it is close to the nominal level.

3.7 Conclusion

In this chapter, we proposed a novel goodness-of-fit statistic inspired by the cyclic topology of network. Our proposed test is based on the maximum singular value of the centered and

Table 3.6: Estimated type one error at different nominal significant levels for n=100 network size.

	edge probability	$\alpha = 0.1$	$\alpha = 0.05$	$\alpha = 0.01$
Non-cyclic method	$p = 0.2$	0.122	0.066	0.008
	$p = 0.35$	0.072	0.028	0.004
	$p = 0.5$	0.224	0.138	0.054
RNBRW	$p = 0.1$	0.1	0.058	0.018
	$p = 0.15$	0.094	0.036	0.01
	$p = 0.25$	0.084	0.038	0.1

rescaled observed weighted adjacency matrix obtained by RNBRW. We further demonstrated statistical properties of RNBRW, both empirically and analytically. We provided conjectures on the asymptotic null distribution of RNBERWGOF under the setting of ignoring the correlation between edge weights. Although our proposed test is detecting community structure but for the it is not a powerful test for sparse networks. Moreover, our proposed test is powerful when edges are much more frequent within community than across i.e p and q are further. Furthermore, our simulation results of type one error is desirable and the proportion of rejections of our proposed test is close to the nominal level α especially when n is larger.

For the next step, we plan to provide analytical proof for our conjectures. Additionally, we are investigating a different setting that considers whitening Mahalanobis transformation of the edge weights to omit the existing correlation. Finally, we would like to challenge ot developed test with real data set.

Chapter 4

Probabilistic properties of renewal non-backtracking random walks

4.1 Introduction

Random walk (RW) processes are used to describe attributes of electrical networks and belief propagation in social networks[61]. A random walker on a network starts from a node and then chooses an adjacent neighbor uniformly at random as the next step. If the current node has k neighboring nodes, then each of its adjacent node has $\frac{1}{k}$ probability of being selected for the next hop. Hence, some of the selected nodes in the walk might be already traversed by the walker. A variant of random walk process is self avoiding random walk if revisiting nodes are prohibited. I other words, a difference between a simple RW and a self avoiding RW is that the latter keeps record of the set of already visited nodes whereas a RW does not.

A special type of random walk is non backtrack random walk (NBRW) such that the walk is prohibited to hop in its immediate previous step Tishby et al.[62]. Alon et al.[37] showed that non-backtrack random walk can explore random graphs more efficiently than a simple random walk and they reach stationary distribution faster than a RW. Also, Tishby et al.[62] studied the first hitting time distribution of a NBRW on ER networks. They obtained the

termination distribution of a NBRW considering both termination by trapping and retracing for the special case of ER network under some specific assumption.

In previous work[63], we studied the use of non-backtracking random walk by incorporating the cyclic structure of the graph in order to obtain weighting measures as a pre-processing step to improve community detection. Our results showed that our weighting method can improve community detection substantially. Now, in this chapter we obtain the analytical weights and necessary proofs for the consistency of the results and also the sufficient number of iterations that a NBRW needs to run in order to obtain a satisfactory community detection accuracy. Since, the distribution of NBRW depends on the graph topology, we obtain our results for a special case of stochastic block model (planted partition model).

4.2 Random walk on network

Random walk processes are studied extensively in probability theory. The classical problem of random walk studies if RW returns to its initial point with probability one. A special type of RW is Bernoulli random walk, a random walk that the walker can select any step with p/q probability.

A random walk on a graph G is a walk $(v_1 \cdots v_k)$ such that the next step is chosen randomly uniform among the neighbors of the current node. Thus a random walk on a graph starts from a starting point and traverse the graph with probability p such that probability of node v being selected as the the next hop of node u is P_{uv}. This probability is highly related to the degree of node i and when if we choose the next step at uniform it becomes $\frac{1}{d_u}$ where d_u is the degree of node u. Random walks are considered as stochastic process which is defined by the summation of independent, identically distributed random variables[64]. Considering $t \in \{0, 1, 2, \cdots\}$ as time steps, then v_t is considered as a Markov chain which is a sequence of random variables.

Let $\mathbb{P}$ be the matrix transition probability where $\mathbb{P} = P_{uv}$, $u, v \in V$ and π_0 the distribution

of its starting point. The transition probability at each step:

$$P_{u,v} = \begin{cases} 1/d_u & u, v \in E \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Denoting A as the adjacency matrix of graph G, we have $\mathbb{P} = \frac{1}{d}A$. Having D as diagonal matrix of node degrees remains us with $\mathbb{P} = D^{-1}A$

Furthermore, $P^t(u, v)$ is the probability that a walker starts at node u and after t steps arrives at v. Therefore, π_t represents the probability distribution of random walks after t time step, i.e.

$$\pi_t = \pi_0 P^t(u, v) \tag{4.2}$$

For a d-regular graph–every node in the graph has degree d–the Markov chain has symmetric property meaning that probability of traversing from node u to$v$ in t steps is equal to probability of moving from u to v in the same amount of steps. We call π the stationary distribution if $\pi = \pi\mathbb{P}$

The random walk on graph G has Markovian property, i.e., the probability of selecting the next step depends only on the current step probability, thus the walk can be viewed as a classical Markov chain by considering the entire t time steps as a single time step. This property is, however, called memory-less Markov chains.

Lemma 4.2.1 (Stationary distribution)**.** *The Stationary distribution of a simple random walk (Markov chain) on an undirected network G is*

$$\pi_i = \frac{d_i}{2m} \tag{4.3}$$

Proof. For π to be stationary: $\pi = \pi\mathbb{P}$ where $\mathbb{P} = D^{-1}A$, so that:

$$\frac{d_i}{2m} D^{-1} A \tag{4.4}$$

$\frac{1}{2m}$ is a constant, d_i is a n by 1 matrix and D is a diagonal degree matrix, clearly $d_i D^{-1}$ is

Identity matrix. Thus,

$$\frac{d_i}{2m}D^{-1}A = \frac{1}{2m}\mathbb{I}A \tag{4.5}$$

Now, since A is the adjacency matrix, which is an n by n and number of ones in row i represents the number of neighbors of node i, $\mathbb{I}A = d_i$. Therefore:

$$\frac{d_i}{2m}D^{-1}A = \frac{1}{2m}\mathbb{I}A = \frac{d_i}{2m} \tag{4.6}$$

Hence, $\pi_i = \frac{d_i}{2m}$ is the stationary distribution. □

Definition 4.2.2 (Mixing time)**.** *The number of steps that each random walk takes until it gets sufficiently close to its stationary distribution ia called mixing time*[37].

Therefore, mixing rate[37] of an RW determines how fast a walk converges to its stationary distribution, i.e.,

$$\rho = \limsup_{t\to\infty} \max_{u,v\in V} |P^t_{uv} - \pi_v|^{1/k} \tag{4.7}$$

Such that, $P^t_{uv} = P[X_k + t = v | X_k = u]$.

Since, the stationary distribution of an RW strongly depends on the degrees of nodes, thus in the case of dense graph, the mixing rate can be somehow slow[37]. Further putting some restriction on the random walk could lead us to faster stationary distribution. Non-backtracking random walk is an example in this category. Consequently, we will be more focused on non-backtracking random walk which is considerably faster.

4.3 Non-backtracking random walk

In self avoiding random walks, the walker keeps a record of all nodes that already has been traversed[62] to avoid hopping into previously visited nodes. This is done by deleting nodes that are revisited[65]. More specifically, once the walker hops into the next node, the visited node will be deleted from the walk. The walker continues her walk until it reaches a node that all of adjacent nodes of the current node has been already visited.

Therefore, the difference between a simple RW and a self avoiding RW is that the latter keeps record of the set of already visited nodes. The length of a self avoiding walk can take value from one to $|V|-1$. If the length is $|V|-1$ the resultant walk is called a Hamiltonian path. Furthermore, Tishby et al.[65] studied the distribution of path length of self avoiding walks on Erdos-Renyi networks–they called it last hitting time distribution. In other words, they derive the distribution of the latest time the walker can avoid itself, the called it the last hitting time.

More generally, self avoiding random walk models keep track of the last S steps, the last S visited node such that $S \in \{0, 1, \cdots, |V|-1\}$. For $s = 1$, we have non-backtrack random walk (NBRW), i.e., the walker is prohibited to hop into its immediate previous step. An NBRW on graph G is the sequence $(v_1 \cdots v_k)$, where the next step is chosen among the neighbors of the current node except the one that has been already visited before current one. Therefore, NBRW is considered as one special case of self avoiding random walk where the walker keeps record of visited nodes and need this memory in order to choose the next hop, thus it is not memory less and non-Markovian.

The Markovian property facilitates our analytical approach by making the computation of the transition probability easier and on the other hand, the lack of Markovian property will add extra complexity. Luckily, this process can be Markovian by replacing each edge ij in E with two directed edges $\vec{ij}$ and $\vec{ji}$ and translate NBRW on nodes to directed edgesKempton[51]. Hence, defining NBRW on G with the state space $\overrightarrow{E}$ is a finite Markov process.

Alon et al.[37] showed that for a connected d-regular graph and in particular when $d \geq 3$ the stationary distribution is unique and uniform. However, for the case of $d = 2$ it does not reaches its stationary distribution.

A non-backtracking random walk starts from a initial point and traverse the graph with probability P such that probability of node v being selected as the the next hop of node u is P_{uv}. Because of non-backtracking property, this probability of choosing the next step is uniform $\frac{1}{d_u - 1}$ where d_u is the degree of node u. Thus, the transition probability of NBRW

over $\overrightarrow{E}$ state space, at each step is as following:

$$\Pr(\vec{uv}, \vec{ij}) = \begin{cases} \frac{1}{d_i - 1} & \text{if } v = i \text{ and } j \neq u \\ 0 & \text{o/w} \end{cases} \tag{4.8}$$

For each directed edge we have an entry for this matrix, thus the dimension is $2m \times 2m$. Further, each row and each column of matrix P adds up to 1, thus, its double stochastic process. This is very obvious for the rows and to verify this, it is easy to consider the entry of column uv and row ij, if $v = i$ this entry is $1/(d_i - 1)$ otherwise is zero. Hence, in each column we have $d_u - 1$ nonzero entry adding up to one.

Lemma 4.3.1. *The Stationary distribution of a non-backtracking Random walk on a non-bipartite connected directed network G, is :*

$$\pi_i = \frac{1}{2m} \tag{4.9}$$

Proof. It has been proofed by Kempton[51].

Lemma 4.3.2. *Kempton[51] showed that every probability distributionf_u on node could be translated to probability distribution on edge set by $f^*_{uv} = (1/d_u)f_u$. $\pi = \pi\mathbb{P}$ where $\mathbb{P}$ is the transition matrix of the directed edge set with D^* as diagonal matrix of degrees. Each entry of D^* is $d_u - 1$. Therefore, D^* is equivalent to $D - \mathbb{I}$*

So that:

$$\frac{d_i}{2m} D *^{-1} A^* = \frac{d_i}{2m}(D - \mathbb{I})^{-1} A^* \tag{4.10}$$

where A^* is the adjacency matrix for the directed edge set, $\frac{1}{2m}$ is a constant, d_i is a 2m by 1 matrix and thus $d_i D^{-1}$ becomes Identity matrix. Consequently,

$$\frac{d_i}{2m} D^{-1} A = \frac{1}{2m} \mathbb{I} A \tag{4.11}$$

□

Alon et al.[37] showed that Non backtrack random walk can explore graphs more efficiently than a simple random walk and they reach stationary distribution faster than a RW. They proofed that the mixing rate of NBRW is twice faster that the mixing rate of a simple random walk. Considering all mentioned characteristics of NBRW, we are particularly interested to explore the idea of incorporating the cyclic structure of the graph through this fast and efficient special case of random walk. In the next section, we briefly discuss our weighting algorithm namely, Renewal non-backtracking random walk, starting with some definitions that we need through the chapter.

4.4 Renewal non-backtracking random walk

In this section, we first introduce some definitions.

Definition 4.4.1 (Retracing edge)**.** *Retracing edge is the last hopped edge when a NBRW meets one of the previously visited node (the edge that completes a cycle).*

Definition 4.4.2 (renewal non-backtracking random walk)**.** *A renewal non-backtracking random walk (RNBRW) is a NBRW that terminates and restarts when visits a retracing edge.*

Definition 4.4.3 (Set of visited nodes at step t, Γ_t)**.** *For a random non-backtracking walk w at step t, we define the set of visited node Γ_t as the nodes traversed by the walk by that time.*

$$\Gamma_t := \{v \in w_t\}$$

Definition 4.4.4 (Retracing time)**.** *For a random non-backtracking walk $w = \{v_1, v_2, \cdots\}$, we define the retracing time α as:*

$$\alpha = \{t | v_2 \notin \Gamma_1, v_3 \notin \Gamma_2, \cdots, v_t \in \Gamma_{t-1}\},$$

therefore, $\alpha = t$ represents the step of the non-backtracking walk that a node in Γ_t is revisited.

4.4.1 Terminating conditions of NBRW and Importance of the last edge

A motivation for renewing the walk is to avoid strong dependency on node degrees of long walks and intuition behind our approach is to find communities of networks based on the capability of community on circulating the message within itself. Nevertheless, the dependency on the node degree still presents when the walkers start their trip from each directed edge uniformly at random.

Highly retraced edges are parts of loops that overlap more, this makes them essential elements in their communities. Therefore, to quantify the importance of edges in this regard, we look at the number of appearances of each edge as a retracing edge for a NBRW that serves as an alternative for the $\eta(e)$ or $\mathbb{P}_\mu(e \in \underline{c})$ in MECO or Loop Modulus.

4.5 Weighting probability of an edge to be retraced terminal

For the retraced non-backtrack walk for $i = 1 \cdots \rho$, let X_t be a random variable in which $X_t = e_{uv}$ if the edge e_{uv} is chosen by walker to be in the random path and consider $0 < \alpha < \infty$ as the stopping rule $\mathbb{1}_t =$

$$\begin{cases} 1 & \alpha = t \\ 0 & \alpha \neq t \end{cases}$$

where $\mathbb{1}_t$ is the indicator function. Let $w^t(e_{uv})$ be the weight of the terminal edge as probability of the edge e_{uv} in the i^{th} iteration of a NBRW stared at a uniformly chosen directed edge and terminates at step $t = \alpha$.

$$w(e_{ij}) = \sum_{t=3}^{\infty} w^t(e_{uv}) = \omega_e = \mathbb{E}[\sum_{\alpha=3} \mathbb{1}_{X_\alpha = e_{uv}}] = \sum_{t=3}^{\infty} Pr(X_t = e_{uv}, \alpha = t | \alpha > t-1)$$

4.5.1 Occupation probability

Occupying process of a non-Markovian NBRW is difficult to compute because finding the transition probability of the process is a challenge. However, translating the process from nodes to directed edges make it a Markovian process. The transition probability matrix for this part of random walk will be P^t.

$$\Pr(X_t = \vec{e}_{ij}) = \tilde{P}^t \overrightarrow{\pi_0}$$

Further, suppose that f is a probability distribution on node set of the given network, the edge distribution can be obtained $f(u,v) = (1/du)f_u$ (see[51]). Thus the initial edge distribution $\overrightarrow{\pi_0}$:

$$\overrightarrow{\pi_0} = (1/ds)(f(s)) = (1/ds)(1/|V|)$$

where s is the source node, d_s is the degree of s and $f(s)$ is the probability of selecting s which is $1/|v|$. Also, π_0 is a $2m \times 1$ vector and $\tilde{P}$ is a $2m \times 2m$ matrix[51].

$$\Pr(u\vec{v}, i\vec{j}) = \begin{cases} 1/(d_i - 1) & \text{if } v = i \text{ and } j \neq u \\ 0 & \text{o/w} \end{cases} \tag{4.12}$$

$$Pr(X_t = \overrightarrow{e}_{ij}) = 1/(d_i - 1)^t (1/ds)(1/|V|)$$

4.5.2 Retracing probability

Computing probability of retracing in a non-backtracking random walk depends on the topology of graphs thus making it impossible to obtain a closed-form equation for all graph types. Therefore, we compute this quantity for a planted partition stochastic block model $G(n,p,q)$ where the probability matrix P is constant p on the diagonal and another constant q off the diagonal, nodes connecting with probability p are within same community however those connected with probability q, their corresponding edges fall between community. Our approach to solve this problem is considering each block as an ER sub-network $g(n,p)$. Thus,

we review the retracing probability for nodes within blocks first and then for inter blocks.

Retracing Probability in Erdős-Rényi graph models

To obtain edge retracing probability at step t, we first compute its complement probability which is $P(\alpha > t|\alpha > t-1)$ (see Tishby et al.[62] for a comprehensive treatment of this subject).

Other than current and previous node, there exists other possible $N-2$ nodes which may be connected to the current one, each with probability p. Since we assume our walk is not terminated by trapping in a dead end edge thus it is guaranteed that there exists one node adjacent (out of $N-2$ nodes) to the current node (with probability 1), thus there are $N-3$ nodes with probability p and a node guaranteed to be adjacent node.

The expected number of neighbors of the current node that are eligible for walkers to step in at the $t+1$,, is $(N-3)p+1$[1]. Moreover, the number of visited nodes at step $t-1$ is t and the number of unvisited nodes is $N-t$. Note that, the local tree-like structure of Erdős-Rényi networks makes it unlikely that the node which is guaranteed to be connected to the current node has already been visited and if we subtract it from $N-t$ this gives us $(N-t-1)p+1$ possible unvisited neighbors,

$$\mathbb{E}(\mathbb{1}_{\alpha>t|\alpha>t-1}) = \Pr(\alpha > t|\alpha > t-1) = \frac{(N-t-1)p+1}{(N-3)p+1} \tag{4.13}$$

$\Pr(\alpha = t)$ can be computed by:

$$Pr(\alpha = t|\alpha > t-1) = 1 - Pr(\alpha > t|\alpha > t-1) = \frac{tp-2p}{(N-3)p+1}$$

Considering mean degree $c = (N-1)p$ and c_t mean degree of the subnetwork of the unvisited

[1]For a connected Erdős-Rényi network with fixed probability p, otherwise it will be $\sum_{i=1}^{N-3} p_i + 1$, where p_i is the probability of each of $N-3$ being connected to the current node

nodes step t, $c_t = (N - t - 1)p$, thus for large N:

$$\Pr(\alpha > t | \alpha > t - 1) = \frac{c_t - p + 1}{c - 2p + 1} \approx \frac{c_t + 1}{c + 1}$$

$$\Pr(\alpha > t | \alpha > t - 1) \approx \frac{c - c_t}{c + 1}$$

4.5.3 Retracing Probability for SBM

Erdős-Rényi networks are considered as homogeneous graphs that lack mesoscopic structures. In this section, we employ a mixture of ER networks known as stochastic block model (SBM), more specifically its simplest form: planted partition model. In this model, the probability of having an edge between nodes of the same block is constant, p and for nodes in different block, q where $p > q$.

Without loss of generality, we obtain the probability of retracing in the non-backtracking walk of a planted partition model with two blocks of the same sizes which we denote them as C_1 and C_2, where the numbering is only for the ease of following presentation.

We are particularly interested in quantifying the probability of an edge been retraced at step t by a NBW such that it is not terminating in a dead-end with degree 1. To find this probability, we first find its complement: $P(\alpha > t | d > t - 1)$, where d is the length of walk and α is the stopping rule. Rewriting this probability as a product when the step t is chosen within the block and when it happens between them:

$$P(\alpha > t | d > t - 1) = P_w(\alpha > t | d > t - 1) P_b(\alpha > t | d > t - 1) \tag{4.14}$$

Let us focus on the probability of retracing on edges between clusters, i.e. P_b. To find the number of nodes available for step t, when the next hop will be in a different cluster given the walk has $d > t - 1$, there exists $(n - 1)q$ edges for the walker to the next cluster

and since one hop (can be anywhere) is certain, we have:

$$\left(n - \frac{q}{p+q}\right) q + \frac{q}{p+q}$$

where n is the size of each block and $\frac{q}{p+q}$ is the fraction of nodes that is chosen to have $d > t-1$ and is in the next block. Note that, we consider the steps t and $t-1$ are in the same cluster which the latter enforces a difficult condition for our subsequent proofs and thus an acceptable condition. One can easily improves this by considering different cases but we skip this here.

Moreover, the number of unvisited nodes is less by $(t+1)q$ or the expected number of jumps between the blocks (we assume the jumping edges are not sharing nodes in C_2).

$$\left(n - (t+1)q - \frac{q}{p+q}\right) q + \frac{q}{p+q}.$$

leading to:

$$\Pr_b(\alpha > t | d > t-1) = \frac{\left(n - tq - q - \frac{q}{p+q}\right) q + \frac{q}{p+q}}{\left(n - \frac{q}{p+q}\right) q + \frac{q}{p+q}}$$

$$\Pr_b(\alpha = t | d > t-1) = 1 - \Pr_b(\alpha > t | d > t-1)$$

$$\Pr_b(\alpha = t | d > t-1) = \frac{tq^2 + q^2}{nq + \frac{q(1-q)}{p+q}}$$

The first term of the product in (4.14) represents the case when the $t+1^{th}$ hop is remaining in the same block as t. Thus the number of available nodes for $t+1$ given that one of them is certain (which can be anywhere) and two is already in t and $t-1$ ($t-1$ is in the same block):

$$\left(n - 2 - \frac{p}{p+q}\right) p + \frac{p}{p+q}.$$

For the number of visited nodes we have:

$$\left(n - E(S_{vc}) - 2 - \frac{p}{p+q}\right) p + \frac{p}{p+q},$$

resulting in:

$$\Pr_w(\alpha > t | d > t-1) = \frac{\left(n - E(S_{vc}) - 2 - \frac{p}{p+q}\right)p + \frac{p}{p+q}}{\left(n - 2 - \frac{p}{p+q}\right)p + \frac{p}{p+q}}$$

where t_m is the expected numbers of nodes that have been visited in C_1 and S_{vc} is the number of visited nodes in the current cluster. Let S be the number of hops traversed by the NBW until step t, which is $S = t$. We can write S as a sum of i.i.d variables:

$$S_k = X_1 + X_2 + \cdots + X_k,$$

where X_i is the number of hops in one cluster before jumping to another, $i = 1 \cdots k$ and as it is shown below, k is random. It is enough to show X_i follows a geometric distribution with the following probability mass function:

$$\Pr(X_i = x) = (1 - q^*)^{x-1} q^*$$

and $\mathbb{E}(X_i) = \frac{1}{q^*}$, where $q^* = \frac{q}{p+q}$.

Using Wald equality:

$$\mathbb{E}(S_k) = \mathbb{E}(X_i)\mathbb{E}(k) = \frac{\mathbb{E}(k)}{q^*},$$

and since we know the total number of steps up to step t is t thus, $\mathbb{E}(k)$ can be found from $t = \frac{\mathbb{E}(k)}{q^*}$ or

$$\mathbb{E}(k) = tq^*.$$

Since $\mathbb{E}(k)$ should be an integer we choose its floor. We wish to find the number of visited neighbor in current cluster, i.e., S_{vc}. We can look into even and odd cases of $\mathbb{E}(k)$;

$$S_{vc} = \begin{cases} X_1 + X_3 + \cdots + X_k & k \text{ even} \\ X_2 + \cdots + X_k & k \text{ odd} \end{cases}$$

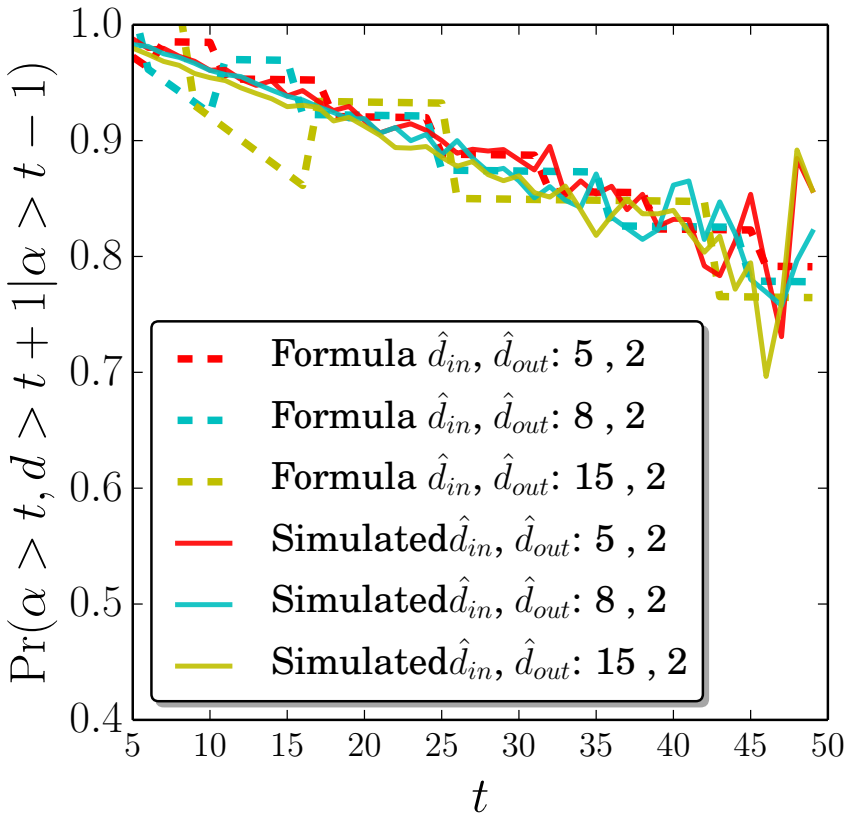


Figure 4.1: The conditional probability (4.14) for NBW on a planted partition model with $n = 100$ on each partition and different average within block degree and compared to the empirical probability for different t

For example for even k:

$$\mathbb{E}(S_{vc}) = \mathbb{E}(\frac{k}{2} + 1)\frac{1}{q^*}$$

where the root of the walk and step t are in the same cluster, e.g. C_1.

In Figure 4.1, we illustrate the found conditional probability (4.14) for NBW on a planted partition model with $n = 100$ on each partition and different average within-block degree with comparison to the empirical probability.

Now we can write the retracing probability in step $t-1$ to t as: Retracing probability for within cluster edges is:

$$\Pr_w(\alpha = t | d > t-1) = 1 - \Pr_w(\alpha > t | d > t-1)$$

$$\Pr_{w_{even}}(\alpha = t | d > t-1) = \frac{\frac{1}{2} + \frac{(p^2+qp)}{q}}{np - 2p + \frac{p(1-p)}{p+q}}$$

$$\Pr_{w_{odd}}(\alpha = t | d > t-1) = \frac{\frac{tp}{2} + \frac{(p^2+qp)}{2q}}{np - 2p + \frac{p(1-p)}{p+q}}$$

Theorem 4.5.1 (Independence for models such as ER and SBM)**.** *Since, for ER and SBM, retracing probability in time t is independent of node and edge locations, we can decouple the joint probability in (4.14) and*

$$\sum_{t=3}^{\infty} Pr(X_t = e_{ij}, \alpha = t | \alpha > t-1) = \sum_{t=3}^{\infty} Pr(X_t = e_{ij}) Pr(\alpha = t | \alpha > t-1)$$

$$\omega_e = \mathbb{E}\left(\sum_{\alpha=3} \mathbb{1}_{X_\alpha = e_{ij}}\right) = \sum_{\alpha=3} \Pr(\textit{edge occupancy at } t = \alpha) \Pr(t = \alpha) \tag{4.15}$$

Theorem 4.5.2. *The probability that the NBRW is retraced between clusters is smaller than reracing between clusters.*

Proof. We have the within block retracing probability for even and odd k respectively

$$\Pr_{w_{even}} = \frac{\frac{1}{2} + \frac{(p^2+qp)}{q}}{np - 2p + \frac{p(1-p)}{p+q}},$$

and

$$\Pr_{w_{odd}} = \frac{\frac{tp}{2} + \frac{(p^2+qp)}{2q}}{np - 2p + \frac{p(1-p)}{p+q}}$$

also, the between block retracing probability is

$$\Pr_{b} = \frac{tq^2 + q^2}{nq + \frac{q(1-q)}{p+q}}.$$

Thus, it suffices to prove the stronger claim by finding quantities A and B such that $\Pr_b < B$ and $A < \Pr_{w_{odd}}, \Pr_{w_{even}}$ and showing $B < A$.

Let $A = \frac{t/2}{n+\frac{1}{p+q}}$ such that:

$$A = \frac{\frac{t}{2}}{n+\frac{1}{p+q}} < \frac{\frac{tp}{2}}{n+\frac{p}{p+q}} < \Pr_{w_{even}}, \Pr_{w_{odd}}$$

Further let $B = \frac{t+1}{n}$ such that:

$$B = \frac{t+1}{n} = \frac{tq+q}{nq-\frac{q}{p+q}+\frac{q}{p+q}} < \frac{tq+q}{nq-\frac{q^2}{p+q}+\frac{q}{p+q}} < \frac{tq^2+q^2}{nq-\frac{q^2}{p+q}+\frac{q}{p+q}} = \Pr_b$$

Now it suffices to show $B < A$; assume, for the sake of contradiction $B > A$, i.e.,

$$\frac{\frac{t}{2}}{n+\frac{1}{p+q}} > \frac{t+1}{n},$$

after simple calculation we get

$$n + \frac{nt}{2} + \frac{t+1}{p+q} < 0.$$

This is a contradiction, therefore, $B < A$ must be true. □

4.5.4 Computing the edge retracing weights

After enough iterations of the RNBRW the edge weights for within block edges become

$$w(e_{ij}) = \sum_{t=3} w^t(e_{ij}) = \sum_{t=3} Pr(X_t = e_{ij}) Pr(\mathbb{1}_{t=\alpha|t-1>\alpha}) =$$
$$\sum_{t=3} \frac{1}{(d_i-1)^t d_s |V|} \frac{tq^2+q^2}{nq+\frac{q(1-q)}{p+q}} =$$

Final weight for between block edges (odd k):

$$w(e_{ij}) = \sum_{t=3} w^t(e_{ij}) = \sum_{t=3} Pr(X_t = e_{ij}) Pr(\mathbb{1}_{t=\alpha|t-1>\alpha}) = \sum_{t=3} \frac{1}{(d_i-1)^t d_s |V|} \frac{\frac{tp}{2} + \frac{(p^2+qp)}{2q}}{np - 2p + \frac{p(1-p)}{p+q}}$$

Final weight for between block edges (even k):

$$w(e_{ij}) = \sum_{t=3} w^t(e_{ij}) = \sum_{t=3} Pr(X_t = e_{ij}) Pr(\mathbb{1}_{t=\alpha|t-1>\alpha}) = \sum_{t=3} \frac{1}{(d_i-1)^t d_s |V|} \frac{\frac{1}{2} + \frac{(p^2+qp)}{q}}{np - 2p + \frac{p(1-p)}{p+q}}$$

4.6 Asymptotic properties and consistency

We define dichotomous random variable $Y_{e_{uv}}$ such that:

$$Y_{e_{uv}} = \mathbf{1}_{x_t = e_{uv}} \mathbf{1}_{\alpha = t | d \geq t} = \begin{cases} 1 & \text{if the walk stops at edge } uv \\ 0 & otherwise \end{cases}$$

Lemma 4.6.1 (Hoffeding inequality)**.** *Let $Y_{e_1} \cdots Y_{e_\rho}$ be independent random variables with $\bar{Y_e} = \frac{1}{\rho} \sum_{i=1}^{\rho} Y_e$ as their empirical mean and $a \leq Y_e \leq b$ thus for $\epsilon \geq 0$*[66]*;*

$$P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq 2 \exp(\frac{(2\epsilon^2)\rho^2}{-\sum_{i=1}^{\rho}(b-a)^2})$$

Each run of our algorithm is independent of the other, thus this parallelization results on having $Y_{e_{uv}}$ as independent random variables (each $Y_{e_{uv}}$ is defined on each walk). Because of the Bernoulli nature of our random variable, it is bounded by 0 and 1. Thus, using Hoeffeding inequality results:

$$P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq 2\exp(-2\rho\epsilon^2), \bar{Y_e} = \frac{1}{\rho}\sum_{i=1}^{\rho} Y_e$$

Because Y_e's are not identically distributed, we need to show for each edge

$$\lim_{\rho\to\infty} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon, i.o) = 0$$

then $\bar{Y_e} \to E[Y_e]$ almost surely. Hence, it is enough to show:

$$\lim_{\rho\to\infty} \max P(|\bar{Y_e} - E[Y_e]| \geq \epsilon, i.o) = 0$$

Also,

$$P(\max|\bar{Y_e} - E[Y_e]|) \leq P(\sum_{e\in E}|\bar{Y_e} - E[Y_e]|) \leq \sum_{e\in E} P(|\bar{Y_e} - E[Y_e]|)$$

then;

$$P(\max|\bar{Y_e} - E[Y_e]| \geq \epsilon, i.o) \leq \sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon, i.o)$$

Using the result of Hoeffding inequality for Bernoulli random variables;

$$\sum_{1}^{\rho}\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq \sum_{1}^{\rho}\sum_{e\in E} 2\exp(-2\rho\epsilon^2)$$

$$\sum_{1}^{\rho}\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq \sum_{1}^{\rho} 2m\exp(-2\rho\epsilon^2)$$

$$\lim\sum_{1}^{\rho}\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq \lim\sum_{1}^{\rho} 2m\exp(-2\rho\epsilon^2) < \infty$$

Using of Borel-Cantelli Lemma;

$$\lim_{\rho\to\infty} \max P(|\bar{Y_e} - E[Y_e]| \geq \epsilon, i.o) = 0$$

Hence, for every edge, $\bar{Y_e}$ converges almost surely to $E[Y_e]$.

Theorem 4.6.2. *(Sufficient number of iterations) As the number of edges increases, we are able to obtain a satisfactory result by only $C \log m$ iterations in average for each edge, where C is a positive constant.*

Proof. Since $E[Y_e] < \infty$ and $E[Y_e] = E(\mathbf{1}_{x_t=e_{uv}} \mathbf{1}_{\alpha=t|d\geq t}) = p(x_t = e_{uv}, \alpha = t|d \geq t) < \infty$

$$P(\max |\bar{Y_e} - E[Y_e]|) \leq P(\sum_{e\in E} |\bar{Y_e} - E[Y_e]|) \leq \sum_{e\in E} P(|\bar{Y_e} - E[Y_e]|)$$

Using the result of Hoeffding inequality for Bernoulli random variable;

$$\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq \sum_{e\in E} 2\exp(-2\rho\epsilon^2)$$

$$\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) \leq 2m\exp(-2\rho\epsilon^2)$$

Let us assume for $\epsilon_2 > 0$, we want to choose ρ such that:

$$\sum_{e\in E} P(|\bar{Y_e} - E[Y_e]| \geq \epsilon) < \epsilon_2,$$

and

$$2m\exp(-2\rho\epsilon^2) < \epsilon_2,$$

then

$$\rho > \frac{\log 2m/\epsilon_2}{2\epsilon^2}.$$

Since ϵ and ϵ_2 are fixed and > 0, then $\rho > C\log(m)$, where C is a constant. □

Chapter 5

Conclusions and future work

Communities can be identified by the "richness" of cycles; more short cycles occur within a community than across communities. Hence, existing community detection algorithms may be enhanced by incorporating information about the participation of edges in rhe cyclic structure of the graph. We investigate how renewal non-backtracking random walks (RNBRW) can improve popular community detection algorithms. RNBRW quantifies edge importance as the likelihood of an edge completing a cycle in a non-backtracking random walk, providing a scalable alternative to analyse real-world networks. We show that weighting the graph with RNBRW can substantially improve the detection of ground-truth communities even in the sparse networks. Furthermore, these preprocessing steps can overcome the problem of detecting small communities known as resolution limit in modularity maximization methods. Finally, in the case of large, sparse, networks, RNBRW has been shown to be quite effective; RNBRW can improve the efficacy of available community detection algorithms without sacrificing their computational efficiency.

Motivated by minimum expected overlap problem on the all cyclic topologies of graph where we minimize the overlap of all cycles using the optimal allocation of edge usage across them. These usages represent the role of edges in the cyclic structures, we introduce RNBRW to quantify the edge importances as the likelihood of an edge completing a cycle in a non-backtracking random walk. Furthermore, by switching our decision variables from cycles to

edges we provide a scalable alternative to both MECO and LM.

The renewal non-backtracking random walk (RNBRW) is a variant of a random walk in which the walk is prohibited from returning back to a node in exactly two steps and terminates and restarts once it completes a loopas a way of quantifying this cyclic structure. Specifically, we propose using the retracing probability of an edgethe likelihood that the edge completes a cycle in an RNBRW–as a way of quantifying cyclic structure. Intuitively, edges with larger retracing probabilities should be more important to the formation of cycles, and hence, to the detection of communities.

The independent nature of edge retracings allows deploying large swarm of walkers in parallel and collecting the retracing information to estimate the edge roles in the cycles efficiently. We give simulation results that suggest pre-weighting edges through RNBRW can improve the performance of popular community detection algorithms substantially. This improvement is most significant when the network is sparse. Moreover, the performance of community detection algorithms with this weighting are competitive to other scalable methods that do not allow for weighting of edges. Furthermore, these preprocessing steps can overcome the problem of resolution limit in modularity maximization methods. Finally, in the case of large, sparse, networks, RNBRW has been shown to be quite effective; RNBRW can improve the efficacy of available community detection algorithms without sacrificing their computational efficiency.

We develop a goodness-of-fit test to infer the existence of modular structures in networks. We begin with a given network that follows an unknown random graph model, we test the null hypothesis that the network is a realization of a null-graph or an Erdős-Rényi graph. Rejecting this null implies that the network comes from a distribution with inherent community structure (for example, a planted partition model). To perform our test, we form an $n \times n$ matrix where entries are the retracing probabilities (estimated through RNBRW) of the corresponding edges of the network. We use a scaled version of the largest eigenvalue of this matrix as our test statistic. We perform a simulation study to compare the Type I Error probability and power of our method to that of other spectral approaches for network inference. We conclude by describing connections between RNBRW and the maximum ex-

pected cyclic overlap problem and giving theoretical results of RNBRW under the stochastic block model.

Further investigation is required to answer multiple conjectures that we develop during this work, for example, proving the independence of the approximated retracing weights and theoretical proof for the distribution of test statistics for correlated weight matrix. We follow assumptions in Chapter 4 that makes our proofs serve as upperbound and better results can be obtained with less simplified assumptions.

In this dissertation, we approximate the role of each edges in the cyclic structures but we can do a similar approach for other topological structures and graph objects such as min-cuts, flows, trees, etc.

Bibliography

[1] M Rosvall and CT Bergstrom. Maps of information flow reveal community structure in complex networks. Technical report, Technical report, 2007.

[2] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[3] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

[4] Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(1):016110, 2006.

[5] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In *International Symposium on Computer and Information Sciences*, pages 284–293. Springer, 2005.

[6] Mark EJ Newman. The structure and function of complex networks. *SIAM review*, 45 (2):167–256, 2003.

[7] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[8] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, 2008.

[9] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[10] Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107, 2011.

[11] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007.

[12] Alex Arenas, Alberto Fernandez, and Sergio Gomez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, 2008.

[13] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[14] Mark S Granovetter. The strength of weak ties. *American journal of sociology*, 78(6): 1360–1380, 1973.

[15] Marek Ciglan, Michal Laclavík, and Kjetil Nørvåg. On community detection in real-world networks and the importance of degree assortativity. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1007–1015. ACM, 2013.

[16] A Meligy, Ahmed H Samak, and Mai E Saad. A new pre-processing strategy for improving community detection algorithms. *International Journal of Computer Applications*, 119(16), 2015.

[17] Santo Fortunato, Vito Latora, and Massimo Marchiori. Method to find community structures based on information centrality. *Physical review E*, 70(5):056104, 2004.

[18] Alireza Khadivi, Ali Ajdari Rad, and Martin Hasler. Network community-detection enhancement by proper weighting. *Physical Review E*, 83(4):046104, 2011.

[19] Haiyan Zhang, Chenxi Zhou, Xun Liang, Xi Zhao, and Yaping Li. A novel edge weighting method to enhance network community detection. In *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*, pages 167–172. IEEE, 2015.

[20] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM, 2002.

[21] Roger Guimera and Luis A Nunes Amaral. Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, 2005.

[22] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[23] I Vragović and E Louis. Network community structure and loop coefficient method. *Physical Review E*, 74(1):016105, 2006.

[24] Hyun-Joo Kim and Jin Min Kim. Cyclic topology in complex networks. *Physical Review E*, 72(3):036109, 2005.

[25] Filippo Radicchi, Claudio Castellano, Federico Cecconi, Vittorio Loreto, and Domenico Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663, 2004.

[26] Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[27] Peng Zhang, Jinliang Wang, Xiaojia Li, Menghui Li, Zengru Di, and Ying Fan. Clustering coefficient and community structure of bipartite networks. *Physica A: Statistical Mechanics and its Applications*, 387(27):6869–6875, 2008.

[28] Heman Shakeri, Pietro Poggi-Corradini, Nathan Albin, and Caterina Scoglio. Network clustering and community detection using modulus of families of loops. *Physical Review E*, 95(1):012316, 2017.

[29] Barry D Hughes. Random walks and random environments. *Oxford*, 2:1995–1996, 1995.

[30] Darong Lai, Hongtao Lu, and Christine Nardini. Enhanced modularity-based community detection by random walk network preprocessing. *Physical Review E*, 81(6):066118, 2010.

[31] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. Enhancing community detection using a network weighting strategy. *Information Sciences*, 222: 648–668, 2013.

[32] Jing Lei et al. A goodness-of-fit test for stochastic block models. *The Annals of Statistics*, 44(1):401–424, 2016.

[33] Peter J Bickel and Purnamrita Sarkar. Hypothesis testing for automated community detection in networks. *arXiv preprint arXiv:1311.2694*, 2013.

[34] Vishesh Karwa, Debdeep Pati, Sonja Petrović, Liam Solus, Nikita Alexeev, Mateja Raič, Dane Wilburne, Robert Williams, and Bowei Yan. Exact tests for stochastic block models. *arXiv preprint arXiv:1612.06040*, 2016.

[35] Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Community extraction for social networks. *Proceedings of the National Academy of Sciences*, 108(18):7321–7326, 2011.

[36] Benjamin Golub and Matthew O Jackson. Does homophily predict consensus times? testing a model of network structure via a dynamic process. *Review of Network Economics*, 11(3), 2012.

[37] Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. Non-backtracking random walks mix faster. *Communications in Contemporary Mathematics*, 9(04):585–603, 2007.

[38] Robert Fitzner and Remco van der Hofstad. Non-backtracking random walk. *Journal of Statistical Physics*, 150(2):264–284, 2013.

[39] John Scott. *Social network analysis*. Sage, 2017.

[40] Sun Sook Chung, Alessandro Pandini, Alessia Annibale, Anthony CC Coolen, N Shaun B Thomas, and Franca Fraternali. Bridging topological and functional information in protein interaction networks by short loops profiling. *Scientific reports*, 5, 2015.

[41] Bruce Hendrickson and Tamara G Kolda. Graph partitioning models for parallel computing. *Parallel computing*, 26(12):1519–1534, 2000.

[42] Santo Fortunato. Community detection in graphs. *Physics reports*, 486(3):75–174, 2010.

[43] Peng Gang Sun. Weighting links based on edge centrality for community detection. *Physica A: Statistical Mechanics and Its Applications*, 394:346–357, 2014.

[44] Christine Klymko, David Gleich, and Tamara G Kolda. Using triangles to improve community detection in directed networks. *arXiv preprint arXiv:1404.5874*, 2014.

[45] Claudio Castellano, Fabio Cecconi, Vittorio Loreto, Domenico Parisi, and Filippo Radicchi. Self-contained algorithms to detect communities in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):311–319, 2004.

[46] Linton C Freeman. A set of measures of centrality based on betweenness. *Sociometry*, pages 35–41, 1977.

[47] Ulrik Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001.

[48] Nathan Albin and Pietro Poggi-Corradini. Minimal subfamilies and the probabilistic interpretation for modulus on graphs. *The Journal of Analysis*, 24(2):183–208, 2016.

[49] Nathan Albin, Faryad Darabi Sahneh, Max Goering, and Pietro Poggi-Corradini. Modulus of families of walks on graphs, 01 2017.

[50] Jason Clemens. *Spanning tree modulus: deflation and a hierarchical graph structure.* PhD thesis, 2018.

[51] Mark Kempton. Non-backtracking random walks and a weighted ihara's theorem. *arXiv preprint arXiv:1603.05553*, 2016.

[52] Mark Kempton. *High Dimensional Spectral Graph Theory and Non-backtracking Random Walks on Graphs*. University of California, San Diego, 2015.

[53] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[54] Leon Danon, Albert Diaz-Guilera, Jordi Duch, and Alex Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005 (09):P09008, 2005.

[55] Yunpeng Zhao, Elizaveta Levina, Ji Zhu, et al. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, 40 (4):2266–2292, 2012.

[56] Karl Rohe, Sourav Chatterjee, Bin Yu, et al. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

[57] Behnaz Moradi, Heman Shakeri, Pietro Poggi-Corradini, and Michael Higgins. New methods for incorporating network cyclic structures to improve community detection. *arXiv preprint arXiv:1805.07484*, 2018.

[58] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[59] Craig A. Tracy and Harold Widom. On orthogonal and symplectic matrix ensembles. *Communications in Mathematical Physics*, 177(3):727–754, Apr 1996. ISSN 1432-0916. doi: 10.1007/BF02099545. URL https://doi.org/10.1007/BF02099545.

[60] Ji Oon Lee and Jun Yin. A necessary and sufficient condition for edge universality of wigner matrices. *Duke Math. J.*, 163(1):117–173, 01 2014. doi: 10.1215/00127094-2414767. URL https://doi.org/10.1215/00127094-2414767.

[61] Prasad Tetali. Random walks and the effective resistance of networks. *Journal of Theoretical Probability*, 4(1):101–109, 1991.

[62] Ido Tishby, Ofer Biham, and Eytan Katzav. The distribution of first hitting times of non-backtracking random walks on erdős–rényi networks. *Journal of Physics A: Mathematical and Theoretical*, 50(20):205003, 2017.

[63] Behnaz Moradi, Heman Shakeri, Pietro Poggi-Corradini, and Michael Higgins. New methods for incorporating network cyclic structures to improve community detection. *CoRR*, abs/1805.07484, 2018. URL http://arxiv.org/abs/1805.07484.

[64] Gregory F Lawler and Vlada Limic. *Random walk: a modern introduction*, volume 123. Cambridge University Press, 2010.

[65] Ido Tishby, Ofer Biham, and Eytan Katzav. The distribution of path lengths of self avoiding walks on erdős–rényi networks. *Journal of Physics A: Mathematical and Theoretical*, 49(28):285002, 2016.

[66] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

Appendix A

RNBRW source code

```
1
2  def walk_hole(G, Niter):
3      """
4      Random walk that traps at a hole and weights the terminal edge
5      """
6      m = 0
7      for a,b in G.edges_iter():
8          G[a][b]['enum'] = m
9          m += 1
10
11     m = G.number_of_edges()
12     T = np.zeros(m, dtype= int)
13
14     E =G.edges()
15     E_reversed = [(y,x) for x,y in G.edges() ]
16     E_directed = E+E_reversed
17
18     for k in range(Niter):
19         E_sampled = [E_directed[i] for i in np.random.choice(range(2*m)
               , 1)]
20         u = E_sampled[0][0]
```

```
21          v= E_sampled[0][1]
22          walk = [u,v]
23          while True:
24              nexts = G.neighbors(v)
25              try:
26                  nexts.remove(u)  #removing the walk head, for the next
                        step
27              except ValueError:  #check if head is  in the next step.
                    Avoid problem in the start
28                  pass
29
30              if nexts == []:#if no hole found
31                  break
32
33              next = np.random.choice(nexts) #next step of the walker
34              if next in walk: #if encounters a hole
35                  T[  G[v][next]['enum']  ] += 1
36                  break
37
38              walk.append(next) #still no hole, prepare for the next step
39              u = v
40              v = next
41      return T
42
43
44
45  def cycle_prop(G, nsim = 20, parallel = True):
46      """
47      Repeating walk_hole nsim times\n",
48      """
49      m = G.number_of_edges()
50      Niter = m
51      T = np.zeros(m, dtype= int)
```

```
52      for i in range(nsim):
53          T += walk_hole(G, Niter)
54
55
56      m = 0
57      for a,b in G.edges_iter():
58          G[a][b]['ret'] = T[m]
59          m += 1
60      return G
```