FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

LOW-RANK RIEMANNIAN OPTIMIZATION APPROACH TO THE ROLE EXTRACTION PROBLEM

By

MELISSA SUE MARCHAND

A Dissertation submitted to the
Department of Mathematics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

2017

Melissa Sue Marchand defended this dissertation on September 21, 2017.
The members of the supervisory committee were:

Kyle A. Gallivan

Professor Co-Directing Dissertation

Paul Van Dooren

Professor Co-Directing Dissertation

Gordon Erlebacher

University Representative

Giray Okten

Committee Member

Mark Sussman

Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.

This work is dedicated to my parents and family, for their encouragement and support.

ACKNOWLEDGMENTS

I would like to acknowledge the people whose help made this dissertation possible.

First, I would like to thank my advisor, Professor Kyle A. Gallivan, for his constant guidance and support throughout my graduate research. His expertise and knowledge in optimization and numerical linear algebra has been an invaluable resource to me during my time at Florida State University. His guidance has improved my research skills and prepared me for future challenges. I would also like to thank my co-advisor Professor Paul Van Dooren. His detailed and delicate instructions have helped me a lot throughout this entire process. Next, I would like to thank my committee members, Professor Gordon Erlebacher, Professor Giray Okten, and Professor Mark Sussman, for their contributions and counsel on this dissertation.

I also give my appreciation to my friends and church family for their support during the process of pursuing my Ph.D. They have been my family away from home and I will never forget their words of encouragement during times of great stress.

Most importantly, I am grateful to my family. They have encouraged me, supported me, fed me, and helped me maintain my sanity as I have completed my dissertation. I would never have been able to have done this without them.

Finally, I appreciate the financial support from NSF Grant 1262476.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ABSTRACT

This dissertation uses Riemannian optimization theory to increase our understanding of the role extraction problem and algorithms. Recent ideas of using the low-rank projection of the neighborhood pattern similarity measure and our theoretical analysis of the relationship between the rank of the similarity measure and the number of roles in the graph motivates our proposal to use Riemannian optimization to compute a low-rank approximation of the similarity measure.

We propose two indirect approaches to use to solve the role extraction problem. The first uses the standard two-phase process. For the first phase, we propose using Riemannian optimization to compute a low-rank approximation of the similarity of the graph, and for the second phase using k-means clustering on the low-rank factor of the similarity matrix to extract the role partition of the graph. This approach is designed to be efficient in time and space complexity while still being able to extract good quality role partitions. We use basic experiments and applications to illustrate the time, robustness, and quality of our two-phase indirect role extraction approach.

The second indirect approach we propose combines the two phases of our first approach into a one-phase approach that iteratively approximates the low-rank similarity matrix, extracts the role partition of the graph, and updates the rank of the similarity matrix. We show that the use of Riemannian rank-adaptive techniques when computing the low-rank similarity matrix improves robustness of the clustering algorithm.

CHAPTER 1

INTRODUCTION

This dissertation investigates solving the role extraction problem for graphs. There are two basic classes of methods to extract roles from a graph: direct and indirect. The properties and differences of the two classes are discussed in detail in Chapter 2 and this dissertation concentrates on indirect methods. A typical indirect method is a two-phase process, where the first phase is to compute the similarity of the network and the second phase is to extract the role structure of the network from the similarity matrix by grouping highly similar nodes together. For our two-phase indirect method, the first phase uses Riemannian optimization to compute a low-rank approximation of a neighborhood pattern similarity measure to determine a similarity score for each pair of nodes in the graph. In the second phase, we use k-means clustering on the low-rank factor of the similarity matrix. This approach is designed to be efficient in time and space complexity for large networks.

Several different similarity measures have been discussed in the literature and the choice of the measure for the role extraction problem is crucial. Given the choice of similarity metric, a drawback of indirect methods is the inability to adjust the choice of parameters based on the results of both phases. If a good similarity metric for the network or appropriate parameter values are not used, then the clustering algorithm in the second phase struggles to extract the role structure. Thus, our second indirect approach combines the two phases of the indirect method into a single, low-rank iterative process that allows for effective adaptation of the parameters associated with the similarity metric and the clustering algorithm. This approach exploits current methods of determining the low-rank approximation of a matrix to develop a Riemannian optimization algorithm that computes the numerical rank of the neighborhood pattern similarity measure and the optimal number of roles using k-means clustering and the silhouette statistic. The list of assumed roles to test for the silhouette statistic is determined by the rank and an assumed number of roles, which is adjusted until convergence.

The organization of this dissertation is as follows. In Chapter 1, we define the role extraction problem. In addition, we review basic concepts and definitions in both graph theory and Rieman-

nian optimization used throughout this dissertation. The chapter ends with an overview of our research and dissertation statement. Chapter 2 is an overview of the direct and indirect approaches to solving the role extraction problem. For the indirect approaches, we review four pairwise node self-similarity measures. Also, we summarize three community detection (or clustering) algorithms that can be used for both direct and indirect approaches. Chapter 3 analyzes the relationship between the number of roles and the rank of the similarity matrix for the neighborhood pattern similarity measure, which is the similarity metric we use in this dissertation.

Chapter 4 describes our two-phase approach. In this chapter, we derive: the Riemannian geometry for the symmetric positive semidefinite fixed-rank manifold; a cost function for the Riemannian optimization approach to approximating the low-rank neighborhood pattern similarity measure; and the Riemannian gradient and action of the Riemannian Hessian. Finally, in this chapter, we describe the motivation behind using k-means clustering for the second phase of our role extraction approach. In Chapter 5, we rework Browet and Van Dooren's iterative algorithms for computing the neighborhood pattern similarity measure in the Euclidean optimization framework based on our cost function. Chapter 6 empirically evaluates the strengths and weaknesses of Browet and Van Dooren's two-phase role extraction approach and our proposed two-phase approach. In Chapter 7, we describe our one-phase role extraction method, and provide evidence about its robustness compared to our two-phase approach.

In Chapter 8, we observe the effectiveness of our indirect approach on signed networks. Then, in Chapter 9, we compare role structures with overlapping community structures and show a relationship between the two. Lastly, in Chapter 10, we summarize our research and describe some interesting open questions for efficient and effective role extraction.

1.1 Motivation and Problem

Many complex systems can be represented as network structures, e.g., human interactions, food webs, and gene interactions. Recent work has focused on the extraction of clusters to analyze large networks and obtain relevant statistical properties. Various measures and algorithms to identify community structures, i.e., subgroups of densely connected nodes [POM09,For10,Tra14]. However, this structural distribution of nodes in a graph is not always representative. For example, bipartite and cyclic graphs do not contain communities, even though they are structured. Less attention

has been paid to discovering more general structure by role extraction or as it is sometimes called block modeling [WF94, DBF05, Rei09, Cas12].

The role extraction problem determines a representation of a network by a smaller structured graph, called the reduced graph, role graph, or image graph, where nodes are grouped together in roles based upon their interactions with nodes in either the same role or different roles. This problem is a generalization of the community detection problem where each node in a community mainly interacts with other nodes within the same community. Many other role interactions can be defined, such as a bipartite graph for human protein-protein interaction networks [PSR10] or a block cycle model for food webs [GSSP$^+$10]. There are many real world applications to which role extraction can be applied and from which characterizations of interactions that define roles can be taken. These include studying trade networks between countries [RW07]; evaluating the resilience of peer-to-peer networks [HKYH02]; ranking web pages in search engines [PBMW99]; studying human interaction by email correspondence [AMC10]; modeling protein-protein interactions [KKKR02]; and analyzing food webs [GSSP$^+$10].

The role extraction problem takes an adjacency matrix A of a weighted and directed graph that represents the network to be analyzed. A representative role structure is determined and is specified by a permutation matrix P, such that the edges of the relabeled graph, the adjacency matrix PAP^T, are primarily concentrated in particular blocks (see Figure 1.1).

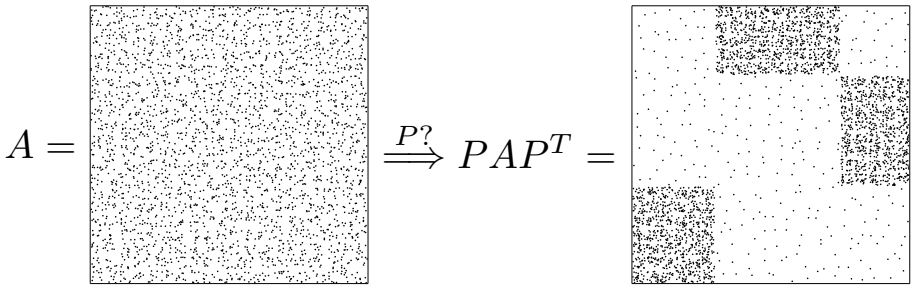


Figure 1.1: Block modeling example: find permutation matrix P such that the adjacency matrix A is permuted to have a block structure.

The role extraction problem is based on the assumption that nodes can be grouped according to a suitable measure of equivalence. Lorrain and White [LW71] introduced the first relation of equivalence between nodes. Two nodes are structurally equivalent if they have exactly the

same neighbors. As a result, all blocks in the permuted adjacency matrix must either be null or complete [DBF05]. Null blocks occur when there are no edges connecting a node in one role to a node in the other role, i.e.,

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

and complete blocks when each node in a role is connected to all nodes of the other role, i.e.,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

When applied to real graphs, structural equivalence tends to extract many small roles and so structural equivalence was relaxed to regular equivalence defined in [WR83,EB94,EB96]. Two nodes are considered to be regularly equivalent if they are connected to the same equivalence classes while the number of connections is unimportant. That is, the nodes are regularly equivalent if, while they do not necessarily share the same neighbors, they have the neighbors who are themselves structurally or regularly equivalent. Therefore, the blocks in the permuted adjacency matrix must contain at least one element per row and column (called a regular block) [DBF05], e.g.,

$$\begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix}.$$

Two other types of regular blocks are row-regular blocks (where there is at least one 1 in each row) and column-regular blocks (where there is at least one 1 in each column). Note that a regular block is also row-regular and column-regular, but a row-regular (column-regular) block is not necessarily regular.

Structural equivalence implies regular equivalence, but regular equivalence does not imply structural equivalence. So, every group of regularly equivalent nodes is represented by a single node in the role graph where nodes in the role graph are linked (or not linked) if connections between nodes in the respective classes exist (or are absent) in the original graph. Reichardt and White assumed no two nodes in the role graph may be structurally equivalent to avoid redundancy in the role graph [RW07, Rei09].

Earlier research involved creating a quality function to optimize over both the role structure and role assignment of nodes in the graph based on a choice of equivalence relation [WF94, DBF05,

RW07, Rei09]. That is, if B is the adjacency matrix of the reduced graph (called the *image matrix*) and σ is the assignment of each node to a role, then the problem can be stated as

$$(B^*, \sigma^*) = \arg\max_{B,\sigma} \mathcal{Q}_A(B, \sigma), \tag{1.1}$$

where $\mathcal{Q}_A$ depends on the graph topology and chosen equivalence criterion. Note that (1.1) is a combinatorial optimization problem with respect to two variables and is generally harder than the community detection problem, which is in general NP-hard [BDG$^+$06, BDG$^+$08]. The quality function $\mathcal{Q}_A(B, \sigma)$ can either be constructed indirectly, based on a (dis)similarity measure between pairs of nodes, or directly, based on a measuring of the fit of clusters compared to an ideal clustering with perfect relations within and between clusters. In Chapter 2, we describe how the quality function is constructed for each approach.

1.2 Review of Basic Concepts of Graph Theory

Throughout this dissertation, we consider graph theory and its application to the role extraction problem. Therefore, we define some graph notation and review some basic definitions to characterize graph properties. In addition, we describe some graph structures. Further background information on graph theory can be found in [GY05, BM08, New10].

1.2.1 Basic Definitions

A *graph*, denoted $G(V, E)$, is a mathematical structure with two finite sets V and E where the elements of the set $V = \{1, \ldots, n\}$ are called *nodes* (or *vertices*) and the elements of the set $E = \{(i, j) \mid i, j \in V\}$ are called *edges* (or *links*) [GY05]. A pair (i, j) belongs to E if nodes i and j are connected. An edge joins vertices i and j, which means that i is a *neighbor* of j. The cardinality of the set E (or the number of edges in the graph) is denoted by $|E| = m$ [GY05, BM08].

A *self-loop* (or *self-edge*) is an edge that connects a node i to itself, i.e., $(i, i) \in E$. A *multi-edge* is two or more edges between nodes i and j. A *simple graph* has neither self-loops or multi-edges, while a *multi-graph* has multi-edges, but does not have self-loops. A *(general) unweighted graph* may have self-loops and/ or multi-edges [GY05, New10]. In this dissertation, we do not consider multi-edges for unweighted graphs.

An unweighted simple graph can be represented by an $n \times n$ $\{0,1\}$-matrix A, called the *(unweighted) adjacency matrix* [New10], where

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

A graph associated with the adjacency matrix is denoted $G_A(V,E)$. The adjacency matrix is important throughout this dissertation because it allows us to analyze graphs using matrix theory and linear algebra.

Graphs with negative edges are called *signed graphs* [Zas82, DBF05, TKVD13]. A signed graph is denoted $G_A(V, E^-, E^+)$, where $E^- \subseteq V \times V$ are the negative edges, $E^+ \subseteq V \times V$ are the positive edges, and no edge can be both positive and negative (i.e., $E^- \cap E^+ = \emptyset$) [DBF05]. Then, a signed adjacency matrix is defined as

$$A_{i,j} = \begin{cases} -1, & \text{if } (i,j) \in E^-, \\ 1, & \text{if } (i,j) \in E^+, \\ 0, & \text{otherwise.} \end{cases}$$

A graph is called *undirected* if each edge $(i,j) \in E$ is unordered and $(i,j) = (j,i)$. Thus, the adjacency matrix of an undirected graph is symmetric, i.e., $A = A^T$ [BM08]. Also, self-loops are represented by a 2 in the adjacency matrix because every self-loop (i,i) has two ends which are both connected to node i [New10].

A *directed* graph (or *digraph*) is a graph where each edge (i,j) has a direction, called *directed edges* (or *arcs*), such that node i is the source and node j is the destination [BM08, New10]. Also, for directed graphs, a neighbor j of a node i is called a *child* when $(i,j) \in E$ and a *parent* when $(j,i) \in E$. Note that the adjacency matrix of a directed graph is not symmetric. Also, self-loops are represented by a 1 in the adjacency matrix [New10]. For the role extraction problem, unless stated otherwise, we only consider directed graphs, and if the graph is undirected, then we assume that the graph is *bidirectional*, i.e., an undirected edge between nodes i and j is replaced by two directed edges in opposite directions between the same nodes [New10].

The *degree* of a vertex i is the number of neighbors the vertex has and is denoted by k_i [New10]. For directed graphs, there is a distinction between the *in-degree* k_i^{in} neighbors and the *out-degree* k_i^{out} neighbors [New10]. These are referred to as the number of parents and the number of children, respectively. The vectors of the number of in-degree and out-degree neighbors can be computed as

$$k^{out} = A\mathbf{1} \quad , \quad k^{in} = A^T\mathbf{1},$$

where $\mathbf{1}$ is a vector of all 1's of length n and A is an $n \times n$ adjacency matrix [New10].

In practice, the edges in a graph may be weighted representing the intensity of the interaction between the nodes. That is, for a pair $(i, j) \in E$, there is an associated real number $w_{i,j}$ called the *weight* [New10]. The graph $G(V, E)$ together with weights on its edges is called a *weighted graph* and is denoted $G_W(V, E)$, where the subscript W denotes the *weighted adjacency matrix*. The weighted adjacency matrix is an $n \times n$ real matrix such that $W_{i,j} \neq 0$ if and only if $A_{i,j} = 1$ [New10]. Note that for an undirected graph, the weighted adjacency matrix is not necessarily symmetric. Also, as with the degree of a node, for weighted graphs, the *strength* s_i of node i is defined as the sum of the weights of its neighbors. For a weighted, directed graph, the *in-strength* and *out-strength* is the sum of weights of incoming and outgoing edges, respectively, and are represented by the, respective, vectors

$$s^{out} = W\mathbf{1} \quad , \quad s^{in} = W^T\mathbf{1}.$$

The *density* of a graph is the ratio between the number of edges in a graph and the maximal number of possible edges, i.e., m/n^2 [New10]. A graph is considered *sparse* if the density of the graph is low [New10]. Since the maximal number of edges in a graph is the number of nodes in the graph squared, then for sparse graphs, the number of edges in the graph should grow linearly with the number of nodes, i.e., $m = O(n)$, or the average degree, which is defined as

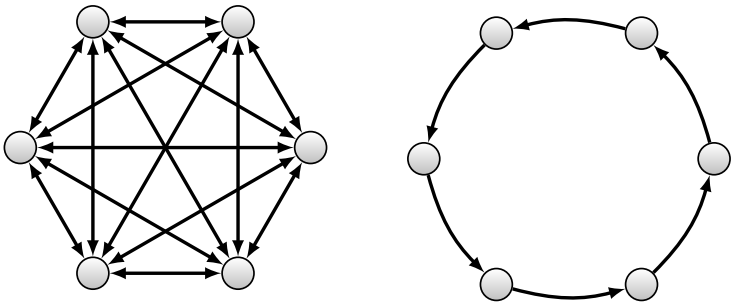
$$\bar{k} = \frac{1}{n}\sum_{i=1}^{n} k_i,$$

should be smaller than the number of nodes $\bar{k} \ll n$ [New10].

1.2.2 Graph Structures

We next define some special families of graphs. A directed graph is *complete* if every pair of nodes is joined by an edge in both directions (possibly excluding self-loops). A complete directed graph over n nodes is denoted K_n and has $n(n-1)$ edges (or n^2 edges if there are self-loops) [Bro14]. For example, Figure 1.2a is a complete graph over 6 nodes.

A *directed path* is a set of undirected edges connecting node i to node j where the destination of each edge in the sequence (except for the last edge) is the source of the following edge. Note that there might exist multiple paths between nodes i and j. A *simple* directed path is when each edge in the path is used at most once. A *cycle* is a simple path where the origin and the



(a) Complete Graph K_6 (b) Cycle Graph C_6

Figure 1.2: Example of a complete graph and a cycle graph

destination are the same node (see Figure 1.2b). The *length* of a path or cycle is the number of edges it contains [BM08, New10].

A *walk* is a simple directed path where all of the nodes are distinct. A *semiwalk* is a sequence of vertices where the direction of the edge is ignored [BM08].

A *subgraph* $H(V_H, E_H)$ of a graph $G(V, E)$ is a graph whose nodes are a subset of the nodes of G (i.e., $V_H \subset V$) and whose edges are a subset of the edges of G such that $E_H \subset \{(i,j) \mid i,j \in V_H,\ (i,j) \in E\}$ [BM08]. If the edge set E_H contains edges that have both ends in V_H, then the subgraph is called *induced*. A *spanning* graph is a subgraph of G where its node set is the same as G (i.e., $V_H = V$) [BM08].

An undirected graph is *connected* if there exists a path between any pair of nodes [BM08]. If a graph is *disconnected*, then the graph can be divided into multiple induced subgraphs such that every subgraph is connected and there does not exist any edges between any induced subgraph. The induced subgraphs H_i's are called the *connected components* of G [BM08]. If the graph is directed, then there are two types of connected components. The first type of connected components is a *strongly connected component* (SCC), which is a maximal set of nodes such that there exists a directed path between every pair of nodes, where the term *maximal* means that no node can be added to the set while maintaining the property [New10]. The second type of connected components is a *weakly connected component* (WCC). A weakly connected component is a maximal set of nodes such that there exists a path between every pair of nodes and direction is not considered [New10].

Lastly, a *clique* is a subset of nodes in an undirected graph such that its induced subgraph is complete. A clique is called *maximal* if it cannot be extended by adding nodes to it [New10].

1.3 Review of Basic Concepts of Riemannian Optimization

This section reviews basic definitions and concepts of Riemannian manifolds necessary for Riemannian optimization. Additionally, this section characterizes Riemannian optimization algorithms of interest that are used in this dissertation. More details of these concepts and algorithms can be found in [CE75], [O'N83], [Lee97], [AMS08], [Bak08], [Qi11], [Hua13], and [Zho15].

1.3.1 Optimization on a Manifold

Optimization on Riemannian manifolds (also known as Riemannian Optimization) finds an (global or local) optimum of a real-valued function f defined over a Riemannian manifold, i.e.,

$$\min f(x), \quad \text{subject to} \quad x \in \mathcal{M}, \tag{1.2}$$

where $\mathcal{M}$ is a Riemannian manifold [Hua13].

Typically, Riemannian optimization is considered as unconstrained optimization on a constrained space and ideas from unconstrained optimization algorithms on a Euclidean space have been adapted to optimization on manifolds. However, to use these ideas, we must reconsider many basic definitions, constructs and algorithmic techniques, since extending them from the Euclidean space to the manifold is not trivial. For example, the addition and subtraction of two points in the Euclidean space is well-defined but does not extend, in general, to two points on a manifold.

Roughly speaking, a manifold is a set covered with a collection of coordinate patches that overlap smoothly, i.e., manifold is a set of points that is locally Euclidean. Specifically, each point of a d-dimensional manifold has a neighborhood that is homeomorphic to the Euclidean space of dimension d. In Riemannian geometry, a smooth manifold of dimension d is defined as a set $\mathcal{M}$ that locally looks like a d-dimensional Euclidean space but can be different globally. Since optimization usually requires computing derivates and gradients of a function, we require that $\mathcal{M}$ have a smooth structure. A Riemannian manifold is a smooth set with a smoothly-varying inner product on the tangent spaces [AMS08].

1.3.2 Tangent Vector and Tangent Space

To apply optimization algorithms based on line-search methods, we must define the concept of direction on a manifold. Consider a smooth mapping $\gamma : \mathbb{R} \to \mathcal{M} : t \mapsto \gamma(t)$ where $\gamma(t)$ is a curve on $\mathcal{M}$ and satisfies $\gamma(0) = x$. Given a smooth function f on $\mathcal{M}$, the function $f \circ \gamma : t \mapsto f(\gamma(t))$ is a smooth function from $\mathbb{R}$ to $\mathbb{R}$ with a well-defined classical derivative [AMS08]. This approach, combining curves and smooth functions on differentiable manifolds, allows us to define a tangent vector. If $\mathcal{F}_x(\mathcal{M})$ denotes the set of smooth functions defined on a neighborhood of x, then a tangent vector can be defined as follows.

Definition 1.3.1 *(Tangent Vector) [AMS08] A tangent vector ξ_x to a manifold $\mathcal{M}$ at a point x is a mapping from $\mathcal{F}_x(\mathcal{M})$ to $\mathbb{R}$ such that there exist a curve γ on $\mathcal{M}$ with $\gamma(0) = x$, satisfying*

$$\xi_x f = \dot{\gamma}(0) f := \left. \frac{d(f(\gamma(t)))}{dt} \right|_{t=0}$$

for all $f \in \mathcal{F}_x(\mathcal{M})$. Such a curve γ is said to realize the tangent vector ξ_x. The point x is called the foot of the tangent vector ξ_x.

The set of all tangent vectors to $\mathcal{M}$ at x is called the *tangent space* to $\mathcal{M}$ at x, denoted $\mathrm{T}_x\mathcal{M}$ [AMS08]. This is a linear space, i.e., closed under linear combinations, with the same dimension as the manifold. The *tangent bundle*, denoted $\mathrm{T}\mathcal{M}$ is the union of all tangent spaces at all elements of $\mathcal{M}$ [AMS08], i.e.,

$$\mathrm{T}\mathcal{M} := \bigcup_{x \in \mathcal{M}} \mathrm{T}_x\mathcal{M}.$$

The property that a tangent space is a vector space is important because it provides a vector space definition of local motion on the manifold with which we work rather than motion directly on the manifold. However, such local motion must be mapped back to the manifold. A map from the tangent space to the manifold is called a *retraction* and is discussed later.

A *vector field* is a smooth mapping $\xi : \mathcal{M} \to \mathrm{T}\mathcal{M}$ that assigns to each point $x \in \mathcal{M}$ a tangent vector $\xi_x \in \mathrm{T}_x\mathcal{M}$ and the set of all smooth vector fields on $\mathcal{M}$ is denoted by $\chi(\mathcal{M})$ and is endowed with the operations of addition of two vector fields and multiplication of a vector field by a function $f \in \mathcal{F}_x(\mathcal{M})$ [AMS08], i.e., for all $x \in \mathcal{M}$,

$$(f\xi)_x = f(x)\xi_x,$$
$$(\xi + \zeta)_x = \xi_x + \zeta_x.$$

1.3.3 Riemannian Metric

A Riemannian metric computes angles and lengths of directions (tangent vectors) in any tangent space of $\mathcal{M}$. In other words, a *Riemannian metric* g is defined on each tangent space of x as an inner product $g_x : \mathrm{T}_x\mathcal{M} \times \mathrm{T}_x\mathcal{M} \to \mathbb{R}$ that varies smoothly with x and is denoted as

$$g_x(\xi_x, \zeta_x) = \langle \xi_x, \zeta_x \rangle_x$$

where $\xi_x, \zeta_x \in \mathrm{T}_x\mathcal{M}$ [AMS08]. The notation, flat $\flat$, relates tangent vectors to the metric and $\xi_x^\flat$ denotes a function from $\mathrm{T}_x\mathcal{M}$ to $\mathbb{R}$ where $\xi_x^\flat \zeta_x = g_x(\xi_x, \zeta_x)$ for all $\xi_x, \zeta_x \in \mathrm{T}_x\mathcal{M}$ [Hua13]. A *Riemannian manifold* is the combination of $(\mathcal{M}, g)$ [AMS08].

1.3.4 Affine Connection

Many optimization algorithms require second-order information about the cost function. In general, this second-order information is obtained by taking the derivative of one vector field with respect to another. In the Euclidean space, taking the derivative of one vector field along another is called the directional derivative and is defined as

$$D\eta(x)[\xi_x] = \lim_{t \to 0} \frac{\eta(x + t\xi_x) - \eta(x)}{t}. \tag{1.3}$$

This always returns a vector field. However, on a Riemannian manifold, for any two vector fields ξ and η on $\mathcal{M}$, (1.3) need not be a vector field on $\mathcal{M}$ even if all of the operations in (1.3) are well-defined [AMS08]. Thus, on Riemannian manifolds, the concept of taking the directional derivative of a vector field is generalized to the affine connection.

Definition 1.3.2 *(Affine Connection) [AMS08] Let $\mathcal{F}_x(\mathcal{M})$ be the set of all smooth functions in $x \in \mathcal{M}$ and $\chi(\mathcal{M})$ be the set of all smooth vector fields on $\mathcal{M}$. Then the affine connection is a smooth mapping, denoted by*

$$\nabla : \chi(\mathcal{M}) \times \chi(\mathcal{M}) \to \chi(\mathcal{M}) : (\xi, \eta) \mapsto \nabla_\xi \eta$$

that satisfies the following properties: for all $f, g \in \mathcal{F}_x(\mathcal{M})$, $a, b \in \mathbb{R}$, and $\eta, \xi, \zeta \in \chi(\mathcal{M})$,

(i) $\mathcal{F}_x(\mathcal{M})$-linearity in the first argument η: $\nabla_{f\eta g\zeta}\xi = f\nabla_\eta \xi + g\nabla_\zeta \xi$;

(ii) $\mathbb{R}$-linearity in the second argument ξ: $\nabla_\eta(a\xi + b\zeta) = a\nabla_\eta \xi + b\nabla_\eta \zeta$;

(iii) Product rule/ Leibniz's law: $\nabla_\eta(f\xi) = (\eta f)\xi + f\nabla_\eta\xi$.

Note that given a vector field η on $\mathcal{M}$ and a (smooth) real-valued function $f \in \mathcal{F}_x(\mathcal{M})$, ηf denotes the real-valued function on $\mathcal{M}$ defined by

$$(\eta f)(x) := \eta_x f$$

for all $x \in \mathcal{M}$ [AMS08]. For any smooth manifold $\mathcal{M}$, there are an infinite number of affine connections. For a Riemannian manifold $(\mathcal{M}, g)$, there exists a unique affine connection that also satisfies the Levi-Civita conditions [AMS08].

Theorem 1.3.3 *(Levi-Civita) [AMS08] On a Riemannian manifold $(\mathcal{M}, g)$ there exists a unique affine connection ∇ that satisfies, for all $\eta, \xi, \zeta \in \chi(\mathcal{M})$*

(i) symmetry: $\nabla_\eta\xi - \nabla_\xi\eta = [\eta, \xi]$;

(ii) compatibility with the Riemannian metric: $\zeta \langle \eta, \xi \rangle_x = \langle \nabla_\zeta \eta, \xi \rangle_x + \langle \eta, \nabla_\zeta \xi \rangle_x$

This affine connection ∇, called the Levi-Civita connection (or the Riemannian connection) of $\mathcal{M}$, is characterized by the Koszul formula

$$2\langle \nabla_\zeta \eta, \xi \rangle_x = \zeta \langle \eta, \xi \rangle_x + \eta \langle \xi, \zeta \rangle_x - \xi \langle \zeta, \eta \rangle_x - \langle \zeta, [\eta, \xi] \rangle_x + \langle \eta, [\xi, \zeta] \rangle_x + \langle \xi, [\zeta, \eta] \rangle_x . \tag{1.4}$$

1.3.5 Riemannian Gradient and Riemannian Hessian

In the Euclidean space, the gradient of a scalar-valued function is the direction of steepest ascent of the objective function and is very useful in optimization. Since the gradient is a direction on the manifold, it should be a tangent vector. Definition 1.3.4 defines the Riemannian gradient on a Riemannian manifold $(\mathcal{M}, g)$.

Definition 1.3.4 *(Riemannian Gradient) [AMS08] Let f be a function defined on a Riemannian manifold $(\mathcal{M}, g)$. The Riemannian gradient of f at x, denoted as* $\operatorname{grad} f(x)$*, is the unique tangent vector in $\mathrm{T}_x\mathcal{M}$ satisfying*

$$\langle \operatorname{grad} f(x), \xi_x \rangle_x = Df(x)[\xi_x], \quad \forall \xi_x \in \mathrm{T}_x\mathcal{M}, \tag{1.5}$$

where the directional derivative is denoted Df and the definition of a tangent vector identifies $Df(x)[\xi_x] = \xi_x f$.

Newton's method requires second-order information of the Hessian. For a Euclidean function, the Hessian is the second derivative of the objective function and contains the information of differentiating the gradient along some direction. This notion can be used to defined the Riemannian Hessian.

Definition 1.3.5 *(Riemannian Hessian) [AMS08] Given a function f on a Riemannian manifold $(\mathcal{M}, g)$, the Riemannian Hessian of f at a point x is the linear mapping of* $\operatorname{Hess} f(x)$ *from* $\mathrm{T}_x\mathcal{M}$ *to* $\mathrm{T}_x\mathcal{M}$ *defined by*

$$\operatorname{Hess} f(x)[\eta_x] = \nabla_{\eta_x} \operatorname{grad} f(x), \tag{1.6}$$

for all $\eta_x \in \mathrm{T}_x\mathcal{M}$, where ∇ is the Riemannian connection on $\mathcal{M}$.

From the symmetric property of the Riemannian connection, we know that the Hessian is a symmetric operator with respect to the Riemannian metric [AMS08], i.e.,

$$\langle \operatorname{Hess} f(x)[\eta_x], \xi_x \rangle_x = \langle \eta_x, \operatorname{Hess} f(x)[\xi_x] \rangle_x ,$$

for all $\eta_x, \xi_x \in \mathrm{T}_x\mathcal{M}$.

1.3.6 Retraction and Vector Transport

In general, most optimization algorithms work in the tangent space of the current iterate to find a tangent vector to define the next iterate on the manifold. So, we need a way to map the chosen tangent vector to the next iterate, which is called a retraction [AMS08].

In addition to mapping a tangent vector from the tangent space to the manifold, a retraction can also transform cost functions defined in a neighborhood of $x \in \mathcal{M}$ into cost functions defined on the vector space $\mathrm{T}_x\mathcal{M}$. In other words, given a function f on a manifold $\mathcal{M}$ equipped with a retraction R, we let $\hat{f} = f \circ R$ denote the pullback of f through R [AMS08], i.e., for $x \in \mathcal{M}$,

$$\hat{f}_x : \mathrm{T}_x\mathcal{M} \to \mathbb{R} : \eta_x \mapsto f(R_x(\eta_x)). \tag{1.7}$$

Some algorithms need to combine information at different iterates to determine the next search direction. A vector transport is a way to map a tangent vector from one tangent space to another and it is built upon the retraction [AMS08].

A vector transport is called isometric if it also satisfies [Hua13]

$$g_{R(\eta_x)}(\mathcal{T}_{\eta_x}(\xi_x), \mathcal{T}_{\eta_x}(\zeta_x)) = g_x(\xi_x, \zeta_x). \tag{1.8}$$

Vector transport by differentiated retraction is a vector transport given by

$$\mathcal{T}_{\eta_x}(\xi_x) := DR_x(\eta_x)[\xi_x] = \frac{d}{dt} R_x(\eta_x + t\xi_x)\Big|_{t=0}, \tag{1.9}$$

where R is a retraction [AMS08].

Vector transport can also be defined by parallelization [Hua13, HAG15]. In Riemannian optimization, the d-dimensional matrix manifolds have tangent spaces that can be represented by d-dimensional linear subspaces of the w-dimensional Euclidean space, where $w > d$. In practice, w-dimensional tangent vectors are more commonly used; however, they may not be efficient, especially for large problems. Thus, it may be preferable to represent the tangent vectors by their intrinsic representation, i.e., a tangent vector $\eta_x \in \mathrm{T}_X\mathcal{M}$ can be represented by a d-dimensional vector, denoted v, of coordinates in a given basis B_X of $\mathrm{T}_X\mathcal{M}$ [Hua13, Section 9.5]. If the columns of B_X forms an orthonormal basis of $\mathrm{T}_X\mathcal{M}$, then many operations are inexpensive to compute (e.g., vector transport, metric, etc.) [Hua13, Section 9.5]. Thus, vector transport by parallelization is defined by

$$\mathcal{T} = B_Y B_X^\flat \tag{1.10}$$

where B_Y and B_X are bases of $\mathrm{T}_Y\mathcal{M}$ and $\mathrm{T}_X\mathcal{M}$, respectively. Observe that if B_Y and B_X are orthonormal bases of $\mathrm{T}_Y\mathcal{M}$ and $\mathrm{T}_X\mathcal{M}$, respectively, i.e., $B_X^\flat B_X = I$ for all X where I is the identity matrix, then the vector transport by parallelization is the identity [HAG15]. Parallelization is an isometric vector transport.

1.3.7 Tangent Cone

In recent years, rank constrained optimization has become a popular approach when solving applications where the data set is large. The idea is to assume that the data set has a rank smaller than the size of the problem and try to find the rank. That is, given a smooth function f where $f : \mathbb{R}^{m\times n} \to \mathbb{R} : X \mapsto f(X)$, find X such that the $\mathrm{rank}(X) \leq k$ where $k \ll \min(m, n)$, i.e.,

$$\min f(X) \;\text{ subject to }\; X \in \mathcal{M}_{\leq k}, \tag{1.11}$$

where $\mathcal{M}_{\leq k} := \{X \in \mathbb{R}^{m\times n} | \operatorname{rank}(X) \leq k\}$, i.e., the set of matrices of rank at most k. In general, (1.11) is NP-hard due to the combinatorial nature of the rank function [VB96]. Also, the set $\mathcal{M}_{\leq k}$ usually does not have a manifold structure for any $X \in \mathcal{M}_{\leq k}$ with rank less than k since the set may not have a tangent space at these points [Zho15, Section 4.3]. However, for all $X \in \mathcal{M}_{\leq k}$, a tangent cone, which is an extension of the tangent space, exists. The tangent cone to $\mathcal{M}_{\leq k}$ at a point $X \in \mathcal{M}_{\leq k}$ is the set

$$\mathrm{T}_X\mathcal{M}_{\leq k} := \left\{\dot{\gamma}(0) \mid \gamma \in \mathcal{C}^1,\ \gamma(0) = X,\ \exists \delta > 0\ :\ \forall t \in (0,\delta)\ :\ \gamma(t) \in \mathcal{M}_{\leq r}\right\},$$

where $\dot{\gamma}(0)$ denotes the derivate of the curve γ at 0 [OW04, Zho15, ZHG$^+$16]. Note that that the tangent cone is not necessarily closed under addition. Thus, the points of $\mathcal{M}_{\leq k}$ where the tangent space does not exist are matrices with rank $r < k$ and are elements of the fixed-rank manifold $\mathcal{M}_r = \{X \in \mathbb{R}^{m\times n} | \operatorname{rank}(X) = r\}$ where the tangent space $\mathcal{M}_r$ is a subset of the tangent cone $\mathrm{T}_X\mathcal{M}_{\leq k}$ [Zho15, ZHG$^+$16]. Observe that the set $\mathcal{M}_{\leq k}$ is equivalent to the union k of fixed-rank manifolds, i.e.,

$$\mathcal{M}_{\leq k} = \bigcup_{r=1}^{k} \mathcal{M}_r.$$

1.3.8 Riemannian Optimization Algorithms

In the early 1970s, Luenberger in [Lue72, Lue73] explored the idea of optimization on manifolds. In his work, he views equality constraints as defining a surface in $\mathbb{R}^n$ and describes a line-search method along geodesics on the surface. However, in general, this approach is not computationally feasible. In most optimization algorithms on manifolds, it has been shown that an approximation of the geodesic is enough to guarantee the desired convergence properties. Furthermore, many classical mathematical definitions in Riemannian geometry (e.g., geodesic, Levi-Civita connection, parallel vector transport, etc.) can be replaced by approximations.

The idea of efficient computations of Riemannian objects necessary for manifold optimization has been investigated by many researchers. In 1982, Gabay in [Gab82] proposed a Newton method on an embedded submanifold of $\mathbb{R}^n$ by using projective methods to compute the gradient vector tangent to the submanifold. Then, he computed a minimum in $\mathbb{R}^n$ along this direction and projected the minimum back onto the submanifold. In 1993, the dissertation of Smith analyzed optimization on differentiable functions on general Riemannian manifolds, generalized three algorithms (steepest

descent, Newton's method, and conjugate-gradient method) onto Riemannian manifolds and proved their convergence [Smi93]. Many other authors have attempted to improve the efficiency of manifold optimization methods (see [EAS98, OW00, MM02, Man02, DPM03, HT04, OHM06]).

Current research has focused on making optimization on manifolds more practical and flexible. Absil et al. [AMS08] provides an introduction to manifold optimization with an emphasis on certain concepts in differential geometry necessary for algorithmic development. In 2008, Baker [Bak08] developed a complete theory for the Riemannian trust-region Newton family of methods. Riemannian trust-region methods construct a quadratic model of the objective function around the current iterate and produces a candidate for the new iterate by (approximately) minimizing the model with a region where it is "trusted". Baker's approach follows the "lift-solve-retract" procedure where he first uses a retraction R on the Riemannian manifold $\mathcal{M}$ to "lift" the cost function f on $\mathcal{M}$ to a cost function $\hat{f}_x = f \circ R_x$ defined on the tangent space $\mathrm{T}_x\mathcal{M}$ for any point $x \in \mathcal{M}$. Since the tangent space is a Euclidean space, a quadratic model is defined on $\mathrm{T}_x\mathcal{M}$ and a point that sufficiently reduces the cost function is computed by the "inverse free" truncated conjugate-gradient method [Ste83]. This point is retracted from $\mathrm{T}_x\mathcal{M}$ to $\mathcal{M}$ using the retraction and is a candidate for the new iterate depending on the quality of the agreement between the lifted cost function $\hat{f}$ and the original cost function f. This approach requires the exact second-order term (i.e., the Hessian of f), or the action of the Hessian on a tangent vector which may not be acceptable in terms of computational cost. However, the quadratic model only needs an "approximate Hessian" H_x that satisfies the approximation condition [AMS08, Equation (7.36)]. Thus, one can choose $H_x := \mathrm{Hess}(f \circ \tilde{R}_x)(0_x)$, where $\tilde{R}_x$ is any retraction and, assuming sufficient smoothness of f, and expand the quadratic model around $\hat{f}_x = f \circ \tilde{R}_x$ such that it is second-order. In particular, if $\tilde{R}_x = \mathrm{Exp}_x$, then $H_x := \nabla \operatorname{grad} f(x)$ $(= \mathrm{Hess}\, f(x))$, where ∇ denotes the Riemannian connection. If $\mathcal{M}$ is a Riemannian submanifold or a Riemannian quotient of a Euclidean space, then $\nabla \operatorname{grad} f(x)$ has a simple formula [AMS08, Section 5.3].

Recent research has concentrated on generalizing Euclidean line-search based algorithms that achieve superlinear and quadratic convergence to a Riemannian setting in a systematic manner. For example, quasi-Newton methods are frequently used in Euclidean optimization because they achieve superlinear convergence without computing the Hessian or a good approximation of the

linear system defined by the Hessian. A popular quasi-Newton method is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method and the associated restricted Broyden family of methods.

In 2011, Qi proposed and analyzed an approach to generalize the BFGS method on Riemannian manifolds and developed a convergence analysis [Qi11]. This generalization combines the retraction-based ideas described above with vector transport. Given a smooth cost function on a Riemannian manifold $\mathcal{M}$ with a Riemannian metric g, the Riemannian BFGS (RBFGS) defines a search direction $p_k \in \mathrm{T}_{x_k}\mathcal{M}$ at iteration k as the solution to the equation $\mathcal{B}_k p_k = -\operatorname{grad} f(x_k)$, where $\mathcal{B}_k$ is a linear operator that approximates the action of the Hessian in the appropriate direction and is updated for each iteration. Similar to the Euclidean BFGS, a version that propagates the inverse of $\mathcal{B}_k$ was also developed. Then, the new iteration x_{k+1} is generated by a line-search method with stepsize α_k, i.e., $x_{k+1} = R_{x_k}(\alpha_k p_k)$ [Qi11, Section 2].

To update $\mathcal{B}_k$, Qi proposed the following update formula based on the vector transport $\mathcal{T}$ with associated retraction R to define the linear operator $\mathcal{B}_{k+1} : \mathrm{T}_{x_{k+1}}\mathcal{M} \to \mathrm{T}_{x_{k+1}}\mathcal{M}$,

$$\mathcal{B}_{k+1} = \tilde{\mathcal{B}}_k - \frac{\tilde{\mathcal{B}}_k s_k (\tilde{\mathcal{B}}_k^* s_k)^\flat}{(\tilde{\mathcal{B}}_k^* s_k)^\flat s_k} + \frac{y_k y_k^\flat}{y_k^\flat s_k},$$

where $\xi^\flat$ denotes the flat of ξ, $\mathcal{A}^*$ denotes the adjoint operator of $\mathcal{A}$, $s_k = \mathcal{T}_{\alpha_k p_k}(\alpha_k p_k)$, $y_k = \operatorname{grad} f(x_{k+1}) - \mathcal{T}_{\alpha_k p_k}(\operatorname{grad} f(x_k))$, and $\tilde{\mathcal{B}}_k = \mathcal{T}_{\alpha_k p_k} \circ \mathcal{B}_k \circ \mathcal{T}^{-1}_{\alpha_k p_k}$. The update formula for $\mathcal{H}_k = \mathcal{B}^{-1}$ is given by

$$\mathcal{H}_{k+1} = \tilde{\mathcal{H}}_k - \frac{(\tilde{\mathcal{H}}_k^* y_k)^\flat s_k}{y_k^\flat s_k} + \frac{s_k^\flat (\tilde{\mathcal{H}}_k^* y_k)}{s_k^\flat y_k} + \frac{s_k y_k^\flat (\tilde{\mathcal{H}}_k^* y_k) s_k^\flat}{(y_k^\flat s_k)^2} + \frac{s_k^\flat s_k}{s_k^\flat y_k},$$

where $\tilde{\mathcal{H}}_k = \mathcal{T}_{\alpha_k p_k} \circ \mathcal{H}_k \circ \mathcal{T}^{-1}_{\alpha_k p_k}$. The advantage of this approach was that it made it unnecessary to solve the a system of equations. Qi included a generalization of the Dennis and Moré condition to the Riemannian setting [Qi11, Section 2.3]. However, Qi's convergence analysis for RBFGS restricted the approach on a Riemannian manifold based on exponential mapping and parallel transport, which may be computationally expensive [Qi11, Section 2.4].

In 2012, Ring and Wirth extended Qi's BFGS approach by considering infinite dimensional Riemannian manifolds [RW12]. Their convergence analysis is for both finite and infinite dimensional Riemannian manifolds with the latter requiring some specific assumptions [RW12, Corollary 13]. In addition, their method does not require exponential mapping and parallel transport, but requires differentiated retraction which may be computationally expensive.

In his 2013 dissertation, Huang expanded the understanding and design of Riemannian quasi-Newton methods and computational efficiency for both line-search based and trust-region based algorithms [Hua13]. Huang proposed a systematic generalization of three well-known unconstrained optimization approaches from the Euclidean space to Riemannian manifolds: the Broyden family of methods [Hua13, Section 4], the symmetric rank-one trust-region method [Hua13, Section 3], and the gradient sampling method for both continuous and partly smooth cost functions [Hua13, Section 7]. His dissertation included a complete convergence analysis, a comprehensive implementation strategy for library design, and strategies for large scale problems for an appropriate subset of the methods [Hua13].

Similar to the Euclidean case, the Riemannian Broyden family is defined by taking a linear combination of the Riemannian Davidon-Fletcher-Powell (DFP) and the Riemannian BFGS methods based on a parameter ϕ_k. Huang defined the update formula as

$$\mathcal{B}_{k+1} = \tilde{\mathcal{B}}_k - \frac{\tilde{\mathcal{B}}_k s_k (\tilde{\mathcal{B}}_k^* s_k)^\flat}{(\tilde{\mathcal{B}}_k^* s_k)^\flat s_k} + \frac{y_k y_k^\flat}{y_k^\flat s_k} + \phi_k g(s_k, \tilde{\mathcal{B}} s_k) v_k v_k^\flat,$$

where $v_k = \frac{y_k}{g(y_k, s_k)} - \frac{\tilde{\mathcal{B}} s_k}{g(s_k, \tilde{\mathcal{B}} s_k)}$, $\tilde{\mathcal{B}}_k = \mathcal{T}_{S_{\alpha_k p_k}} \circ \mathcal{B}_k \circ \mathcal{T}^{-1}_{S_{\alpha_k p_k}}$, and $\mathcal{T}_S$ is an isometric vector transport (i.e., $\mathcal{T}_S$ satisfies (1.8)) [Hua13, Section 4]. When $\phi_k = 0$, the Riemannian Broyden family of methods reduces to RBFGS methods. The restricted Riemannian Broyden family is defined by the convex combination and the update preserves the positive definiteness of the Hessian approximation when suitable restrictions are placed on the stepsize and vector transport. When the combination is nonconvex, then the family becomes the entire Riemannian Broyden family. In the non-restricted case, convergence behavior and the choice of the parameter ϕ_k is more involved as in the Euclidean case. Huang analyzed the well-posedness of the Broyden family (restricted and non-restricted) and the convergence rate as a function of ϕ_k [Hua13, Section 6].

Huang further developed Qi's Riemannian Dennis Moré conditions so that they characterize the required correspondence between the action of $\mathcal{B}_k$ and the true Hessian to ensure superlinear convergence for optimization problems and related (more general) problems of finding zeros of Riemannian vector fields [Hua13, Section 5].

The theory also introduced the locking condition, which allows superlinear convergence to the restricted Riemannian Broyden family while avoiding the unacceptably large computational complexity of the differentiated retraction required by Ring and Wirth. The locking condition specifies

the relationship between the vector transport used and the differentiated associated retraction. The locking condition and the Riemannian Wolfe conditions are key to guarantee both superlinear convergence and well-posedness of the Riemannian Broyden family. In general, the theory relaxes the requirements on the retraction and vector transport, hence it subsumes the earlier RBFGS work of Qi [Qi11] and Ring and Wirth [RW12], and extends the understanding of Riemannian quasi-Newton methods.

In the Euclidean space, the symmetric rank-one (SR1) method is a member of the Broyden family defined by a nonconvex combination. The updated in SR1 does not preserve the positive definiteness and was considered to be an ineffective method for a long time. However, the SR1 updates are different from the Broyden family updates in that they provide a better approximation of the action of the Hessian on the entire space and not just in a single search direction. Huang generalized the SR1 to the Riemannian setting and proposed combining Riemannian SR1 with a Riemannian trust-region method that uses all of the direction information of the Hessian approximation [Hua13, Section 3]. The Riemannian symmetric rank-one trust-region method (RTR-SR1) is an efficient way to solve problems. The convergence analysis of RTR-SR1 in the Riemannian setting when restricted to the Euclidean setting extends the Euclidean results in the literature [Hua13, Section 3.3]. It does not require satisfying the locking condition since it is not a line-search approach.

In the case of large scale problems, an efficient way to store information is necessary. Huang developed, analyzed, and empirically evaluated limited-memory versions of RTR-SR1 and RBFGS that only store a few vectors to implicitly represent the rank-one update $\mathcal{B}_k$ [Hua13, Sections 3.4 and 4.5].

To solve optimization of partly smooth functions, Huang generalized the gradient sampling methods from the Euclidean space to Riemannian manifolds. Since we do not have a partly smooth function, we refer the reader to [Hua13, Section 7] for further details of this method.

1.4 Overview and Dissertation Statement

We propose to use Riemannian optimization theory and the associated efficient algorithms to increase our understanding of the role extraction problem and algorithms for its solution; to develop a significantly more efficient and robust approach applicable to various types of role structures and

a wide range of problem sizes; and to develop and evaluate robust implementations of the associated algorithms.

This dissertation asserts that the proposal above can be achieved by the following:

1. an analysis of the rank of the neighborhood pattern similarity measure with respect to the rank of the adjacency matrix of a graph and its number of roles (Chapter 3);
2. the development of a cost function and the use of Riemannian optimization to approximate the neighborhood pattern similarity measure on the symmetric positive semidefinite fixed-rank manifold for the first phase of the role extraction problem (Chapter 4);
3. the development of new Riemannian objects for the symmetric positive semidefinite fixed-rank manifold to improve the performance of state-of-the-art Riemannian algorithms (Chapter 4);
4. the use of k-means clustering and the silhouette statistic on the low-rank factor of the similarity measure to extract the role partition for the second phase of the role extraction problem (Chapter 4);
5. the proof that Browet and Van Dooren's full-rank iterative algorithm is a Euclidean gradient projection method along a projection arc with a fixed stepsize (Chapter 5);
6. the application of a stepsize and Armijo line-search method to the Euclidean gradient projection method along a projection arc, and the proof that the choice of the Armijo stepsize is dependent upon the similarity measure parameter β (Chapter 5);
7. empirical evidence of the two-phase role extraction approach being more efficient in time and robustness with Browet and Van Dooren's role extraction approach (Chapter 6);
8. the development and analysis of techniques to form a one-phase indirect approach to the role extraction problem and empirical evidence of robustness of the approach compared to our two-phase approach (Chapter 7);
9. empirical evidence that our two-phase indirect approach can partition signed networks such that the network is balanced (Chapter 8);
10. empirical evidence that there exists a relationship between overlapping community structures and role structures (Chapter 9).

CHAPTER 2

OVERVIEW OF CURRENT ROLE EXTRACTION METHODS

2.1 Introduction

The quality function used to solve the role extraction problem (1.1) is constructed either directly or indirectly. To directly construct the quality function, one must choose a cost function that is sensitive to a measure of distance between the blocks of edges in $A(\sigma_i, \sigma_j)$ connecting roles σ_i and σ_j and the ideal blocks between those roles, which is dependent upon the choice of equivalence relation. Let $\mathcal{K}(\sigma_i, \sigma_j)$ denote the set of ideal blocks between roles σ_i and σ_j, where these ideal blocks are a combination of a set of blocks defined in [DBF05, Rei09]. The direct approach can be defined by the optimization problem

$$\sigma^* = \arg\min_{\sigma} \sum_{\sigma_i, \sigma_j} \min_{B \in \mathcal{K}(\sigma_i, \sigma_j)} d\left(A(\sigma_i, \sigma_j), B\right),$$

where d is an appropriate measure of distance associated with a type of equivalence [DBF05]. For example, a distance measure associated with structural equivalence is given by

$$d\left(A(\sigma_i, \sigma_j), B\right) = \sum_{x \in \sigma_i} \sum_{y \in \sigma_j} |A(x, y) - B(x, y)|,$$

where $A(x, y)$ is the observed edge in the block and $B(x, y)$ is the corresponding value in the ideal block [DBF05]. Any community detection (or clustering) algorithm can be used to optimize the quality function to find the optimal assignment of roles [Rei09].

Another direct quality function based on the reduced graph is by Reichardt and White [RW07, Rei09]. We summarize this quality function in Section 2.3 since it is an extension of the modularity cost function for community detection [RW07, Rei09]. A more thorough explanation of other direct quality functions can be found in applications to social network analysis [DBF05].

A problem with direct methods is that there is no rigorous measure to determine whether or not a role assignment fits the data. Thus, multiple ideal role assignments may need to be tested

to determine the best fit for the data [DBF05]. Also, the number of roles needs to be specified in advance. So, direct methods require theoretical and empirical knowledge of the network in advance. Therefore, for networks where the number of roles is unknown, the method may need to be applied multiple times, increasing the number of roles each time [DBF05, Rei09].

The indirect approach to the role extraction problem comprises of two tasks:

(1) construct the quality function based on a (dis)similarity measure;

(2) define clusters as groups of nodes close to each other to extract the roles.

There are two approaches to constructing the quality function for indirect methods. The first is to embed the graph nodes in $\mathbb{R}^p$ based on some structural properties of the nodes (e.g., the degree, the number of neighbors at distance d, the measure of centrality, etc.) [WF94, DBF05, NG04, Kle10]. Once p measures have been computed for each node i and aggregated into an indicator vector t_i, then the pairwise dissimilarity measure is computed as the distance between the indicator vectors $D(i, j) = d(t_i, t_j)$, where d is defined as a vector norm [WF94, DBF05].

The other approach used to construct the quality function for indirect methods is to define the (dis)similarity measure based on the adjacency matrix. This approach is preferred when one wants to establish roles based on a specific equivalence criterion [Cas12, Bro14]. Pairwise node similarity measures based on regular equivalence are most often used [BGH$^+$04, LHN06, CB11, BDVB13, CAD13, CLVD16]. For these methods, each node from the input graph is compared with the nodes from the reduced graph by measuring how similar they are in terms of flows or connectivity patterns. In other words, given an input graph $G_A(V_A, E_A)$ and a reduced graph $G_B(V_B, E_B)$, the pairwise node similarity measure $S(A, B)$ defines a positive real value for every pair of nodes (i, j) with $i \in V_A$ and $j \in V_B$. Each node of the input graph can then be associated with the role, i.e., a node in the reduced graph, to which it is most similar.

In practice, the reduced graph is unknown. Thus, instead of assuming the structure of the reduced graph, each node of the input graph is compared to all the nodes in the same graph to define a pairwise node self-similarity measure for the input graph. The role structure is extracted from the similarity matrix using a community detection (or clustering) algorithm to group highly similar nodes together.

Several pairwise node similarity measures have been defined in the literature [BGH$^+$04, LHN06, CB11, BDVB13, CAD13, CLVD16]. However, many of these measure were later deemed unsuitable

for the role extraction problem due to difficulties encountered when extracting role structures from certain types of graphs (e.g., regular graphs and normal graphs) [Bro14]. In addition, these measures suffered from a loss of information (e.g., the origin, the destination, and the intermediate nodes involved in the transmission of the flow), or were more suited to detect community structures than role structures [Bro14]. Since the focus of this dissertation is an indirect approach to the role extraction problem, we summarize a few popular pairwise node similarity measures in Section 2.4.

Browet and Van Dooren solved the role extraction problem by using a similarity measure that assumes there exists more than one role in a network and identifies groups of nodes that have similar flow patterns [Bro14, BD14]. To represent these flow patterns, they used the self-similarity measure by Denayer [Den12] to compare the neighborhood patterns of every node, where the measure is high for any pair of nodes sharing analogous flow properties. Their two-phase role extraction method first computes the measure of neighborhood pattern based similarity using a matrix iterative scheme; and, second, extracts the role partitions using the fast community detection algorithm by Browet et al. [Bro14, BAD13]. For large networks, a modification of the two-phase algorithm was developed to reduce time and space requirements. The iterative scheme to compute the matrix with elements that measured pairwise node similarity was modified to include a projection onto low-rank matrices to converge to a low-rank approximation of the similarity matrix. So rather than producing the true $n \times n$ similarity matrix S where n in the number of nodes, an $n \times r$ matrix factor is computed such that $\hat{S} = XX^T$ approximates S. Since X is computed directly, i.e., $\hat{S}$ is not formed, these modifications resulted in a noticeable reduction in computational time for large networks without loss of robustness in determining the role structure [Bro14, BD14]. We summarize the neighborhood pattern similarity measure, as well as Browet and Van Dooren's low-rank approximation algorithm, in Section 2.4.3. The second phase of their algorithm requires computing the approximate similarity matrix $\hat{S}$ for use in community detection to extract the role partition. While the community detection algorithm by Browet et al. [BAD13, Bro14] is faster than other known community detection algorithms, the complexity of formulating the full similarity matrix at the beginning may be unacceptably costly for large networks in terms of both time and space.

Also, while any community detection (or clustering) algorithm can be used to find the optimal role assignment for direct and indirect quality functions, certain algorithms have been shown to de-

termine the optimal role assignment better than others and are preferred due to either convergence properties or efficiency [KR05, Rei09, Bro14, Tra14]. In Section 2.5, we summarize three algorithms used to find community structure; however more community detection and clustering algorithms can be found in [KR05, Rei09, Bro14, Tra14].

2.2 Quality Functions for Community Detection

Since the role extraction problem is a generalization of the community detection problem, we first summarize quality functions used to determine community structures in networks.

Girvan and Newman in [GN02] introduced community detection, which is used to cluster together highly similar nodes in a network when the pairwise node similarity is simply taken to be the edge weight. Community detection has been of interest in many research areas such as epidemiology [BCG$^+$09, ST12, TBD$^+$14], the influence and spread of information over social networks [LH05,WL08], the analysis of air transportation networks [GMTA05,LT13], and the detection of roles in metabolic networks [GNA05]. While there is no universally accepted definition of communities, communities can be, informally, defined as sets of nodes with a high internal density, either in the number of internal edges or their weight, and a low external density with the rest of the network. Therefore, to extract community structures, several quality functions have been proposed based on suitable rewards for edges that are present, or penalties for edges that are absent, within each community and are optimized over the community assignment for each node. However, in general, there may exist many local minima due to the flatness of the quality functions. Therefore, the community detection problem is NP-hard [BDG$^+$06, BDG$^+$08].

Assuming that rewards and penalties are given for the presence of the edges proportional to the weight of edges, a general cost function proposed by Reichardt and Bornholdt in [RB06] is

$$\mathcal{H}_{RB}(\sigma) = \sum_{i,j=1}^{N} (W_{i,j} - \gamma_{RB} p_{i,j}) \delta(\sigma_i, \sigma_j), \tag{2.1}$$

where W is the weighted adjacency matrix of the graph, N is the number of nodes, $p_{i,j}$ is the expected weight of an edge between nodes i and j known as the random null model, γ_{RB} is the resolution parameter, and σ_i denotes the community index of node i, where $\delta(\sigma_i, \sigma_j) = 1$ if $\sigma_i = \sigma_j$, and 0 otherwise. This cost function is known as the null model, and is often used as the general framework for other cost functions.

A popular cost function for community detection is modularity by Newman and Girvan [New04, NG04, LN08], and is obtained by choosing $\gamma_{RB} = 1$ and $p_{i,j} = (s_i^{out} s_j^{in})/m$ in (2.1), i.e.,

$$\mathcal{H}_{NG}(\sigma) = \frac{1}{m} \sum_{i,j=1}^{N} \left(W_{i,j} - \frac{s_i^{out} s_j^{in}}{m} \right) \delta(\sigma_i, \sigma_j), \tag{2.2}$$

where m is the sum of the weights of all the edges in the graph and $s_i^{in(out)}$ is the incoming (outgoing) strength of node i. Based on a community partition, the modularity cost function compares the actual edge density within each community to the expected edge density in a random network using the configuration null model. However, modularity suffers from a resolution limit, i.e., the size of the communities detected depends on the size of the network [FB07, SDYB12].

Traag et al. in [TVDN11] defined the Constant Potts Model (CPM) to avoid the resolution limit. For CPM, the expected weight $p_{i,j} = \gamma$ produces communities whose sizes are independent of the scale of the network and for which the optimal partition of the entire graph is also optimal for any subgraph induced by the set of communities. However, while CPM provides a better partition of the network, it might also require a procedure to optimize the parameter γ to extract the most significant partition [LDB08, DYB10, TVDN11, TKVD13].

There are several other cost functions that are used to measure the quality of the graph partitions and a detailed overview and analysis can be found in [POM09, For10].

2.3 Direct Quality Function for Role Extraction: Reichardt and White

Reichardt and White in [RW07, Rei09, PSR10] solve the role extraction problem by constructing a direct quality function that is an extension of the modularity cost function for community detection for directed graphs [GN02, New04, LN08]. For community detection, the generic modularity cost function is given by

$$\mathcal{H}(\sigma) = \sum_{i,j \in V} \left[a_{i,j} A_{i,j} \delta(\sigma_i, \sigma_j) + b_{i,j} (1 - A_{i,j})(1 - \delta(\sigma_i, \sigma_j)) \right], \tag{2.3}$$

where $a_{i,j} \geq 0$ (respectively $b_{i,j} \geq 0$) is the weight of the presence (absence) of an edge from node i to node j [GN02, New04, LN08, Tra14].

Assuming that the image graph is known to have r roles and can be represented by a $q \times q$ unweighted adjacency matrix B such that $B(\sigma_i, \sigma_j) = 1$ if the edges are allowed from role σ_i to role

σ_j and $B(\sigma_i, \sigma_j) = 0$ if the edges are forbidden, then the quality function by Reichardt and White to measure the fit between a partition σ for the graph, represented by its adjacency matrix A, and the reduced graph is given by

$$\mathcal{Q}_{RW}(\sigma, B) = \frac{1}{m} \sum_{i \neq j} \left[a_{i,j} A_{i,j} B(\sigma_i, \sigma_j) + b_{i,j}(1 - A_{i,j})(1 - B(\sigma_i, \sigma_j)) \right], \tag{2.4}$$

where $a_{i,j}$ (respectively $b_{i,j}$) are weights of the presence (absence) of an edge from node i to node j if an edge is allowed (forbidden) in the role model between σ_i and σ_j [RW07, Rei09]. Therefore, (2.4) is based on regular equivalence between the nodes inside each role. For structural equivalence, penalties for the presence of forbidden edges and for the absence of allowed edges are considered.

Equation (2.4) can be rearranged to be

$$\mathcal{Q}_{RW}(\sigma, B) = \frac{1}{m} \sum_{i \neq j} \left[(a_{i,j} + b_{i,j}) A_{i,j} - b_{i,j} \right] B(\sigma_i, \sigma_j). \tag{2.5}$$

Reichardt and White proposed to balance the weights $a_{i,j}$ and $b_{i,j}$ since there are more missing edges than present edges. Hence, they imposed that $a_{i,j} + b_{i,j} = w_{i,j}$ for all $i, j \in V_A$, where $w_{i,j}$ is the weight of edge $(i, j) \in E_A$, and that

$$\sum_{i \neq j} a_{i,j} A_{i,j} = \sum_{i \neq j} b_{i,j}(1 - A_{i,j}) \text{ or } \sum_{i \neq j} w_{i,j} A_{i,j} = \sum_{i \neq j} b_{i,j}.$$

Therefore, they set $a_{i,j} = w_{i,j} - b_{i,j}$ and $b_{i,j} = \gamma p_{i,j}$ where $p_{i,j}$ is the probability that an edge from node i to node j exists and is defined as $p_{i,j} = k_i^{out} k_j^{in} / m^2$, and parameter γ tunes the presence (or absence) of edges [Rei09]. If $\gamma = 1$, then (2.5) gives equal total weights to edges and missing edges, while $\gamma < 1$ (or $\gamma > 1$) gives more total weight to present (absent) edges [Rei09]. If the image matrix B is diagonal, i.e., the roles have community structure, then (2.5) is the modularity cost function for community detection [GN02, New04, LN08].

Observe that (2.5) can be written as summation over the role indices rather than summation over the node indices, i.e.,

$$\mathcal{Q}_{RW}(\sigma, B) = \frac{1}{m} \sum_{s,t}^{q} \left(e_{s,t} - \gamma \langle e_{s,t} \rangle \right) B(s, t), \tag{2.6}$$

where $e_{s,t}$ is the actual number of edges between clusters s and t and $\langle e_{s,t} \rangle$ is the expected number of edges such that

$$e_{s,t} = \sum_{i \neq j} w_{i,j} A_{i,j} \delta(\sigma_i, s) \delta(\sigma_j, t),$$
$$\langle e_{s,t} \rangle = \sum_{i \neq j} p_{i,j} \delta(\sigma_i, s) \delta(\sigma_j, t).$$

Maximizing (2.6) is equivalent to maximizing every term $(e_{s,t} - \gamma \langle e_{s,t} \rangle) B(s,t)$. Therefore, Reichardt and White define the image matrix as

$$B(s,t) = \begin{cases} 1 & , \text{ if } e_{s,t} - \gamma \langle e_{s,t} \rangle > 0, \\ 0 & , \text{ otherwise,} \end{cases} \tag{2.7}$$

and suggest to extract the role structure of the input graph by first maximizing

$$\mathcal{Q}^*_{RW}(\sigma) = \frac{1}{2} \sum_{s,t}^{q} \| e_{s,t} - \gamma \langle e_{s,t} \rangle \|. \tag{2.8}$$

They use simulated annealing to maximize the quality function (see Section 2.5.1); however, any community detection algorithm can be used to optimize $\mathcal{Q}_{RW}$ [RW07, Rei09].

Reichardt and White proved in [RW07, Rei09] that the maximum value of $\mathcal{Q}_{RW}$ is obtained when every edge in the network is allowed and missing edges are not allowed. Thus, the minimal image graph is one that captures the connectivity of the classes of structural equivalence within the network, and therefore, the maximum number of roles $q_{\max}$ in the image graph is equal to the number of structural equivalence classes. Then, $\mathcal{Q}^{max}_{RW}$ is computed by

$$\mathcal{Q}^{max}_{RW} = \frac{1}{m} \sum_{i \neq j} \left(w_{i,j} - \gamma \frac{k^{out}_i k^{in}_j}{m} \right) A_{i,j}, \tag{2.9}$$

where $A_{i,j}$ replaces $B(s,t)$ in (2.5) [RW07, Rei09].

Reichardt and White successfully applied their quality function to extract the role structure for the commodity trade networks [RW07] and protein-protein interaction networks [PSR10]. However, their quality function suffers from the fact that there is no guarantee that the optimization of (2.8) extracts a relevant role structure since the reduced graph is removed from the optimization scheme. Also, they use simulated annealing to optimize their quality functions since it yields good results, is very general in applications, and simple to implement. However, simulated annealing is known to be very slow and is not recommended for very large networks [GSPA04, GMTA05, Bro14].

2.4 Indirect Quality Function for Role Extraction: Similarity Measure

The focus of this dissertation is to solve the role extraction problem using an indirect method where the quality function is constructed by a self-similarity measure from the adjacency matrix. There are several graph similarity measures defined in the literature [BGH$^+$04, LHN06, CB11, BDVB13, CAD13, CLVD16], and, in this section, we summarize several similarity measures designed for the extraction of role structures based on regular equivalence.

2.4.1 The Similarity Measure of Blondel et al.

2.4.1.1 Definition of the Similarity Measure of Blondel et al.. The similarity measure of Blondel et al. is a pairwise similarity measure that considers two nodes $i \in V_A$ and $j \in V_B$ in different graphs $G_A(V_A, E_A)$ and $G_B(V_B, E_B)$ to be similar if the children of i are similar to the children of j or the parents of i are similar to the parents of j [BGH$^+$04]. That is, the similarity score between nodes $i \in V_A$ and $j \in V_B$ is

$$S_{i,j} = \sum_{s:(s,i)\in E_A, t:(t,j)\in E_B} S_{s,t} + \sum_{s:(i,s)\in E_A, t:(j,t)\in E_B} S_{s,t}.$$

This similarity score can be written in the compact matrix form

$$S_{k+1} = AS_kB^T + A^TS_kB = \Gamma_{A,B}[S_k],$$

where A and B are adjacency matrices of G_A and G_B with n_A and n_B nodes, respectively, S_k is the $n_A \times n_B$ matrix of entries $S_{i,j}$ at iteration k, and $\Gamma_{A,B}[S_k]$ is a linear map on the matrix S_k.

Since only the relative score of each pair of nodes is of interest and not the value of $S_{i,j}$, the pairwise node similarity measure of Blondel et al. $S^B_{A,B}$ is a fixed point of the iterative sequence

$$S_{k+1} = \frac{AS_kB^T + A^TS_kB}{\|AS_kB^T + A^TS_kB\|_F} = \frac{\Gamma_{A,B}[S_k]}{\|\Gamma_{A,B}[S_k]\|_F}, \tag{2.10}$$

where $S_0 = \mathbf{1}\mathbf{1}^T$. Equation (2.10) can be written in vector form as

$$vec(S_{k+1}) = \frac{\left(B \otimes A + (B \otimes A)^T\right) vec(S_k)}{\|\left(B \otimes A + (B \otimes A)^T\right) vec(S_k)\|_2} = \frac{\Gamma vec(S_k)}{\|\Gamma vec(S_k)\|_2},$$

where $vec(S_k)$ is the vectorization of the matrix S_k formed by stacking vertically the columns S_k into a column vector, and the matrix Γ is symmetric and nonnegative.

It has been shown that the sequence converges to $vec(S) = \frac{\Pi vec(S_0)}{\|\Pi vec(S_0)\|_2}$, where Π is the orthogonal projector onto the invariant subspace of Γ associated to the dominant eigenvalue $\rho(\Gamma)$ [BGH$^+$04]. However, $-\rho(\Gamma)$ is another possible eigenvalue of Γ. Thus, there exists two converging and alternating sequences based on the even or odd iterates [BGH$^+$04]. The authors suggest to define the pairwise node similarity matrx by the sequence of even iterates, since the sequence converges to a unique nonnegative vector with the largest possible 1-norm, i.e., the sum of the magnitudes of all entries [BGH$^+$04].

For the role extraction problem, if the hypothetical role structure is unknown for the network, then (2.10) can be computed as a pairwise self-similarity measure $S^B_{A,A} = S^B_A$ and is the fixed point solution of the sequence

$$S_{k+1} = \frac{AS_kA^T + A^TS_kA}{\|AS_kA^T + A^TS_kA\|_F}. \tag{2.11}$$

However, (2.11) tends to give higher similarity scores for pairs of nodes involving one high degree node, which can lead to the situation where pairs of nodes are more similar to other nodes than to themselves. An effective method to avoid this situation is to diagonally scale the matrix such that each node is exactly similar to itself, i.e., $S_{i,i} = 1$, and all other similarity scores are less than or equal to one, i.e., $S_{i,j} \leq 1$, that is

$$\overline{S^B_A} = D_S^{-0.5} S^B_A D_S^{-0.5},$$

where D_S is the diagonal matrix of the unscaled self-similarity scores [Cas12].

The Blondel et al. similarity measure has been applied to automatically extract synonyms from a dictionary by assuming that synonyms have many words in common in their definitions and appear together in the definition of many words [BGH$^+$04]. However, the similarity measure cannot extract relevant similarity scores when the graph is regular or the adjacency matrix A is a normal matrix, i.e., a matrix A is normal if it satisfies $AA^T = A^TA$, because the rank of the similarity matrix is 1 [BGH$^+$04]. Thus, after scaling, the role structure cannot be extracted because $\overline{S^B_A} = \mathbf{1}\mathbf{1}^T$.

2.4.1.2 Low-rank Approximation. The computational complexity of the similarity measure of Blondel et al. is $O(n^3)$. However, two low-rank approximation schemes have been proposed to compute a low-rank approximation of the similarity matrix. The scheme by Cason et al. in [Cas12, CAD13] is an iterative algorithm, while the method by Zhou in [Zho15] is a Riemannian

optimization method on the set of matrices with Frobenius norm 1 and rank at most k, i.e.,

$$\mathcal{S}_{\leq k}(n,q) = \left\{ UDV^T \mathbb{R}^{n\times r} \mid U \in St(q,n), V \in St(k,q), D \text{ diagonal }, \|D\|_F = 1 \right\},$$

where q is the number of roles, $St(q,n)$ denotes the Stiefel manifold (the set of all $n \times q$ orthonormal matrices). Note that $\mathcal{S}_{\leq k}(n,q)$ is not a manifold, rather it is the union of fixed-rank manifolds, i.e.,

$$\mathcal{S}_{\leq k}(n,q) = \bigcup_{r=1}^{k} \mathcal{S}_r(n,q), \tag{2.12}$$

where

$$\mathcal{S}_r(n,q) = \left\{ UDV^T \mathbb{R}^{n\times q} \mid U \in St(q,n), V \in St(k,q), D_r \text{ diagonal }, \|D_r\|_F = 1 \right\},$$

is a fixed-rank manifold with r nonzero singular values [Zho15].

Both methods find solutions of the optimization problem

$$S_{k+1} := \underset{\|S\|_F=1}{\arg\max} \left\langle S, \Gamma_{A,B}[S_k] \right\rangle_F, \tag{2.13}$$

where $\langle \cdot, \cdot \rangle_F$ denotes the Frobenius inner product. This optimization problem can be written as

$$S_{k+1} := \underset{S\in\mathcal{S}_{\leq k}(n,r)}{\arg\max} \left\langle S, \Gamma^2_{A,B}[S_k] \right\rangle_F, \tag{2.14}$$

where $\Gamma^2_{A,B}[S_k] = \Gamma_{A,B}[\Gamma_{A,B}[S_k]]$ [Cas12, CAD13, Zho15]. See [Cas12, CAD13] for a proof of convergence of Cason et al.'s low-rank iterative algorithm, and see [Zho15] for the convergence of Zhou's Riemannian optimization approach.

2.4.2 The Similarity Measure of Cooper and Barahona

The self-similarity similarity measure of Cooper and Barahona is based on the number of paths originating from or leading to each node [CB11, BDVB13]. The number of paths of length ℓ from node i to node j is given by $[A^\ell]_{i,j}$ and the number of paths from j to i is $[(A^T)^\ell]_{i,j}$. Then, the total number of outgoing and incoming paths of length ℓ from and to node i is $[A^\ell \mathbf{1}]_i$ and $[(A^T)^\ell \mathbf{1}]_i$.

Intuitively, if two nodes have similar connectivity patterns, then they should roughly have the same number of neighbors at various distances. Based on this intuitive idea, the indicator matrix X containing the total number of paths of length $\ell \leq \ell_{max}$ is given by

$$X = \left[\beta A\mathbf{1} \mid \cdots \mid (\beta A)^{\ell_{max}}\mathbf{1} \mid \beta A^T\mathbf{1} \mid \cdots \mid (\beta A^T)^{\ell_{max}}\mathbf{1} \right], \tag{2.15}$$

where the first ℓ_{max} columns correspond to the outgoing paths and the last ℓ_{max} columns correspond to the incoming paths. The parameter $\beta = \alpha/\rho(A)$ guarantees convergence of the sequence where $\alpha \in [0, 1]$ weighs the importance of long paths (global connectivity) with respect to short paths (local connectivity), and $\rho(A)$ is the dominant eigenvalue of A, i.e., $\rho(A)$ is the spectral radius of A [CB11, BDVB13].

Since each row of X is an indicator vector of the total flow profile of the associated node, then the similarity score of Cooper and Barahona S_A^{CB} between nodes i and j is

$$S_{i,j} = \frac{x_i x_j^T}{\|x_i\|\|x_j\|}, \tag{2.16}$$

where $\|\cdot\|$ is the Euclidean norm, and scaling ensures that $S_{i,i} = 1$ and $S_{i,j} \leq 1$ [CB11, BDVB13]. The similarity matrix can be computed for $\ell_{max} \to \infty$ as the normalized sum of the two iterative sequences

$$S_{k+1}^{out} = A\left(\mathbf{1}\mathbf{1}^T + \left(\frac{\alpha}{\rho(A)}\right)^2 S_k^{out}\right) A^T \tag{2.17}$$

$$S_{k+1}^{in} = A^T\left(\mathbf{1}\mathbf{1}^T + \left(\frac{\alpha}{\rho(A)}\right)^2 S_k^{in}\right) A, \tag{2.18}$$

which converge when $\alpha < 1$ [CB11, BDVB13].

Note that the similarity measure of Cooper and Barahona is cheaper to compute than the similarity measure by Blondel et al. when ℓ_{max} is finite. However, information (like the origins or destinations of the paths) is lost since the measure only considers the total number of paths by computing the product of powers of A and $\mathbf{1}$.

2.4.3 Neighborhood Pattern Similarity Measure

2.4.3.1 Definition of Neighborhood Pattern Similarity Measure. The next similarity measure we describe is the neighborhood pattern similarity measure, first proposed by Denayer in [Den12] and further analyzed by Browet and Van Dooren in [Bro14, BD14]. Given a directed graph $G_A(V, E)$, the neighborhood pattern similarity measure $S \in \mathbb{R}^{n\times n}$ is defined as a fixed point of the operator

$$S = \Gamma_A[I + \beta^2 S] \tag{2.19}$$

where I is the identity matrix, $\beta \in \mathbb{R}$, and, given a matrix $X \in \mathbb{R}^{n\times n}$, the operator $\Gamma_A[\cdot]$ is defined as

$$\Gamma_A : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n} : X \mapsto \Gamma_A[X] = AXA^T + A^TXA. \tag{2.20}$$

To derive (2.19), Browet and Van Dooren defined a neighborhood pattern of length ℓ as the number of incoming (I) and outgoing (O) edges starting from a node, which they called the source node [Bro14]. For example, neighborhood patterns of length 1 are patterns where two nodes are similar if they have common parents, i.e., Figure 2.1a, or common children, i.e., Figure 2.1b. The number of common parents between two nodes (i, j) is the number of nonzero row elements shared by the i-th and j-th columns of A, i.e., $[A^TA]_{i,j}$ and the number of common children is the number of nonzero column elements shared by the i-th and j-th rows of A, i.e., $[AA^T]_{i,j}$. Therefore, the number of common reachable nodes, called target nodes, between every pair of source nodes for neighborhood patterns of length 1 is $N_1 = AA^T + A^TA$ [Bro14].

For neighborhood patterns of length 2, there are four possible neighborhood patterns (see Figure 2.2) and the number of common target nodes between every pair of source nodes for neighborhood patterns of length 2 is given by

$$N_2 = AAA^TA^T + AA^TAA^T + A^TAA^TA + A^TA^TAA = AN_1A^T + A^TN_1A = \Gamma_A^2[I],$$

where $\Gamma_A^2[\cdot]$ corresponds to the operator $\Gamma_A[\cdot]$ defined by (2.20) applied two times.

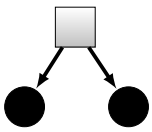
Following this same pattern, there are eight possible neighborhood patterns of length 3 (see Figure 2.3) and the number of common target nodes can be computed by N_3. In general, the number of possible neighborhood patterns of length ℓ is 2^ℓ and the number of common target nodes is given by

$$N_\ell = AN_{\ell-1}A^T + A^TN_{\ell-1}A = \Gamma_A^\ell[I],$$

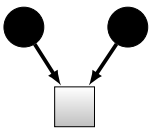
where $\Gamma_A^\ell[\cdot]$ corresponds to applying the operator $\Gamma_A[\cdot]$ defined by (2.20) ℓ times. Therefore, the neighborhood pattern pairwise node similarity measure can be defined as the weighted sum of the number of common target nodes of the neighborhood patterns of any length, i.e.,

$$S = \sum_{\ell=1}^{\infty} \beta^{2(\ell-1)} N_\ell = \sum_{\ell=1}^{\infty} \beta^{2(\ell-1)} \Gamma_A^\ell[I], \tag{2.21}$$

where $\beta \in \mathbb{R}$ is a scaling parameter [Bro14].

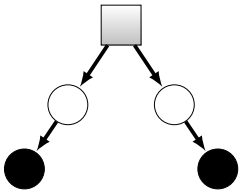


(a) pattern I: A^TA

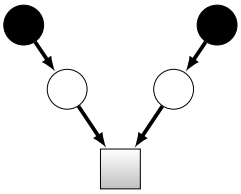


(b) pattern O: AA^T

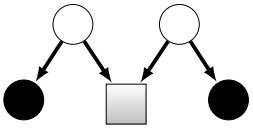
Figure 2.1: All possible neighborhood patterns of length 1 for the similarity pattern (2.19) where the source nodes i and j are the black circles and the target node is the gray square



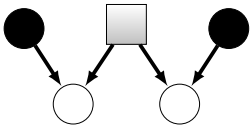
(a) pattern I-I: A^TA^TAA



(b) pattern O-O: AAA^TA^T



(c) pattern I-O: A^TAA^TA



(d) pattern O-I: AA^TAA^T

Figure 2.2: All possible neighborhood patterns of length 2 for the similarity pattern (2.19) where the source nodes i and j are the black circles and the target node is the gray square

Browet and Van Dooren showed in [Bro14, BD14] that (2.21) can be computed as the limit $k \to \infty$ of the iterative sequence (2.19), since for an initial matrix S_0,

$$S_{k+1} = \Gamma_A[I] + \left(\beta^2\right)^2 \Gamma_A^2[I] + \cdots + \left(\beta^2\right)^k \Gamma_A^{k+1}[I] + \left(\beta^2\right)^{k+1} \Gamma_A^{k+1}[S_0].$$

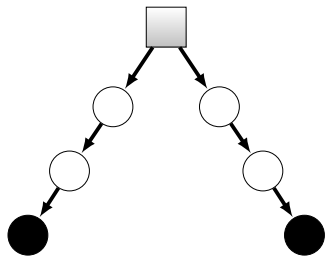
They set $S_0 = 0$ to get the enumeration of patterns discussed previously. Therefore, (2.19) can be written for $k \geq 1$ as

$$S_{k+1} = S_1 + \beta^2 \Gamma_A[S_k], \tag{2.22}$$

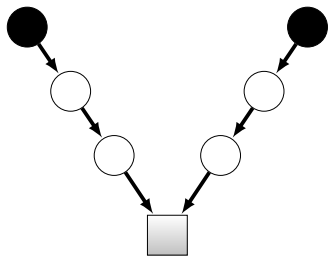
where

$$S_1 = \Gamma_A[I] = AA^T + A^TA. \tag{2.23}$$

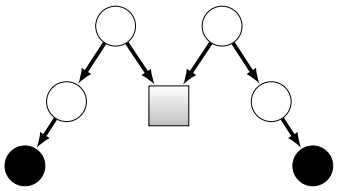
The similarity matrix S is a symmetric positive semidefinite matrix and the parameter β can be chosen to increase the weight of long neighborhood paths while guaranteeing convergence of the



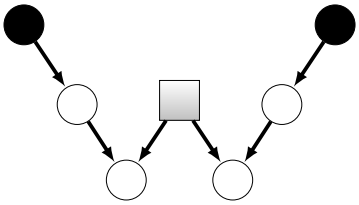
(a) pattern I-I-I: $A^T A^T A^T AAA$



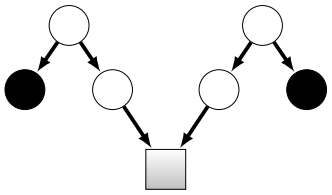
(b) pattern O-O-O: $AAAA^T A^T A^T$



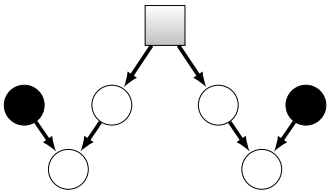
(c) pattern I-I-O: $A^T A^T AA^T AA$



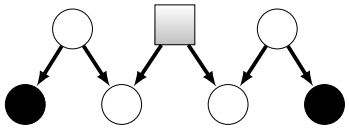
(d) pattern O-O-I: $AAA^T AA^T A^T$



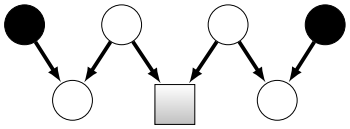
(e) pattern I-O-O: $A^T AAA^T A^T A$



(f) pattern O-I-I: $AA^T A^T AAA^T$



(g) pattern I-O-I: $A^T AA^T AA^T A$



(h) pattern O-I-O: $AA^T AA^T AA^T$

Figure 2.3: All possible neighborhood patterns of length 3 for the similarity pattern (2.19) where the source nodes i and j are the black circles and the target node is the gray square

sequence S_k in (2.19) [Bro14, BD14]. The fixed point of (2.22) of interest, S_*, is

$$vec(S_*) = \left[I - \beta^2 \left(A \otimes A + (A \otimes A)^T\right)\right]^{-1} vec(S_1), \tag{2.24}$$

where $vec(S)$ denotes the vectorization of the matrix S formed by stacking the columns of S into one column vector.

It was shown in [Bro14, BD14] that

$$\beta^2 < \frac{1}{\rho\left(A \otimes A + (A \otimes A)^T\right)}, \tag{2.25}$$

is a necessary and sufficient condition for the sequence to converge to S_*, i.e., to have a well-defined similarity metric. A sufficient bound that is less computationally expensive to approximate to ensure that constraint (2.25) is satisfied is given by

$$\beta^2 \leq \frac{1}{\rho\left((A + A^T)\right)^2}. \tag{2.26}$$

Unfortunately, even when S_* is well-defined, the fixed point may be impractical to compute due to high computational cost and memory requirements. That is, even if A is sparse, the matrix S_k usually becomes denser as k increases and the complexity of an iteration is $O(mn^2)$ where m is the number of edges, i.e., nonzero elements, in A [Bro14]. However, Browet and Van Dooren addressed the issue of computational complexity with a low-rank approximation of the similarity measure.

Also, as with the similarity measure of Blondel et al., S_* may have pairs of nodes that are more similar to other nodes than to themselves. Therefore, diagonally scaling the matrix such that $S_{i,i} = 1$, and $S_{i,j} \leq 1$ avoids this issue, i.e.,

$$\overline{S_*} = D_{S_*}^{-0.5} S_* D_{S_*}^{-0.5}, \tag{2.27}$$

where D_{S_*} is the diagonal matrix of the unscaled self-similarity scores. Lastly, the rank of $\overline{S_*}$ is equal to the rank of S_* since $D_{S_*}^{-0.5}$ is a nonsingular matrix, which we will use later in Chapter 3.

Observe that the neighborhood pattern similarity measure is a generalization of the similarity measure of Cooper and Barahona, S^{CB}, which only compares the total number of paths originating from or leading to a node and does not compare the target nodes nor source nodes of those paths. Also, S^{CB} is restricted to only direct paths (which are define by patterns I-I-$\cdots$-I and O-O-$\cdots$-O) and does not consider all possible types of neighborhood patterns. Therefore, the similarity measure of Cooper and Barahona for an unweighted regular block cycle graph (where each role contains the same number of nodes and each node is connected to all of the nodes in the next role of the cycle) is rank 1 because all of the nodes have a constant number of incoming and outgoing neighbors for any distance. Thus, the role structure cannot be extracted from S^{CB}. The neighborhood pattern similarity is able to distinguish between the blocks since it considers all possible neighborhood patterns of length ℓ. We show in Chapter 3 why the role structure can be extracted from the neighborhood pattern similarity measure.

2.4.3.2 Low-rank Approximation. Even if the sequence (2.22) for the neighborhood pattern similarity measure is guaranteed to converge, the computational cost to compute the fixed point may be unacceptably high. Therefore, Browet and Van Dooren defined a low-rank approximation of the similarity matrix S_{k+1} [Bro14, BD14]. For $S_{k+1}^{(r)} = X_{k+1}X_{k+1}^T$, where $X_{k+1} \in \mathbb{R}_*^{n\times r}$ is a full-column rank matrix of size $n \times r$, the low-rank similarity approximation scheme is defined as

$$\begin{aligned} S_1^{(r)} &= \Pi^{(r)}\left[[A \mid A^T][A \mid A^T]^T\right] = X_1X_1^T \\ S_{k+1}^{(r)} &= \Pi^{(r)}\left[S_1^{(r)} + \beta^2\Gamma\left[S_k^{(r)}\right]\right] = X_{k+1}X_{k+1}^T \end{aligned} \tag{2.28}$$

where $\Pi^{(r)}[\cdot]$ is an orthogonal projector onto the dominant subspace of dimension at most r, which is computed as a truncated singular value decomposition (SVD) on the R_k-factor, i.e., $R_k = U_k\Sigma_kV_k^T$ where $R_k \in \mathbb{R}^{3r\times 3r}$ is upper triangular, U_k and V_k are orthogonal $3r \times r$ matrices, and Σ_k is a diagonal matrix of the singular values $\sigma_1 > \cdots > \sigma_r \geq 0$, of the QR-factorization of Y_k defined as

$$Y_k = \left[X_1 \mid \beta AX_k \mid \beta A^TX_k\right]. \tag{2.29}$$

The updated factor X_{k+1} is computed as

$$X_{k+1} = Q_kU_k\Sigma_k. \tag{2.30}$$

The stopping criterion for the low-rank iteration of Browet et al. requires the absolute Frobenius norm difference between X_{k+1} and X_k to be less than a given $\epsilon > 0$. To avoid numerical cancellation, this can be computed by

$$\|\tilde{R}_{k+1}\begin{pmatrix} I_r & 0 \\ 0 & -I_r \end{pmatrix}\tilde{R}_{k+1}^T\|_F \leq \epsilon \tag{2.31}$$

where $\tilde{R}_{k+1} \in \mathbb{R}^{2r\times 2r}$ is the upper triangular matrix from the QR-factorization of $[X_{k+1} \mid X_k]$ [Bro14, BD14]. Algorithm 1 summarizes Browet and Van Dooren's low-rank iterative method.

Note that the initial iterate for Algorithm 1 requires the rank r truncated SVD computation of a $n \times 2n$ (ideally sparse) matrix and nr operations of a matrix products yielding a total of $O(nr^2) + O(nr)$ for the initial iterate where the coefficient of the first term depends upon the method used to compute the truncated SVD [Lar98, BR05]. After the initial iterate, the algorithm requires $4nmr$ operations matrix multiplications (line 4), the QR-factorization of a $n \times 3r$ matrix (line 5), the rank r truncated SVD computation of a $3r \times 3r$ upper triangular matrix (line 6), and $9nr^2$ operations of matrix multiplication (line 7) yielding a total of $O(nmr) + O(nr^2) + O(r^3)$

Algorithm 1 Browet and Van Dooren's Low-Rank Similarity Measure Iteration [Bro14, BD14]

Input: $A \in \mathbb{R}^{n \times n}$, $\beta \in \mathbb{R}$, fixed-rank r, and tolerance $\epsilon << 1$

Output: $X_{k+1} \in \mathbb{R}^{n \times r}$

1: Compute rank r truncated SVD of $\begin{bmatrix} A \mid A^T \end{bmatrix}$ for U_1 and Σ_1
2: Compute $X_1 = U_1 \Sigma_1$
3: **for** $k = 1,\ 2, \ldots,$ **do**
4: Set $Y_k = \begin{bmatrix} X_1 \mid \beta A X_k \mid \beta A^T X_k \end{bmatrix}$
5: Compute QR-factorization of Y_k for Q_k and R_k
6: Compute rank r truncated SVD of R_k for $U_k,$ and Σ_k
7: Compute $X_{k+1} = Q_k U_k \Sigma_k$
8: Compute QR-factorization of $[X_{k+1} \mid X_k]$ for $\tilde{R}_{k+1}$
9: **if** $\left\| \tilde{R}_{k+1} \begin{pmatrix} I_r & 0 \\ 0 & -I_r \end{pmatrix} \tilde{R}_{k+1}^T \right\|_F \leq \epsilon$ **then**
10: Stop
11: **end if**
12: **end for**

where the coefficient of the second term depends upon the method used to compute the QR-factorization [GV13].

The authors proved that the low-rank iterative scheme (2.28) converges locally to a fixed point $S^{(r)}$ if the spectral gap at the r^{th} singular value of S_* is sufficiently large [Bro14]. In other words, for $S^{(r)} = XX^T = U\Sigma^2 U^T$ where $X \in \mathbb{R}_*^{n \times r}$, $U \in \mathbb{R}^{n \times r}$ is an orthogonal matrix and the dominant subspace of the low-rank approximation (2.22), and $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix of the dominant singular values $\sigma_1 > \cdots > \sigma_r \geq 0$, Algorithm 1 converges locally when β satisfies [Bro14]

$$\beta^2 < \frac{1}{\|A \otimes A + (A \otimes A)^T\|_F \left(\frac{8\|\Sigma\|_F}{\sigma_r^2 - \sigma_{r+1}^2} + 1 \right)}. \tag{2.32}$$

Note this does not necessarily cover the entire range of condition (2.25).

2.4.4 The Similarity Measure of Cheng et al.

2.4.4.1 Definition of the Similarity Measure of Cheng et al.. The last self-similarity measure we describe considers only the parents and children of two nodes and was proposed by Cheng et al. in [CLVD16]. This similarity measure considers two nodes to be similar if they have the same children or parents. This is similar to the neighborhood pattern similarity measure for neighborhood patterns of length $\ell = 1$.

To derive the similarity matrix, first take $\beta = 0$ in (2.19), i.e.,

$$S = AA^T + A^T A. \tag{2.33}$$

For this measure, only the common number of parents and children are considered. However, (2.33) may lead to incorrect roles because it may fail to distinguish small roles when implementing a low-rank projection [CLVD16]. Thus, Cheng et al. computed the percentage of common children and parents between two nodes for (2.33), where the percentage of common children between nodes i and j is defined as

$$\frac{\left(AA^T\right)_{i,j}}{\sqrt{\sum_{k=1}^{n} A_{i,k}}\sqrt{\sum_{k=1}^{n} A_{j,k}}} \tag{2.34}$$

and the percentage of common parents is defined as

$$\frac{\left(A^T A\right)_{i,j}}{\sqrt{\sum_{k=1}^{n} A_{k,i}}\sqrt{\sum_{k=1}^{n} A_{k,j}}}. \tag{2.35}$$

Therefore the pairwise node similarity measure between nodes i and j (denoted $S^C_{i,j}$) is defined as

$$S^C_{i,j} = \frac{\left(AA^T\right)_{i,j}}{\sqrt{\sum_{k=1}^{n} A_{i,k}}\sqrt{\sum_{k=1}^{n} A_{j,k}}} + \frac{\left(A^T A\right)_{i,j}}{\sqrt{\sum_{k=1}^{n} A_{k,i}}\sqrt{\sum_{k=1}^{n} A_{k,j}}}. \tag{2.36}$$

Note that this method is a modification to the Salton (or Cosine) index method, which only considers the percentage of common parents (2.35) between two nodes [SM83].

Observe that the similarity matrix S^C can be rewritten as

$$S^C = \left[C|D^T\right]\left[C|D^T\right]^T, \tag{2.37}$$

where C and D are the normalized rows and columns, respectively, of the adjacency matrix, i.e.,

$$C_{i,:} = \frac{A_{i,:}}{\sqrt{\sum_{k=1}^{n} A_{i,k}}}, \qquad \text{and} \qquad D_{:,j} = \frac{A_{:,j}}{\sqrt{\sum_{k=1}^{n} A_{k,j}}}. \tag{2.38}$$

2.4.4.2 Low-rank Approximation. As in Browet and Van Dooren's low-rank method for the neighborhood pattern similarity measure, a low-rank approximation of (2.37) was derived in [CLVD16] to reduce the total complexity of computing S. The low-rank similarity approximation scheme is defined as

$$\Pi^{(r)}\left[S^C\right] = \Pi^{(r)}\left[\left[C|D^T\right]\left[C|D^T\right]^T\right] = XX^T, \tag{2.39}$$

where $\Pi^{(r)}[\cdot]$ is the truncated SVD of dimension at most r on the factor $\left[C|D^T\right]$. That is,

$$\left[C|D^T\right] = U\Sigma V^T, \tag{2.40}$$

where Σ is a real $r \times r$ matrix of the r singular values, and $U \in \mathbb{R}^{n\times r}$, and $V \in \mathbb{R}^{n\times r}$ are orthogonal matrices [CLVD16]. Therefore, the low-rank factor X is defined as

$$X = U\Sigma. \tag{2.41}$$

Observe that (2.41) is easier to compute than the low-rank approximation of the neighborhood pattern similarity measure defined by Browet and Van Dooren since it is not an iterative algorithm. However, information may be lost when computing (2.41) since it does not consider neighborhoods of longer lengths (e.g., $\ell = 2$ or $\ell = 3$). For example, in Figure 2.4, we have 3 roles and roles **1** and **3** are almost isomorphic. However, while role **3** has pairwise neighbors in role **1**, it also has long neighborhood paths into role **2**. This information about the longer neighborhoods would be lost with the similarity measure of Cheng et al. and the nodes in roles **1** and **3** would be grouped into one role. Therefore, only 2 roles would be extracted from the similarity measure.

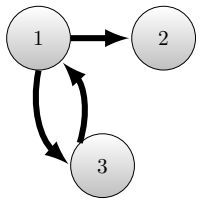


Figure 2.4: Example of role graph with two almost isomorphic roles.

2.5 Clustering Algorithms

In this section, we review some common clustering algorithms that can be used to solve the role extraction problem for either direct or indirect quality functions. While these algorithms are known

to find community structures of graphs and datasets, they can be used to find role structures due to the construction of the quality functions for role detection. However, as with the community detection problem, the algorithms presented are heuristics and may not find the same role partition. Despite this, these algorithms have been shown to work reasonably well for community and role detection.

We summarize three algorithms in this section, but additional community detection and clustering algorithms can be found in [KR05, Bro14, Tra14].

2.5.1 Simulated Annealing

We summarize simulated annealing in this section. Implementation details of the algorithm can be found in [KGV83, GSPA04, GMTA05, Tra14].

2.5.1.1 Summary of Simulated Annealing. Simulated annealing is a general discrete optimization technique to obtain a global optimum for a given cost function [KGV83]. The idea is to explore a large proportion of the feasible set at the beginning, and to narrow the search space as the algorithms progresses towards an optimum. That is, assume there exists two different states σ_1 and σ_2, where the current state is σ_1. Also, let $\Delta\mathcal{H}$ denote the variation of the quality function when there is a switch from state σ_1 to σ_2, i.e., $\Delta\mathcal{H}(\sigma_1, \sigma_2) = \mathcal{H}(\sigma_2) - \mathcal{H}(\sigma_1)$. Simulated annealing accepts a switch to state σ_2 with probability

$$\Pr(\sigma_1 \to \sigma_2) = \frac{1}{k} \exp(\beta \Delta\mathcal{H}(\sigma_1, \sigma_2)),$$

where k is a scaling parameter over all possible transitions from σ_1 [Jay57], i.e.,

$$k = \sum_{\sigma_i \in \Theta(\sigma_1)} exp(\beta \Delta\mathcal{H}(\sigma_1, \sigma_i)), \quad \Theta(\sigma_1) = \{\sigma \mid \sigma_1 \to \sigma\}.$$

The parameter $\beta = 1/T$ is the inverse temperature. The algorithm starts with a high temperature, i.e., β small, which allows for every transition to be chosen with almost equal probability. Then, as the algorithm progresses, the temperature slowly decreases such that the transitions with positive variation of the quality function are accepted with higher probability. As $\beta \to \infty$, only transitions with strictly positive variation of the cost function are accepted until no more such transitions exist. This method will always converge to a stable state that is an arbitrarily good approximation of the true optimum depending on the choice of the parameters [KGV83].

2.5.1.2 Implementation of Simulated Annealing by Guimerà et al.. Guimerà et al. in [GSPA04, GMTA05] applied an alternative form of simulated annealing to community detection. Their algorithm defined two types of transition, where the probability to accept a transition is given by

$$\Pr\left(\sigma_i \to \sigma_j\right) = \begin{cases} 1 & , \text{ if } \Delta\mathcal{H} > 0, \\ \frac{1}{k}\exp\left(\beta\Delta\mathcal{H}(\sigma_i, \sigma_j)\right) & , \text{ if } \Delta\mathcal{H} \leq 0. \end{cases}$$

The first transition type is a local node switch. That is, a random node is removed form its current community and assigned to a neighboring community based on the variation of the cost function. The second type of transition consists of either merging two communities into a new community, or splitting a community into two distinct communities [GSPA04, GMTA05]. The splitting of a community can be implemented in various ways, for example, by spectral bi-sectioning or by constrained simulated annealing of the subgraph induced by the community [Bro14, Tra14].

It has been shown that applying global transitions leads to a better approximation of the optimum than only applying local transitions. In practice, one usually considers n^2 local transitions and n global transitions before updating the temperature of the system. However, the total complexity of the algorithm cannot be determined, since it depends upon the initial temperature and the cooling process. Also, in general, simulated annealing is a very slow algorithm and is not usually recommended for graphs with more than 10^4 nodes [GSPA04, GMTA05, Bro14].

2.5.1.3 Implementation of Simulated Annealing by Reichardt and White. For the role extraction problem, Reichardt and White optimize their direct quality function function using simulated annealing because it yields high-quality results, is very general, and is simple to implement [RW07, Rei09]. The authors interpret their quality function $\mathcal{Q}_{RW}$ (which they maximize with respect to σ and B) as the negative of the modularity cost function, i.e., $\mathcal{H}(\{\sigma\}) = -\mathcal{Q}$. Then, the probability to transition from one state to another is given by [Rei09]

$$\Pr\left(\sigma_i \to \sigma\right) = \frac{\exp\left(-\beta\Delta\mathcal{H}(\sigma_i, \sigma)\right)}{\sum_{s=1}^{r}\exp\left(-\beta\Delta\mathcal{H}(\sigma_i, s)\right)}. \tag{2.42}$$

Suppose that the image matrix B of the network is given. Then the change in variation of the quality function is

$$\Delta\mathcal{H}(\sigma_i \to \sigma) = \sum_{s}^{q}\left(B_{\sigma_i,s} - B_{\sigma,s}\right)\left(k_{i\to s}^{out} - \gamma\left\langle k_{i\to s}^{out}\right\rangle\right) + \sum_{t}^{q}\left(B_{t,\sigma_i} - B_{t,\sigma}\right)\left(k_{t\to i}^{in} - \gamma\left\langle k_{t\to i}^{in}\right\rangle\right), \tag{2.43}$$

where $k^{out}_{i\to s} = \sum_j w_{i,j} A_{i,j} \delta_{\sigma_j,s}$ denotes the weight of outgoing edges from node i to nodes in role s and $\left\langle k^{out}_{i\to s} \right\rangle = \sum_j p_{i,j} \delta_{\sigma_j,s}$ denotes its respective expected value [Rei09]. Similarly, $k^{in}_{t\to i} = \sum_j w_{j,i} A_{j,i} \delta_{\sigma_j,t}$ denotes the weight of the incoming edges to node i from nodes in role t and $\left\langle k^{in}_{t\to i} \right\rangle = \sum_j p_{j,i} \delta_{\sigma_j,t}$ denotes its expected value [Rei09].

For undirected networks, the incoming and outgoing edges are equal; hence, the updating scheme only needs to assess the k_i neighbors of node i and determine which of the q roles is best for node i, which takes $O(q^2)$ operations. Thus, a local update is $O(\langle k \rangle + q^2)$ operations, and can be implemented efficiently for sparse graphs as long as the number of roles is much smaller than the number of nodes in the network [Rei09].

When B is unknown, the variation of the cost function is computed as

$$\begin{aligned} \Delta\mathcal{H}(\sigma_i \to \sigma) = &\sum_{s}^{q} |\alpha_{\sigma_i,s}| + |\alpha_{\sigma,s}| - |\alpha_{\sigma_i - i,s}| - |\alpha_{\sigma+i,s}| \\ &+ \sum_{t}^{q} |\alpha_{t,\sigma_i}| + |\alpha_{t,\sigma}| - |\alpha_{t,\sigma_i - i}| - |\alpha_{t,\sigma+i}|, \end{aligned} \tag{2.44}$$

where $\alpha_{s,t} = e_{s,t} - \gamma \langle e_{s,t} \rangle$, and the subscripts $\sigma_i - i$ denotes all nodes in role σ_i except node i and $\sigma + i$ denotes all nodes in role σ including node i [Rei09]. These coefficients can be computed efficiently, and the computational complexity does not change [Rei09].

2.5.2 Browet et al.'s Fast Community Detection Algorithm

In this section, we summarize Browet et al.'s community detection algorithm. Implementation details of the algorithm can be found in [BAD13, Bro14].

Browet et al.'s fast community detection algorithm [BAD13] is a synchronous version of the Louvain method developed by Blondel et al. [BGLL08]. The Louvain method is a greedy hierarchical clustering algorithm that is divided into two steps. First, each node is initialized as its own community. Then the correction step is performed where, based on the cost function, individual nodes are sequentially moved to one of their neighboring communities if the cost function produces positive gain. The second step (called the aggregation step) is when the graph is collapsed to create a new network based on the communities from the first phase after no correction with a strictly positive gain can be found for any node in the network. Thus, the aggregation step is implemented such that a graphs is collapsed to produce a new network with k nodes represented by the weighted adjacency matrix $\bar{W} = CWC^T$, where $W \in \mathbb{R}^{n\times n}$ is the weighted adjacency matrix of the graph

and C is known as a $k \times n$ community matrix such that $C_{i,j} = 1$ if community i contains node j and 0 otherwise.

In the Louvain method, these two phases are applied repeatedly to collapse the graphs until no more communities can be found. However, while this method has been seen empirically to produce good community structures, the time it requires to create a partition for large networks is high due to the sequential correction steps. Thus, Browet et al. proposed in [BAD13, Bro14] an alternative way to compute the correction steps, speeding up the algorithm.

Compared to the Louvain method, the major improvement of Browet et al.'s algorithm is that it initializes the community structure such that each node is assigned to its best neighbor based on the chosen cost function [BAD13,Bro14]. Then, the algorithm finds the graph's connected components, which are determined by two types correction steps that are applied repeatedly until the algorithm reaches a local optimum. The first type of correction is called the positive correction, which ensures that every nodes has at least a positive contribution to the cost function [Bro14, Section 3.4.2]. The second correction step is the maximal correction step, which assigns each node to the community which provides the largest gain based on the current partition [Bro14, Section 3.4.4].

The positive correction step has linear complexity in the number of edges within the community and a quadratic complexity in the number of nodes in the strongly connected components (SCC), $O(m_c + n_{SCC}^2)$ or $O(n_c + n_{SCC}^2)$ for sparse graphs. Note that the size of each SCC is generally much smaller than the size of the communities. Therefore, the quadratic term does not significantly increase the time for the algorithm [Bro14, Section 3.4.2].. The details of the storage of communities can be found in [Bro14, Section 3.4.3].

The maximal correction step only uses the information from the current community distribution and the community distribution is updated after all the assignment switches have been updated. Also, since the set of possible corrections for a node is constrained to its neighboring communities, then for each node we only need to look for its neighboring communities. Therefore, the complexity of the maximal correction is linear in the number of edges $O(m)$ [Bro14, Section 3.4.4].

The sequence of positive corrections followed by the maximal correction is repeated until all nodes are assigned to the community with maximal nonnegative gain in the current distribution. However, the edge weight distribution may prevent the iterative sequence from converging. For example, in situations where mutually attractive or repulsive nodes permanently switch their com-

munity assignments, an infinite sequence of positive and maximal corrections may occur. To avoid this occurrence, the authors designed the algorithm to accept each individual maximal correction with a probability less than 1 [BAD13, Bro14].

When no more corrections provide a strictly positive gain, the community graph is collapsed as in the aggregated step of the Louvain method and the aggregated graph is used as a new input graph for the procedure to provide a new hierarchical level of clustering. The algorithm is repeated until there is no community structure in the last collapsed graph, that is, the optimal community matrix in the aggregated graph is the identity. Unfortunately, the total complexity cannot be determined because it depends on the number of positive and maximal corrections applied, which cannot be estimated [Bro14, Section 3.4.4].

For the indirect methods to the role extraction problem, after the similarity matrix of the graph $G_A(V, E)$ has been computed, a community detection algorithm is implemented on the similarity matrix to extract the role partition by grouping highly similar nodes together. Any community detection quality function and algorithm can be used to extract the roles (see [Bro14, Tra14] for details on different community detection quality functions and algorithms).

Browet and Van Dooren determine the role structure of the graph by extracting the roles from the low-rank approximation of neighborhood pattern similarity measure. That is, the authors find a permutation matrix P such that $S^{(r)} = (PX)(PX)^T$ (where $X \in \mathbb{R}_*^{n\times r}$ is the low-rank factor of the similarity matrix) is a block matrix and PAP^T is the relabeled graph where the edges are primarily concentrated into particular blocks. However, to implement their a community detection algorithm, the authors require the matrix $S^{(r)} = XX^T$ [BAD13, Bro14]. While using community detection is not dependent upon the numerical rank r of the similarity matrix S, the formulation of $S^{(r)}$ is $O(n^2r)$ in complexity and may be costly if the number of nodes or roles is large.

2.5.3 K-means Clustering

In this section, we summarize k-mean clustering. Implementation details of the algorithm can be found in [AV07, HTF09].

K-means clustering (or Lloyd's algorithm [Llo82]) is a well-known iterative, data-partitioning method of vector classification that finds the clusters and cluster centers in a set of unlabeled data. Given a number of clusters k, the algorithm assigns n observations into the k clusters defined by cluster centers (or centroids) by iteratively moving the centers to minimize the sum of distance

functions of each point in the cluster to the center [HTF09]. The k-means algorithm alternates between the two steps:

(1) for each center, identify the subset of points that are closer to the center and group them together as a new cluster;

(2) minimize the sum of squared distances of the observations in each cluster to obtain k new cluster centers.

These two steps are iterated until the cluster assignments no longer change or until the maximum number of iterations has been reached.

Let the rows of X be the set of n observations, i.e., $\{x_1, x_2, \ldots, x_n\}$, the k-means algorithm assigns each row into one of the k clusters C_i for $i = 1, \ldots, k$ to minimize the sum of distance functions of each point in the cluster to the center, i.e., k-means solves the optimization problem

$$\underset{\{C_j\}_{j=1}^{k}}{\arg\min} \sum_{j=1}^{k} \sum_{i \in C_j} d(x_i, \mu_j), \tag{2.45}$$

where μ_j is the mean of points in cluster C_j and $d(x_i, \mu_j)$ is the distance between observation x_i and mean μ_j [HTF09].

The complexity of the k-means algorithm is $O(I_{iter}knd)$, where I_{iter} is the number of iterations needed for convergence and d is the dimension of the vectors. Note that the convergence of k-means is dependent upon the initial choice cluster centers. Typically, the initial centers are either chosen randomly from the observations or computed using the k-means++ algorithm developed by Arthur and Vassilvitskii in [AV07]. The steps of the k-means++ algorithm are as follows: assuming k clusters,

(1) choose one observation uniformly from X and set it as the first center;

(2) for each observation $x_i \in X$, compute the distance between x_i and the first center;

(3) select each subsequent center with a probability proportional to the distance from itself to the closest center that has already been chosen;

(4) repeat steps 2 and 3 until k centers have been chosen.

Arthur and Vassilvitskii showed in [AV07] that k-means++ algorithm improves the complexity of the Lloyd's algorithm and the quality of the final solution. The complexity of the randomized greedy strategy is $O(nkd)$ and is $O(\log(k))$-competitive with the optimal clustering.

A limitation of k-means is that it only considers the distance between the means and the data points and has no representation of the weight or breadth of each cluster. Also, k-means clustering has no way of representing the size or shape of each cluster. Lastly, k-means clustering requires the user to assume the number of clusters beforehand. However, there are various clustering statistics that we can use to determine the optimal number of clusters. We describe two popular clustering statistics in the next section.

2.6 Clustering Statistics

Unlike with community detection algorithms, which require the full similarity matrix $S^{(r)} = XX^T$, k-means clustering may be applied on the low-rank factor $X \in \mathbb{R}_*^{n\times r}$ of the similarity matrix. One way to do this is group the rows of X into k roles using k-means clustering. Determining the number of roles is an important issue. We show in Chapter 3 that for certain types of reduced graph structures, the numerical rank of the matrix S corresponds to the number of roles in the graph $G_A(V, E)$. In practice, finding the optimal number of roles is difficult.

Cheng et al. in [CLVD16] suggest using a hierarchical k-means clustering method to extract the role structure. Their algorithm assumes an upper bound on the number of clusters, k, and finds clusters such that the rows of X within each cluster are orthogonal to each other. The authors iterate this step until the centroids of the different clusters satisfy the orthogonality criterion.

Once they have determined the optimal number of clusters, they used k-means clustering to extract the role assignment. A problem with this approach is that for noisy graphs, the elements in different roles may not be orthogonal to each other and the algorithm struggles to extract the role partition. Also, the size of each role is very important. If the sizes of the roles are not almost homogeneous, i.e., the same size, then the algorithm fails to extract the role partition [CLVD16]. Lastly, their approach requires an overestimation of the number of roles. However, this is a concern for all indirect role extraction methods that use low-rank approximation methods to construct their quality function. Therefore, we suggest using popular clustering statistics, such as the silhouette or gap statistic, to determine the optimal role partition.

2.6.1 Silhouette Statistic

Rousseeuw proposed the silhouette statistic in [Rou87] to assess clusters and to estimate the optimal number of clusters. The silhouette value for each observation measures how similar the

observation is to points within its own cluster in comparison with points in different clusters [Rou87, KR05].

To compute the silhouette value, take an observation i and assign it to a cluster (denoted $\mathcal{A}$). Then, compute the average distance of i to all other observations in $\mathcal{A}$. Note that this can only be computed when $\mathcal{A}$ contains observations other than i, i.e., $\mathcal{A}$ is not a singleton. Next, consider any other cluster C not equal to cluster $\mathcal{A}$ and define the average distance of i to all observations in C as $d(i, C)$. Compute the average distance $d(i, C)$ for all clusters $C \neq \mathcal{A}$ and select the minimum average distance, i.e., $b(i) = \min_{C \neq \mathcal{A}} d(i, C)$. The cluster $\mathcal{B}$ for which the minimum is attained, i.e. $d(i, \mathcal{B}) = b(i)$, is called the neighbor of observation i [KR05]. The silhouette statistic is defined as

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}. \tag{2.46}$$

Note that $b(i)$ depends on clusters not equal to $\mathcal{A}$ being available. Hence, the silhouette value is not defined if there is only one cluster [KR05].

The silhouette values are between -1 and 1, where a high silhouette value close to 1 that observation i is well classified in its current cluster and poorly matched to its neighboring clusters. In other words, the second best cluster choice $\mathcal{B}$ is not as close to the actual cluster $\mathcal{A}$. If the silhouette value is around 0, then it is unclear whether i should be assigned to cluster $\mathcal{A}$ or cluster $\mathcal{B}$. If $s(i)$ is close to -1, then i is closer to $\mathcal{B}$ than $\mathcal{A}$ and it is assumed that observation i has been misclassified [KR05].

If most of the observations have high silhouette values, then the clustering solution is ideal. However, if many of the points have low (or negative) silhouette values, then the clustering solution may have either too many (or too few) clusters [Rou87, KR05]. Note that the silhouette statistic can be used with any distance metric.

Kaufman and Rousseeuw proposed to choose the optimal number of clusters $\hat{k}$ as the value maximizing the average $s(i)$ over the data set [KR05]. This value is called the silhouette coefficient (SC).

For the role extraction problem, we use k-means clustering on the low-rank factor $X \in \mathbb{R}^{n \times r}$ and assume that the number of roles for the silhouette statistic are $\{2, \ldots, r+1\}$ where r is the rank. We choose the optimal number of roles as the number of roles that maximizes the average silhouette value over the data set.

We use the subjective interpretation of the silhouette coefficient proposed by Kaufman and Rousseeuw in [KR05] to evaluate the quality of the role partition, and if the silhouette coefficient is between 0.71 and 1, then a strong role structure has been found. If the silhouette coefficient is between 0.51 and 0.70, then a reasonable role structure has been found. A silhouette coefficient between 0.26 and 0.50 indicates that the structure is weak and alternative clustering methods should be tested on the data set. Lastly, a silhouette coefficient less than 0.25 indicates that no substantial structure has been found for the data set [KR05].

2.6.2 Gap Statistic

An alternative statistic for estimating the number of clusters in a set of data is the gap statistic proposed by Tibshirani et al. in [TWH01]. In this section, we summarize the gap statistic and refer the reader to [TWH01] for implementation details.

A common graphical approach to determine the number of clusters is to plot an error measurement versus several proposed number of clusters and locate the "elbow" of the plot, where the "elbow" occurs at the most dramatic decrease in error measurement [Sug98, SLO99]. For example, Figure 2.5b displays a typical plot of the within-cluster dispersion error measure W_k (defined below) versus the number of clusters. The error measure W_k decreases monotonically as the number of clusters increase, but from $k = 3$ onwards the decrease flattens. Thus, $k = 3$ is the "elbow" and indicates the appropriate number of clusters.

The gap statistic formalizes this heuristic by estimating the location of the "elbow" as the number of clusters with the largest gap value (e.g., Figure 2.5c). Thus, the optimal number of clusters occurs at the solution with the largest local, or global, gap value within some tolerance range [TWH01].

The gap statistic is defined as

$$\text{Gap}(k) = E_n^*[\log(W_k)] - \log(W_k), \tag{2.47}$$

where n is the sample size, k is the number of clusters being evaluated, $E_n^*[\cdot]$ is the expected value, and W_k is the cluster dispersion measurement given by

$$W_k = \sum_{l=1}^{k} \frac{1}{2n_l} D_l, \tag{2.48}$$

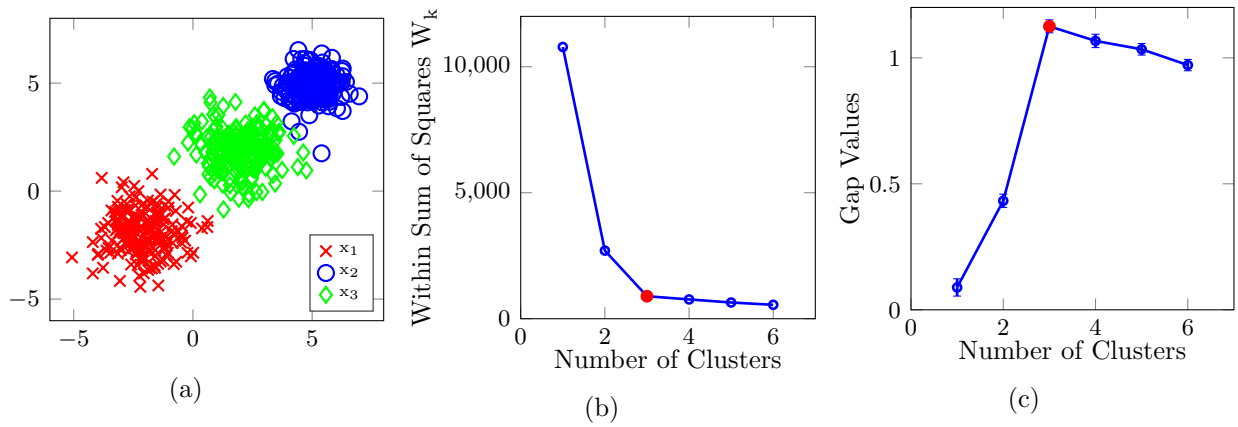


Figure 2.5: Results of three-cluster example: (a) data; (b) within sum of squares function W_k; and (c) gap curve. The red circle highlights the optimal number of clusters for the dataset.

where n_l is the number of points in cluster l and D_l is the sum of pairwise distances for all points in cluster l [TWH01]. The expected value $E_n^*[\log(W_k)]$ is determined by Monte Carlo sampling from a reference distribution, and $\log(W_k)$ is computed from the sample data.

The reference data can either be generated from a uniform distribution over a box aligned with the principal components of the data matrix X or generated uniformly over a range of X. Tibshirani et al. showed empirically that the gap statistic using the uniform reference distribution from the principal component orientation was able to determine the optimal number of clusters better than the gap statistic using the uniform reference distribution over a range of each feature [TWH01].

The optimal number of clusters $\hat{k}$ is the smallest number of clusters satisfying

$$\text{Gap}(\hat{k}) \geq \text{Gapmax} - SE(\text{Gapmax}), \tag{2.49}$$

where Gapmax is the largest gap value and $SE(\text{Gapmax})$ is the standard deviation corresponding to the largest gap value [TWH01]. Note that the gap statistic can be used with any distance metric and any clustering algorithm.

Unlike the silhouette statistic, the gap statistic is defined for solutions that contain only one cluster. However, the gap statistic is more computationally expensive since the clustering algorithm must be applied to the reference data for each proposed clustering solution [TWH01].

For the role extraction problem, we use k-means clustering on the low-rank factor $X \in \mathbb{R}^{n \times r}$ and assume that the number of roles for the gap statistic are $\{1, \ldots, r+1\}$ where r is the rank. Then, we choose the optimal number of roles as the smallest number that satisfies (2.49).

CHAPTER 3

ANALYSIS OF NEIGHBORHOOD PATTERN SIMILARITY MEASURE

For the role extraction problem, the idea of using a low-rank projection of a similarity measure for the construction of the indirect quality function was proposed by Browet and Van Dooren in [Bro14, BD14]. However, Browet and Van Dooren only provided empirical evidence of the relationship between the rank of the neighborhood pattern similarity measure and the number of roles in the network. In this chapter, we analyze this relationship and prove that for an adjacency matrix with complete blocks (additionally rank-1 for a weighted adjacency matrix), that the rank of the neighborhood pattern similarity measure is equal to the number of roles under certain assumptions about the structure of the image matrix B and the rank of the adjacency matrix.

Browet and Van Dooren compared their similarity measure with the similarity measure of Blondel et al. [BGH$^+$04] and the similarity measure of Cooper and Barahona [CB11] and showed empirically for an unweighted regular block cycle graph (where each role contains the same number of nodes and each node is connected to all of the nodes in the next role of the cycle) that the neighborhood pattern based similarity measure computed a similarity measure of rank equal to the number of roles in the network while the similarity measures by Blondel et al. and Cooper and Barahona both computed similarity measures of rank 1 [Bro14].

Consider an $n \times n$ weighted adjacency matrix W with q blocks where the rank of each block is 1 and W can be written in the form

$$W = ZBZ^T, \tag{3.1}$$

where Z is given by the $n \times q$ matrix

$$Z = \begin{bmatrix} z_1 & 0 & \cdots & 0 \\ 0 & z_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z_q \end{bmatrix}, \tag{3.2}$$

z_q denotes a real, positive vector of length n_q, $n = n_1 + n_2 + \cdots + n_q$, and B is the $q \times q$ image matrix. Observe that $Z^TZ = \text{Diag}(||z_1||^2, ||z_2||^2, \ldots, ||z_q||^2)$ where $\text{Diag}(\cdot)$ denotes a diagonal

matrix with the inner products of the z_q's on the diagonal, and the ranks of Z and Z^TZ is q. From this representation of W, we have the following lemmas on the rank of B and the rank of the neighborhood pattern similarity matrix.

To prove the lemmas, we use the following rank properties of real matrices [HJ90]: assuming $C \in \mathbb{R}^{m\times n}$,

(i) $\text{rank}(C) \leq \min(m, n)$;

(ii) $\text{rank}(C) = n -$ dimension of the null space of C, denoted $\dim(\text{Null}(C))$;

(iii) $\text{rank}(C) = \text{rank}(C^T) = \text{rank}(CC^T) = \text{rank}(C^TC)$;

(iv) if $D \in \mathbb{R}^{n\times p}$ is an $n \times p$ matrix where $n \leq p$ and $\text{rank}(D) = n$, then $\text{rank}(CD) = \text{rank}(C)$;

(v) if E is an $l \times m$ matrix where $l \geq m$ and $\text{rank}(E) = m$, then $\text{rank}(EC) = \text{rank}(C)$;

Lemma 3.0.1 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. Then,*

$$\text{rank}(W) = \text{rank}(B) \leq q. \tag{3.3}$$

Proof Let $\tilde{B} = BZ^T$. Then by rank property (iv) and that $\text{rank}(Z) = \text{rank}(Z^T) = r$, $\text{rank}(\tilde{B}) = \text{rank}(BZ^T) = \text{rank}(B)$. Now, let $W = ZBZ^T = Z\tilde{B}$. Then by rank property (v), $\text{rank}(W) = \text{rank}(Z\tilde{B}) = \text{rank}(B)$. Since B is a $q \times q$ matrix, then $\text{rank}(W) = \text{rank}(B) \leq q$. ∎

Also, the rank of the neighborhood patterns of length 1, S_1, is equal to the rank of B augmented with B^T

Lemma 3.0.2 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. Then*

$$\text{rank}(S_1) = \text{rank}([B\ B^T]). \tag{3.4}$$

Proof Note that

$$\begin{aligned}\text{rank}(S_1) &= \text{rank}\left(WW^TW^TW\right) = \text{rank}\left(ZBZ^TZB^TZ^T + ZB^TZ^TZBZ^T\right)\\ &= \text{rank}(BZ^TZB^T + B^TZ^TZB) = \text{rank}\left(\begin{bmatrix}B & B^T\end{bmatrix}\text{Diag}\left(Z^TZ, Z^TZ\right)\begin{bmatrix}B^T\\ B\end{bmatrix}\right).\end{aligned}$$

Then, by properties iii and iv,

$$\begin{aligned}\operatorname{rank}\left(\begin{bmatrix}B & B^T\end{bmatrix}\operatorname{Diag}\left(Z^TZ, Z^TZ\right)\begin{bmatrix}B^T\\ B\end{bmatrix}\right) &= \operatorname{rank}\left(\begin{bmatrix}B & B^T\end{bmatrix}\operatorname{Diag}\left(Z^T, Z^T\right)\right)\\ &= \operatorname{rank}\left(\begin{bmatrix}B^T & B\end{bmatrix}\right). \quad \blacksquare\end{aligned}$$

In addition, the ranks of the neighborhood pattern similarity matrices S_k defined by (2.22) remain constant for all $k \geq 1$ as stated in the following lemma.

Lemma 3.0.3 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. Then the ranks for all iterates S_k of the neighborhood pattern similarity measure defined by* (2.22) *remains constant for all $k \geq 1$.*

Proof Let $W = ZBZ^T$ where Z is an $n \times q$ matrix given by (3.2) and B is the $q \times q$ image matrix. Then the $\operatorname{rank}(Z) = q$ and $\operatorname{rank}(W) = \operatorname{rank}(B) = \hat{q} \leq q$ by Lemma 3.0.1. Moreover, we can replace iteration (2.22) by a reduced iteration of dimension $q \times q$ as follows:

Define $N^2 = Z^TZ = \operatorname{Diag}(||z_1||^2, ||z_2||^2, \ldots, ||z_q||^2)$. Then

$$S_k = Z\hat{S}_kZ^T, \quad \forall k \geq 1 \tag{3.5}$$

with

$$\begin{aligned}\hat{S}_1 &= \Gamma_B\left[N^2\right]\\ \hat{S}_{k+1} &= \hat{S}_1 + \beta^2\Gamma_B\left[N^2\hat{S}_kN^2\right],\end{aligned} \tag{3.6}$$

where $\Gamma_B(\hat{X}) = B\hat{X}B^T + B^T\hat{X}B$. Moreover, the conditions on β for (2.22) to converge are the same for (3.6) since (3.6) is a rewriting of (2.22), not an approximation.

Note that if conditions (2.25) or (2.26) on β are satisfied, then the matrices S_k (and $\hat{S}_k$) for all $k \geq 1$ remain bounded and satisfy in the limit

$$\begin{aligned}S_* &= S_1 + \beta^2\Gamma_W\left[S_*\right]\\ \hat{S}_* &= \hat{S}_1 + \beta^2\Gamma_B\left[N^2\hat{S}_*N^2\right].\end{aligned}$$

Since all of the terms are semidefinite, it follows that

$$\begin{aligned}\operatorname{rank}(S_k) &\geq \operatorname{rank}(S_1)\\ \operatorname{rank}(\hat{S}_k) &\geq \operatorname{rank}(\hat{S}_1),\end{aligned}$$

including S_* and $\hat{S}_*$, i.e., the case $k = \infty$.

Now, we show that all of the ranks of the matrices are constant. Let V be the nullspace of $S_1 \succeq 0$, i.e., $V^T(W^TW + WW^T)V = 0$. Then it follows that $WV = 0$ and $W^TV = 0$ since

$$V^T \begin{bmatrix} W^T & W \end{bmatrix} \begin{bmatrix} W \\ W^T \end{bmatrix} V = 0.$$

Clearly, $\Gamma_W[S_k]V = 0$, and hence $S_{k+1}V = 0$. Since the rank cannot decrease, it must remain constant. ∎

Lastly, we prove a relationship between the rank of the neighborhood pattern similarity measure S_*, the rank of B, and the number of roles in the role graph.

Theorem 3.0.4 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by (3.2) and B is the $q \times q$ image matrix. Then,*

$$\frac{\operatorname{rank}(S_*)}{2} \leq \operatorname{rank}(B) \leq \operatorname{rank}(S_*) \leq q. \tag{3.7}$$

Proof By Lemmas 3.0.2 and 3.0.3, we have that $\operatorname{rank}(S_1) = \operatorname{rank}([B\ B^T]) = \operatorname{rank}(S_*)$. Thus,

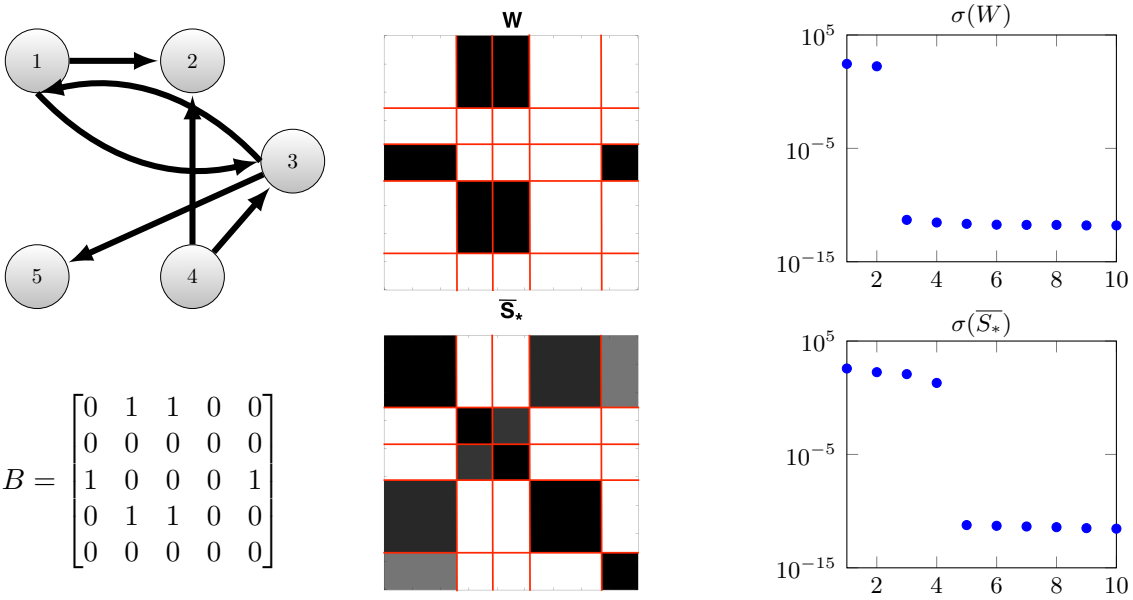
$$\operatorname{rank}(B) \leq \operatorname{rank}([B\ B^T]) \leq 2\operatorname{rank}(B)$$

by the column space of B. Therefore,

$$\frac{\operatorname{rank}(B)}{2} \leq \frac{\operatorname{rank}(S_*)}{2} \leq \operatorname{rank}(B) \leq \operatorname{rank}(S_*) \leq q. \quad ∎$$

Theorem 3.0.4 states that the rank of the image matrix B cannot be less than half of the rank of S_*. Also, the theorem states that the rank of S_* is less than or equal to the number of roles q. However, even if the rank of S_* can be less than q, the structure of S_* may still reveal the number of roles such that the clustering algorithms may extract the role partition. For example, in Figure 3.1, the role graph has 5 roles, but the ranks of B and W is 2 and the rank of S_* is 4. However, observe that there are 5 blocks on the diagonal of S_*. Therefore, S_* can be partitioned into 5 blocks and the 5 role structure can be extracted from S_*. (Note that for visual clarity, we show the matrix $\overline{S_*}$, which is the similarity matrix after it has been diagonally scaled (2.27). However, the rank of $\overline{S_*}$ is equal to the rank of S_*, so the argument still holds.)

From Theorem 3.0.4, we have the following corollaries.



$$B = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 3.1: Example of rank of W and $\overline{S_*}$ when the rank$(S_*) < q$. 10 largest singular values of W and $\overline{S_*}$ where the blocks of W are complete.

Corollary 3.0.5 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. If* rank$(B) = q$*, the number of roles, then the rank of the neighborhood pattern similarity measure defined by* (2.22) *is q.*

Proof If rank$(B) = q$, then, by Theorem 3.0.4, the rank$(S_*) = q$. ∎

Corollary 3.0.6 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with q blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. If B is symmetric and* rank$(B) \leq q$*, then rank of the neighborhood pattern similarity measure defined by* (2.22) *is equal to* rank(B)*.*

Proof Suppose that $B = B^T$ and rank$(B) \leq q$. Recall from Lemmas 3.0.2 and 3.0.3 that rank(S_*) = rank(S_1) = rank$(BN^2B^T + B^TN^2B)$, where rank(BN^2B^T) = rank(B^TN^2B) = rank$(B) \leq q$. Since $B = B^T$, then $BN^2B^T + B^TN^2B = 2BN^2B^T$ and rank(S_*) = rank(S_k) = rank(S_1) = rank(BN^2B^T) = rank(B). ∎

Corollary 3.0.5 states that if the rank of the image matrix B is equal to the number of roles, then the rank of the neighborhood pattern similarity measure S_* is equal to the number of roles.

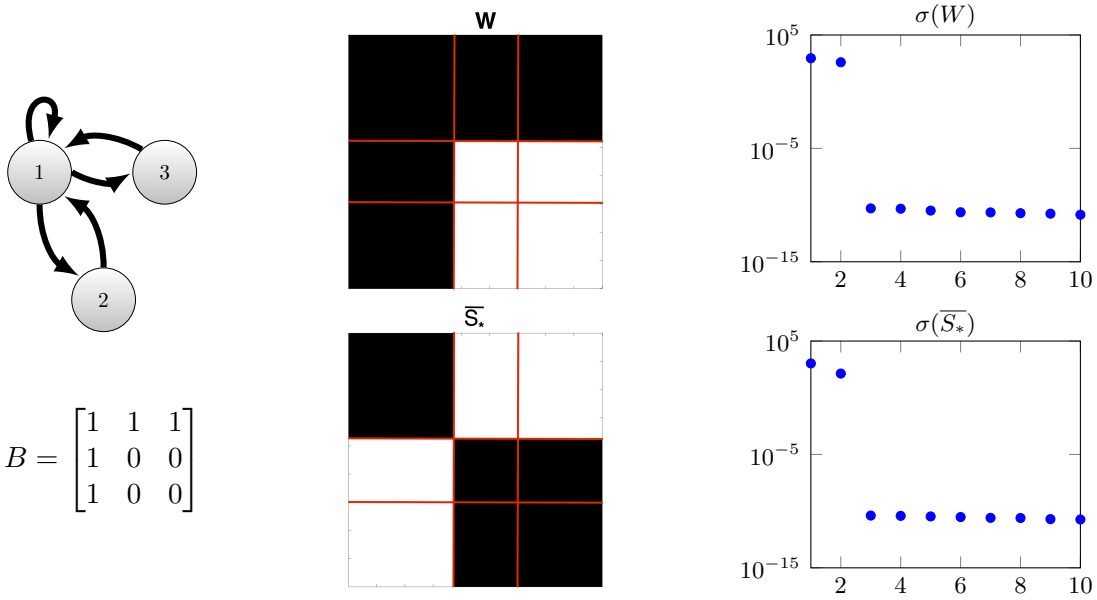
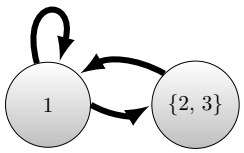


Figure 3.2: Example of rank of W and $\overline{S_*}$ when the image matrix B is symmetric. 10 largest singular values of W and $\overline{S_*}$ where the blocks of W are complete.

This is true regardless of the structure of matrix B. The structure of the image matrix becomes important when the rank of B is less than the number of roles. If the rank of B is less than the number of roles and B is symmetric, then by Corollary 3.0.6 the ranks of B, W, and S_* are equal. For example, in Figure 3.2, the image matrix B is symmetric and the ranks of W and S_* are 2.

Remark In the literature, there is a debate whether the image graph in Figure 3.2 is a good representation of the data since roles **2** and **3** are structurally equivalent. Reichardt and White in [RW07] state that a role graph cannot have structurally equivalent roles, i.e., roles **2** and **3**, while Doreian et al. in [DBF05, Section 7.5] define Figure 3.2 as a core-periphery model. That is, there is one core role that is connected with itself and with all of the other roles. Also, all other roles (periphery) are connected to the core role, but are not connected to each other nor are they connected with themselves [DBF05, Section 7.5].

Throughout this dissertation, we use Reichardt and White's constraint that no two nodes in the role graph may be structurally equivalent. This constraint is implicitly imposed by the neighborhood pattern similarity measure, since the similarity measure is not able to distinguish between the two structurally equivalent roles. Thus, if the role structure has been maliciously chosen without



$$\tilde{B} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Figure 3.3: Alternative role graph and image matrix $\tilde{B}$ for Figure 3.2.

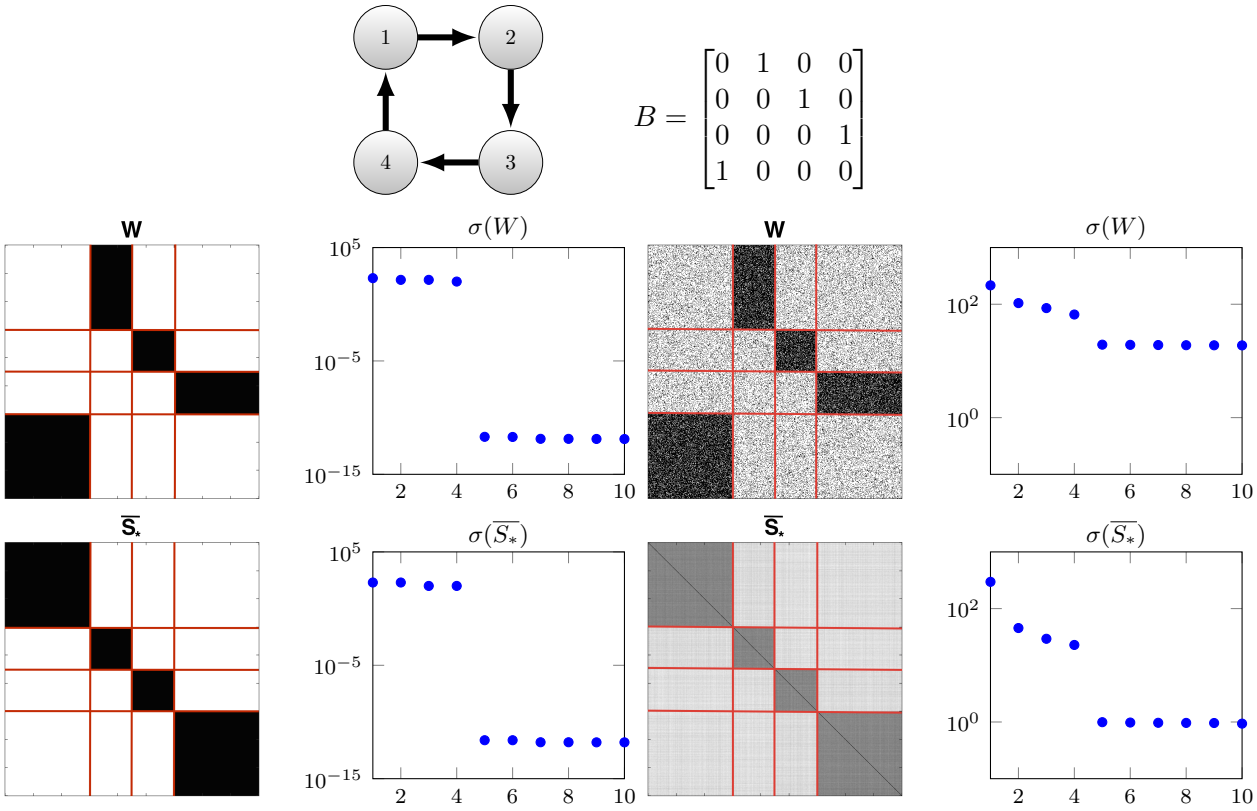
considering the role constraint (which would be the case for Corollary 3.0.6), the neighborhood pattern similarity measure reveals the true role structure. That is, for the graph $G_W(V_W, E_W)$, we can find another factorization $W = \tilde{Z}\tilde{B}\tilde{Z}^T$, where, for $\tilde{q} < q$, the $\tilde{q} \times \tilde{q}$ image matrix $\tilde{B}$ satisfies the role constraint and $\tilde{Z}$ is an $n \times \tilde{q}$ matrix (see Figure 3.3).

If we impose the role constraint that no two roles in the role graph can be structurally equivalent, then that implies that if rows i and j in image matrix B are equal, then columns i and j cannot be equal (or vice versa). If rows i and j are equal and columns i and j are equal, then roles σ_i and σ_j are structurally equivalent. Note that enforcing this constraint when B is symmetric is equivalent to forcing B to be nonsingular. Enforcing this constraint in Corollary 3.0.6 gives Corollary 3.0.7.

Corollary 3.0.7 *Given an $n \times n$ nonnegative, weighted adjacency matrix W with r blocks that can be written as $W = ZBZ^T$, where Z is an $n \times q$ matrix given by* (3.2) *and B is the $q \times q$ image matrix. If B is symmetric and and no two roles are structurally equivalent, then the rank of the neighborhood pattern similarity measure defined by* (2.22) *is q.*

Proof Suppose B is symmetric and no two roles are structurally equivalent, that is B is nonsingular. Then $\text{rank}(B) = q$ and the rank of S_* is q, which follows from Corollary 3.0.5. ∎

Also, note that Theorem 3.0.4 is only true if the blocks in the adjacency matrix are complete (additionally the blocks are all rank-1 for weighted graphs). For example, in Figure 3.4a, the blocks of the adjacency matrix W are complete and the ranks of W and S_* are 4. However, if the blocks of W are regular blocks and there exist elements in the null blocks, Theorem 3.0.4 fails because the blocks of the adjacency matrix are not rank-1 (see Figure 3.4b). That is, the rank of the similarity measure is not equal to the number of roles. However, while the number of roles is not equivalent to the rank of the similarity matrix, there is still approximately a factor of 10 difference between the third and fourth singular values and the similarity measure still has three blocks on the diagonal



(a) complete blocks

(b) regular blocks

Figure 3.4: Block cycle role structure, associated neighborhood pattern similarity, and 10 largest singular values of W and $\overline{S_*}$.

(see Figure 3.4b). Thus, it may be still possible to extract the role partition from an numerical rank based approximation of the similarity measure.

Observe that Theorem 3.0.4 gives us an idea of the minimal rank necessary for Algorithm 1 to extract the role structure for a weighted adjacency matrix with q blocks where the rank of each block is 1. Browet and Van Dooren showed empirically for unweighted random Erdós-Rényi directed graphs with regular blocks, i.e., blocks that are not rank-1, that the minimal rank necessary for Algorithm 1 to extract the role structure is related to the number of roles [Bro14, BD14]. However, for graphs with regularly equivalent roles, a rigorous proof of the relationship between the rank of the similarity matrix and the number of roles in the graph is needed.

Also, since the rank of the similarity matrix may be less than the number of roles, the ability to adjust the rank of the similarity matrix such that the clustering algorithm can extract the role

partition is needed in the case the initial rank is assumed to be too small, or too large, in relation to the number of roles in the network. We discuss the strengths of a rank-adaptive approach to the role extraction problem in Chapter 7.

CHAPTER 4

TWO-PHASE INDIRECT ROLE EXTRACTION METHOD

In this chapter, we describe a Riemannian optimization approach for approximating the low-rank neighborhood pattern similarity measure. In Section 4.1.1, we propose a cost function for the similarity measure that is maximized over the symmetric positive semidefinite fixed-rank Riemannian manifold. The manifold framework is discussed and the Riemannian objects required to apply Riemannian optimization algorithms are stated in Sections 4.1.2, 4.1.3 and 4.1.4. In Section 4.2, we motivate using k-means clustering with the silhouette statistic for the second phase of our indirect approach to the role extraction problem.

4.1 Phase I: Riemannian Optimization Approach to Low-rank Neighborhood Pattern Similarity Approximation

The first phase of the indirect approach to the role extraction problem is to compute the similarity measure of the graph. Motivated by Browet and Van Dooren's low-rank approach to the role extraction problem, we construct our quality function by a low-rank approximation of the neighborhood pattern similarity measure (see Section 2.4.3). However, we approximate the low-rank similarity matrix by Riemiannian optimization on the symmetric positive semidefinite fixed-rank manifold. This improves time and complexity of the similarity measure when computing longer neighborhood patterns or in the presence of graph noise (which we show empirically in Section 6.1).

4.1.1 Cost Function Derivation

In order to replace the low-rank algorithm (Algorithm 1) with a Riemannian optimization approach, we must first write the low-rank iterative scheme (2.28) in a form that is related to optimization and deduce a cost function. Observe that (2.28) can be rewritten as

$$S_{k+1}^{(r)} = \Pi^{(r)}\left[S_k^{(r)} + \left(S_1^{(r)} - S_k^{(r)} + \beta^2\Gamma\left[S_k^{(r)}\right]\right)\right] = \Pi^{(r)}\left[S_k^{(r)} + \operatorname{grad} f\left(S_k^{(r)}\right)\right],$$

where grad denotes the Euclidean gradient of $f(S)$ and

$$f(S) = \text{trace}\left(S^T\left(S_1^{(r)} - \frac{1}{2}S + \beta^2 ASA^T\right)\right) = \left\langle S, S_1^{(r)} - \frac{1}{2}S + \beta^2 ASA^T\right\rangle_F, \tag{4.1}$$

where trace$(\cdot)$ denotes the trace operator, i.e., for matrix $A \in \mathbb{R}^{n\times n}$, $\text{trace}(A) = \sum_{i=1}^n a_{i,i}$, and $\langle\cdot,\cdot\rangle_F$ denotes the Frobenius inner product. One can verify this by considering the directional derivative of f along $Z \in \mathbb{R}^{n\times n}$, that is,

$$\begin{aligned} Df(S)[Z] =& \text{ trace}\left({S_1^{(r)}}^T Z - \frac{1}{2}(Z^T S + S^T Z) + \beta^2(AZA^T S^T + ASA^T Z^T)\right) \\ =& \text{ trace}\left(Z^T(S_1^{(r)} - S + \beta^2(A^T SA + ASA^T))\right) \\ =& \left\langle Z, S_1^{(r)} - S + \beta^2\Gamma[S]\right\rangle_F. \end{aligned}$$

Thus, the Euclidean gradient of f is, as desired,

$$\text{grad}\, f(S) = S_1^{(r)} - S + \beta^2\Gamma[S]. \tag{4.2}$$

If the similarity matrix S is not full-rank, i.e., $S = XX^T$ where $X \in \mathbb{R}^{n\times r}$ and $\text{rank}(X) = r$ then S can be viewed as an element of the symmetric positive semidefinite fixed-rank manifold $\mathcal{S}_+(n,r)$, where $\mathcal{S}_+(n,r)$ is defined as

$$\begin{aligned} \mathcal{S}_+(n,r) &= \{S \in \mathcal{S}^n \mid S \succeq 0,\ \text{rank}(S) = r\} \\ &= \left\{XX^T \mid X \in \mathbb{R}^{n\times r},\ \text{rank}(X) = r\right\} \end{aligned} \tag{4.3}$$

where $\mathcal{S}^n$ is the set of symmetric matrices of $\mathbb{R}^{n\times n}$, i.e.,

$$\mathcal{S}^n = \left\{S \in \mathbb{R}^{n\times n} \mid S = S^T\right\}. \tag{4.4}$$

.

Finally, given the cost function $f : \mathcal{S}^n \to \mathbb{R} : S \mapsto f(S)$, we define the function $F : \mathbb{R}_*^{n\times r} \to \mathbb{R} : X \mapsto F(X)$ where

$$F(X) = f(XX^T) = \left\langle XX^T, X_1X_1^T - \frac{1}{2}XX^T + \beta^2 AXX^T A^T\right\rangle_F, \tag{4.5}$$

and the optimization problem

$$\max_{X\in\mathbb{R}_*^{n\times r}} F(X) \tag{4.6}$$

is considered for the low-rank approximation of the neighborhood pattern similarity matrix.

Observe that (4.6) is invariant by right-multiplication of X by orthogonal matrices Q of size $r \times r$. Hence, we need to remove the degeneracy of the critical points. To do this, we take $\mathcal{S}_+(n,r) := \mathbb{R}_*^{n\times r}/\mathcal{O}_r$ where $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is a quotient manifold. Therefore, (4.6) can be rewritten on the quotient manifold $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ as

$$\max_{\pi(X)\in\mathbb{R}_*^{n\times r}/\mathcal{O}_r} F_r(\pi(X)) \tag{4.7}$$

where the function $F_r : \mathbb{R}_*^{n\times r}/\mathcal{O}_r \to \mathbb{R} : \pi(X) \mapsto F_r(\pi(X)) = F(X)$ and $\pi : \mathbb{R}_*^{n\times r} \to \mathbb{R}_*^{n\times r}/\mathcal{O}_r : X \mapsto \pi(X)$ is the quotient map that maps $X \in \mathbb{R}_*^{n\times r}$ to its equivalence class $[X]$.

4.1.2 Symmetric Positive Semidefinite Fixed-Rank Manifold as Quotient Manifold

The symmetric positive semidefinite fixed-rank manifold is a popular Riemannian manifold that has been used in various fields (e.g., signal and image processing). It is well-known that the manifold is a smooth manifold and that there exist several different geometries: an embedded submanifold in $\mathbb{R}^{n\times n}$ [HM94, HS95, KL07, OHM06, VAV09, VV10], a quotient manifold of $\mathbb{R}^{n\times r}$ [AILH09, BMS10, JBAS10, MBS11], a quotient manifold of the Stiefel manifold and the symmetric positive definite manifold [BMS10, BS10, MJBS09], and a quotient manifold of the general linear group [VAV12]. Since the neighborhood pattern based similarity measure defined by Browet and Van Dooren is a symmetric positive semidefinite or positive definite matrix [Bro14, BD14], we can derive a cost function from Browet and Van Dooren's low-rank iteration method and optimize it over the symmetric positive semidefinite fixed-rank manifold using known Riemannian optimization algorithms. We consider the symmetric positive semidefinite fixed-rank manifold with its quotient manifold representation $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ and derive the Riemannian gradient and action of the Hessian with respect to a special Riemannian metric that improves the storage and computational complexity of some of the Riemannian objects (see Section 4.1.2).

Any symmetric positive semidefinite matrix S of low rank can be written as the factorization $S = XX^T$ where $X \in \mathbb{R}^{n\times r}$ and $\text{rank}(X) = r$. Let $\mathbb{R}_*^{n\times r}$ denote the noncompact Stiefel manifold, i.e., the set of full-rank matrices in $\mathbb{R}^{n\times r}$, and let $\mathcal{O}_r$ denote the orthogonal group, i.e., $\mathcal{O}_r = \left\{Q \in \mathbb{R}^{r\times r} | Q^TQ = I_r\right\}$. In $\mathbb{R}_*^{n\times r}$, we can define the equivalence relation $\sim$ where $X \sim Y$ if and

only if there exists a $Q \in \mathcal{O}_r$ such that $Y = XQ$ and the equivalence class of $\mathbb{R}_*^{n\times r}$ is defined as

$$[X] = \{XQ | Q \in \mathcal{O}_r\}. \tag{4.8}$$

Let $\mathbb{R}_*^{n\times r}/\mathcal{O}_r = \mathbb{R}_*^{n\times r}/\sim$ denote the quotient of $\mathbb{R}_*^{n\times r}$ by this equivalence class relation. Also, let $\pi : \mathbb{R}_*^{n\times r} \to \mathbb{R}_*^{n\times r}/\mathcal{O}_r$ denote the quotient map that sends $X \in \mathbb{R}_*^{n\times r}$ to its equivalence class $[X]$ viewed as an element of $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$. For notation, $\pi(X)$ denotes $[X]$ viewed as an element in $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ and $\pi^{-1}(\pi(X))$ denotes $[X]$ viewed as a subset of $\mathbb{R}_*^{n\times r}$. Absil et al. showed in [AILH09] that

$$\mathcal{S}_+(n, r) \simeq \mathbb{R}_*^{n\times r}/\mathcal{O}_r \tag{4.9}$$

is a quotient manifold.

Journée et al. in [JBAS10] equipped the quotient manifold (4.9) with the Euclidean metric. The Riemannian objects of $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ with the Euclidean metric are derived in [JBAS10]. While the Euclidean metric is a natural choice of a Riemannian metric on $\mathbb{R}_*^{n\times r}$, it may not be the ideal metric because, given the horizontal space (which is the orthogonal complement of the vertical space in the tangent space with respect to the Euclidean metric and derived in [JBAS10])

$$\mathcal{H}_{1,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r = \left\{X(X^TX)^{-1}S + X_\perp K \mid S = S^T,\ S \in \mathbb{R}^{r\times r},\ K \in \mathbb{R}^{(n-r)\times r}\right\}, \tag{4.10}$$

a natural basis of (4.10) is

$$\begin{aligned}\left\{X(X^TX)^{-1}e_ie_i^T,\ i = 1, \ldots, r\right\} \bigcup \left\{\frac{1}{\sqrt{2}}X(X^TX)^{-1}(e_ie_j^T + e_je_i^T),\ i = 1, \ldots, r,\ j = i+1, \ldots, r\right\} \\ \bigcup \left\{X_\perp \tilde{e}_ie_i^T,\ i = 1, \ldots, n-r,\ j = 1, \ldots, r\right\},\end{aligned} \tag{4.11}$$

where $(e_1, \ldots, e_r)$ is the canonical basis of $\mathbb{R}^r$ and $(\tilde{e}_1, \ldots, \tilde{e}_{(n-r)})$ is the canonical basis of $\mathbb{R}^{n-r}$. The basis (4.11) is not an orthonormal basis with respect to the metric. An orthonormal basis is important if the size of the problem is large since it allows us to simplify the intrinsic representation of some of the Riemannian objects (e.g., vector transport, metric, etc.) and to efficiently store the tangent vectors. Thus, we define an alternative Riemannian metric for $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ that provides a natural basis of a new horizontal space that is orthonormal with respect to the new metric.

The vertical space $\mathcal{V}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is defined as

$$\mathcal{V}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r = \left\{X\Omega \mid \Omega^T = -\Omega,\ \Omega \in \mathbb{R}^{r\times r}\right\}. \tag{4.12}$$

An alternative Riemannian metric $\bar{g}$ on $\mathbb{R}_*^{n\times r}$ is

$$\bar{g}_X(\eta_x, \xi_x) = \text{trace}((X^TX)\eta_x^T\xi_x) \tag{4.13}$$

for all $\eta_x, \xi_x \in \mathbb{R}_*^{n\times r}$. Then, the horizontal space is the orthogonal complement to the vector space (4.12) with respect to the metric (4.13) is

$$\mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r = \left\{Z \in \mathbb{R}_*^{n\times r} | (X^TX)^{-1}X^TZ = Z^TX(X^TX)^{-1}\right\} \tag{4.14}$$

$$= \left\{XS + X_\perp K \mid S = S^T,\ S \in \mathbb{R}^{r\times r}, K \in \mathbb{R}^{(n-r)\times r}\right\}. \tag{4.15}$$

The horizontal space is important because it allows us to represent tangent spaces to the quotient manifold. To do this, we define the horizontal lift [AMS08]. Given $X \in \mathbb{R}_*^{n\times r}$ and a tangent vector $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, there exists a unique vector $\eta_{\uparrow_X} \in \mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, called the horizontal lift of $\eta_{\pi(X)}$ at X, that satisfies

$$D\pi(X)[\eta_{\uparrow_X}] = \eta_{\pi(X)}. \tag{4.16}$$

Also, note that a vector field $X \in \mathbb{R}_*^{n\times r} \mapsto \eta_{\uparrow_X} \in \mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is a horizontal lift of a tangent vector to $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ if and only if it satisfies

$$\eta_{XQ} = \eta_X Q, \quad \forall Q \in \mathcal{O}_r. \tag{4.17}$$

The horizontal lift allows us to define a Riemannian metric g on the quotient manifold $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$.

Given a horizontal lift satisfying (4.16), the metric (4.13) on $\mathbb{R}_*^{n\times r}$ defines a metric g on $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$.

Lemma 4.1.1 *The mapping g defined by*

$$g_{2,\pi(X)}(\eta_{\pi(X)}, \xi_{\pi(X)}) = \bar{g}_{2,X}(\eta_{\uparrow X}, \xi_{\uparrow X}) = \text{trace}((X^TX)\eta_{\uparrow X}^T\xi_{\uparrow X}) \tag{4.18}$$

is a Riemannian metric on $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$.

Proof The lifted metric is invariant on the chosen representation for $\pi(X)$

$$\bar{g}_{2,XQ}(\eta_{\uparrow XQ}, \xi_{\uparrow XQ}) = \text{trace}\left((XQ)^T(XQ)(\eta_{\uparrow XQ}^T\xi_{\uparrow XQ})\right) = \text{trace}\left((XQ)^T(XQ)((\eta_{\uparrow X}Q)^T\xi_{\uparrow X}Q)\right)$$
$$= \text{trace}\left((X^TX)\eta_{\uparrow X}^T\xi_{\uparrow X}\right) = \bar{g}_X(\eta_{\uparrow X}, \xi_{\uparrow X}). \quad \blacksquare$$

An orthonormal basis of (4.15) with respect to (4.18) is

$$\left\{XL^{-T}e_ie_i^TL^{-1},\ i=1,\dots,r\right\}\bigcup\left\{\frac{1}{\sqrt{2}}XL^{-T}(e_ie_j^T+e_je_i^T)L^{-1},\ i=1,\dots,r,\ j=i+1,\dots,r\right\}$$
$$\bigcup\left\{X_\perp\tilde{e}_ie_i^TL^{-1},\ i=1,\dots,n-r,\ j=1,\dots,r\right\}, \tag{4.19}$$

where $(e_1,\dots,e_r)$ is the canonical basis of $\mathbb{R}^r$, $(\tilde{e}_1,\dots,\tilde{e}_{(n-r)})$ is the canonical basis of $\mathbb{R}^{n-r}$, and $X^TX=LL^T$ is the Cholesky decomposition.

The orthogonal projections to the horizontal and vertical space (defined by (4.15) and (4.12), respectively) are easily characterized.

Lemma 4.1.2 *Let X be a point on $\mathbb{R}_*^{n\times r}$. For any $\eta_x\in\mathbb{R}^{n\times r}$, the orthogonal projection onto the horizontal space $\mathcal{H}_{2,X}$ (4.15) is*

$$\mathrm{P}_X^h(\eta_x)=\eta_x-X\,\mathrm{skew}\left((X^TX)^{-1}X^T\eta_x\right) \tag{4.20}$$

and the projection onto the vertical space $\mathcal{V}_X$ (4.12) is

$$\mathrm{P}_X^v(\eta_x)=X\,\mathrm{skew}\left((X^TX)^{-1}X^T\eta_x\right) \tag{4.21}$$

where $\mathrm{skew}(A)=\frac{1}{2}(A-A^T)$ *for any matrix $A\in\mathbb{R}^{r\times r}$.*

Proof Any vector $\eta\in\mathbb{R}^{n\times r}$ has the unique decomposition

$$\eta=\eta_{\mathcal{V}_X\mathcal{M}}+\eta_{\mathcal{H}_X\mathcal{M}}$$

where $\mathcal{M}=\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ and each element $\eta_{\mathcal{X}}$ belongs to the vector space $\mathcal{X}$. The orthogonal projection $\mathrm{P}_X^h(\cdot)$ extracts the component that lies in the horizontal space, i.e.,

$$\mathrm{P}_X^h(\eta)=\eta-X\Omega$$

where Ω is a skew-symmetric matrix and $\mathrm{P}_X^h(\eta)$ satisfies

$$(X^TX)^{-1}X^T\mathrm{P}_X^h(\eta)=(\mathrm{P}_X^h(\eta))^TX(X^TX)^{-1}.$$

Therefore, Ω is given by

$$\Omega=\frac{1}{2}\left((X^TX)^{-1}X^T\eta-\eta^TX(X^TX)^{-1}\right)=\mathrm{skew}\left((X^TX)^{-1}X^T\eta\right).\quad\blacksquare$$

4.1.3 Retractions and Vector Transports on Quotient Manifold

A natural choice of retraction for $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is

$$\mathrm{R}_{\pi(X)}(\eta_{\pi(X)}) = \pi(X + \eta_{\uparrow X}) \tag{4.22}$$

which maps a point $\eta_{\pi(X)} \in \mathrm{T}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ to $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$. This mapping can be represented by the mapping $\mathrm{R}_X : \mathcal{H}_X\mathbb{R}_*^{n\times r} \to \mathbb{R}_*^{n\times r}$, where the retraction is defined as

$$\mathrm{R}_X(\eta_x) = X + \eta_x. \tag{4.23}$$

That is, the retraction can be used to map representative elements of the total space and horizontal space to a representative element of the equivalence class. Therefore, it is crucial to quotient spaces to choose representative elements of the equivalence classes carefully for computational and analytical reasons.

Note that (4.23) may fail if X is not full rank. Thus, we check to see if X is full rank by computing the QR factorization of X and checking to see if the diagonal of R is nonzero. If the first p elements on the diagonal of R are nonzero and the $p+1$ diagonal element is zero, i.e., $R_{p,p} \neq 0$ and $R_{p+1,p+1} = 0$, then we truncate X to the be equal to the first p columns of X. This reduces the rank of X and moves to a manifold with a smaller rank. In Chapter 7, we explore other rank-adaptive strategies for the role extraction problem.

A common vector transport used in Riemannian optimization is differentiated retraction. The vector transport by differentiated retraction (which is also the vector transport by projection) is

$$\Big(\mathcal{T}_{\eta_{\pi(X)}}(\xi_{\pi(X)})\Big)_{\uparrow_{(X+\eta_{\uparrow X})}} = \mathrm{P}^h_{(X+\eta_{\uparrow X})}(\xi_{\uparrow X}). \tag{4.24}$$

However, this vector transport is not isometric and may be expensive to compute. Since we have an orthonormal basis of $\mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ with respect to the Riemannian metric (4.18). Vector transport by parallelization is the identity. In other words, for the d-dimensional tangent vector $v_{\uparrow X} = B_X^\flat \eta_{\uparrow X}$, the vector transport by parallelization is given by

$$\mathcal{T}^d v_{\uparrow X} = B_Y^\flat \mathcal{T} \eta_{\uparrow X} = B_Y^\flat B_Y B_X^\flat \eta_{\uparrow X} = (B_Y^\flat B_Y)(B_X^\flat B_X) v_{\uparrow X} = v_{\uparrow X} \tag{4.25}$$

where B_Y and B_X are orthonormal bases of $\mathcal{H}_{2,Y}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ and $\mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, respectively, and $\mathcal{T}^d$ is a d-dimensional representation of the vector transport [Hua13, Section 9.5].

Since B_X forms an orthonormal basis of $\mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, the Riemannian metric (4.18) reduces to the Euclidean metric for the intrinsic representations, i.e.,

$$g_X(\eta_{\uparrow X}, \xi_{\uparrow X}) = v_{\uparrow X}^T u_{\uparrow X}, \tag{4.26}$$

where $\eta_{\uparrow X} = B_X v_{\uparrow X}$ and $\xi_{\uparrow X} = B_{\uparrow X} u_{\uparrow X} \in \mathcal{H}_{2,X}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ [Hua13, Section 9.5].

4.1.4 Riemannian Gradient and the Action of the Hessian on Quotient Manifold

Recall the definition of Riemannian gradient [AMS08, (3.31)]: the gradient of F at a point $\pi(X)$ on $\mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ satisfies

$$DF(\pi(X))[\eta_{\pi(X)}] = g_{2,\pi(X)}(\operatorname{grad} F(\pi(X)), \eta_{\pi(X)}) \tag{4.27}$$

for all $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$. With this definition, we are able to derive the horizontal lift of the gradient of (4.6) at $\pi(X)$ with respect to the metric (4.18), which is given in Lemma 4.1.3.

Lemma 4.1.3 *If the horizontal space is* (4.15) *with respect to the Riemannian metric* (4.18)*, then the horizontal lift of the gradient of* (4.5) *at* $\pi(X)$ *is*

$$(\operatorname{grad} F_r(\pi(X)))_{\uparrow_X} = 2 \operatorname{grad} f(XX^T)X(X^TX)^{-1}. \tag{4.28}$$

Proof Let $\hat{F}$ denote the cost function on $\mathbb{R}_*^{n\times r}$. Then, the directional derivative of $\hat{F}$ along any $Z \in \mathbb{R}_*^{n\times r}$ is

$$\begin{aligned}
D\hat{F}(X)[Z] = \operatorname{trace}\Big(& ZX^TX_1X_1^T + XZ^TX_1X_1^T - \frac{1}{2}ZX^TXX^T - \frac{1}{2}XZ^TXX^T - \frac{1}{2}XX^TZX^T \\
& - \frac{1}{2}XX^TXZ^T + \beta^2 ZX^TAXX^TA^T + \beta^2 XZ^TAXX^TA^T + \beta^2 XX^TAZX^TA^T \\
& + \beta^2 XX^TAXZ^TA^T\Big) \\
= \operatorname{trace}\Big(& \left(ZX^T + XZ^T\right)^T \left(X_1X_1^T - XX^T + \beta^2\Gamma\left[XX^T\right]\right)\Big) \\
= \operatorname{trace}\Big(& \left(ZX^T + XZ^T\right)^T \operatorname{grad} f(XX^T)\Big) \\
= \operatorname{trace}\left(Z^T \operatorname{grad} f(XX^T)X\right) & + \operatorname{trace}\Big(Z^T \left(\operatorname{grad} f(XX^T)\right)^T X\Big).
\end{aligned}$$

Since $\operatorname{grad} f(XX^T) = \left(\operatorname{grad} f(XX^T)\right)^T$, then

$$D\hat{F}(X)[Z] = \operatorname{trace}\left(Z^T 2 \operatorname{grad} f(XX^T)X\right) \tag{4.29}$$

and the gradient in $\mathbb{R}_*^{n\times r}$ is

$$\operatorname{grad}\hat{F}(X) = 2\operatorname{grad} f(XX^T)X. \tag{4.30}$$

Now, observe that the directional derivative along any $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is

$$\begin{aligned} DF(\pi(X))[\eta_{\pi(X)}] &= DF(\pi(X))[D\pi(X)[\eta_{\uparrow X}]] = D\hat{F}(X)[\eta_{\pi(X)}] \\ &= \bar{g}_X(\operatorname{grad}\hat{F}(X), \eta_{\uparrow X}) = \bar{g}_X(\mathrm{P}_X^h(\operatorname{grad}\hat{F}(X)), \eta_{\uparrow X}). \end{aligned}$$

Using the definition of the Riemannian gradient given by (4.27), the directional derivative of the gradient in $\mathbb{R}_*^{n\times r}$ (4.29), the Riemannian metric (4.18), and the fact that

$$g_{2,\pi(X)}(\eta_{\pi(X)}, \xi_{\pi(X)}) = \bar{g}_X(\eta_{\uparrow X}, \xi_{\uparrow X}),$$

then F along any $\eta_X \in \mathbb{R}_*^{n\times r}$ is

$$\begin{aligned} D\hat{F}(X)[\eta_X] &= \operatorname{trace}(\eta_X^T 2\operatorname{grad} f(XX^T)X) \\ &= \operatorname{trace}(\eta_X^T 2\operatorname{grad} f(XX^T)X(X^TX)^{-1}(X^TX)) \\ &= \operatorname{trace}((X^TX)\eta_X^T 2\operatorname{grad} f(XX^T)X(X^TX)^{-1}). \end{aligned}$$

Therefore, the horizontal lift of the gradient is

$$(\operatorname{grad} F_r(\pi(X)))_{\uparrow_X} = \mathrm{P}_X^h\left(2\operatorname{grad} f(XX^T)X(X^TX)^{-1}\right) = 2\operatorname{grad} f(XX^T)X(X^TX)^{-1}. \quad \blacksquare$$

Observe that (4.28) satisfies (4.16). Also, (4.28) is in the horizontal space (4.15) with respect to the metric (4.18) and that it is orthogonal to all vectors in the vertical space (4.12) with respect to the metric (4.18).

For a Riemannian quotient manifold $\mathcal{M} = \bar{\mathcal{M}}/\sim$, if ∇ and $\bar{\nabla}$ denote the Riemannian connections on $\mathcal{M}$ and $\bar{\mathcal{M}}$, respectively, then the horizontal lift of the Riemannian Hessian of a real-valued function f at a point $\pi(x) \in \mathcal{M}$ along $\eta_{\pi(x)} \in \mathcal{H}_x\mathcal{M}$ is defined as

$$\left(\operatorname{Hess} f(\pi(x))[\eta_{\pi(x)}]\right)_{\uparrow x} = \left(\nabla_{\eta_{\pi(x)}} \operatorname{grad} f\right)_{\uparrow x} = \mathrm{P}_x^h\left(\bar{\nabla}_{\eta_{\uparrow x}} (\operatorname{grad} f)_{\uparrow x}\right) \tag{4.31}$$

where $\mathrm{P}_x^h(\cdot)$ is the projection onto the horizontal space and $\eta_{\uparrow x} \in \mathcal{H}_x\mathcal{M}$ is the horizontal lift (see Proposition 5.3.3 in [AMS08] and Definition 1.3.5). Note that we do not have the Riemannian connection for our metric. However, Absil et al. in [AMS08, Section 6.4.3] show that for Newton's

method the affine connection ∇ does not need to be the Riemannian connection induced by the metric $\bar{g}$. Since the geodesics of $\mathbb{R}_*^{n\times r}$ with respect to the metric (4.13) are unknown, we use the quadratic local model, i.e.,

$$m_x(\eta) = f(x) + \langle \operatorname{grad} f(x), \eta \rangle_x + \frac{1}{2} \langle \operatorname{Hess} f(x)[\eta], \eta \rangle_x , \tag{4.32}$$

for all $\eta \in \mathrm{T}_x\mathcal{M}$, and the retraction (4.23) to determine an approximate Hessian along the horizontal lift $\eta_{\uparrow X} \in \mathcal{H}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$. Then, we project this approximate Hessian onto the horizontal space using the projection operator (4.20) to determine the horizontal lift of the approximate Hessian of (4.5) at $\pi(X)$ along $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ (see Lemma 4.1.4).

Lemma 4.1.4 *The horizontal lift of the approximate Hessian of* (4.5) *at* $\pi(X)$ *along* $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ *is*

$$\begin{aligned}\left(\operatorname{Hess} F_r(\pi(X))[\eta_{\pi(X)}]\right)_{\uparrow X} = \mathrm{P}_X^h \big(2 \left(\operatorname{Hess} f(XX^T)[X\eta_{\uparrow X}^T + \eta_{\uparrow X}X^T]X \right.\\ \left. + \operatorname{grad} f(XX^T)\eta_{\uparrow X}\right)(X^TX)^{-1}\big),\end{aligned} \tag{4.33}$$

where the projection operator to the horizontal space $\mathrm{P}_X^h(\cdot)$ *is given by* (4.20).

Proof A function on the manifold can be pulled back to the tangent space through the retraction (1.7), i.e., let $\hat{F}(X) = F(R_X(\eta_{\uparrow X}))$ for $X \in \mathbb{R}_*^{n\times r}$ and $\eta_{\uparrow X} \in \mathcal{H}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ where the cost function F is given by (4.5) and the retraction R is given by (4.23). Then, we expand $\hat{F}$ such that we get something in the form of the second order local model (4.32) with respect to the metric (4.18), i.e., (for simplfication, $\eta = \eta_{\uparrow X}$)

$$\begin{aligned}
F(R_X(\eta)) =& \operatorname{trace}\left((X+\eta)(X+\eta)^T(X_1X_1^T - \frac{1}{2}(X+\eta)(X+\eta)^T + \beta^2 A(X+\eta)(X+\eta)^T A^T)\right)\\
=& \operatorname{trace}\left(XX^T(X_1X_1^T - \frac{1}{2}XX^T + \beta^2 AXX^TA^T)\right) + \operatorname{trace}\left(XX^T(-\frac{1}{2}(X\eta^T + \eta X^T)\right.\\
&\left. + \beta^2 A(X\eta^T + \eta X^T)A^T) + (X\eta^T + \eta X^T)(X_1X_1^T - \frac{1}{2}XX^T + \beta^2 AXX^TA^T)\right)\\
&+ \operatorname{trace}\left(XX^T(-\frac{1}{2}\eta\eta^T + \beta^2 A\eta\eta^TA^T) + (X\eta^T + \eta X^T)(-\frac{1}{2}(X\eta^T + \eta X^T)\right.\\
&\left. + \beta^2 A(X\eta^T + \eta X^T)A^T) + \eta\eta^T(X_1X_1^T - \frac{1}{2}XX^T + \beta^2 AXX^TA^T)\right) + O(\|\eta^3\|)\\
=& F(X) + \operatorname{trace}\left((X\eta^T + \eta X^T)(X_1X_1^T - XX^T + \beta^2(AXX^TA^T + A^TXX^TA))\right)\\
&+ \operatorname{trace}\left(\eta^T(-(X\eta^T + \eta X^T) + \beta^2(A(X\eta^T + \eta X^T)A^T + A^T(X\eta^T + \eta X^T)A))X\right.\\
&\left. + \eta^T(X_1X_1^T - XX^T + \beta^2(AXX^TA^T + A^TXX^TA))\eta\right) + O(\|\eta^3\|).
\end{aligned} \tag{4.34}$$

Observe that the second trace operator in the truncated expression (4.34) simplifies to (4.29). Since the local model is evaluated with respect to a given Riemannian metric, the Riemannian gradient of (4.5) is given by

$$\begin{aligned}\langle \eta, \operatorname{grad} f(X)\rangle_X &= \operatorname{trace}\left(\eta^T 2 \operatorname{grad} f(XX^T)X\right) = \operatorname{trace}\left((X^TX)\eta^T 2 \operatorname{grad} f(XX^T)X(X^TX)^{-1}\right)\\ &= \left\langle \eta, 2 \operatorname{grad} f(XX^T)X(X^TX)^{-1}\right\rangle_X . \end{aligned} \tag{4.35}$$

By Lemma 4.1.3, we know that $2 \operatorname{grad} f(XX^T)X(X^TX)^{-1}$ is the horizontal lift of the gradient of (4.5) at $\pi(X)$; thus, it is already in the horizontal space.

Next, we have that the third trace operator in (4.34) contributes to the expression of the Hessian with respect to the metric (4.13), i.e.,

$$\begin{aligned}\langle \eta, \operatorname{Hess} F(X)[\eta]\rangle_X &= 2 \operatorname{trace}\big(\eta^T((-(X\eta^T + \eta X^T) + \beta^2\Gamma[X\eta^T + \eta X^T])X\\ &\quad + (X_1X_1^T - XX^T + \beta^2\Gamma[XX^T])\eta)\big)\\ &= \operatorname{trace}\left(\eta^T 2(\operatorname{Hess} f(XX^T)[X\eta^T + \eta X^T]X + \operatorname{grad} f(XX^T)\eta)\right)\\ &= \operatorname{trace}\left((X^TX)\eta^T 2(\operatorname{Hess} f(XX^T)[X\eta^T + \eta X^T]X + \operatorname{grad} f(XX^T)\eta)(X^TX)^{-1}\right)\\ &= \left\langle \eta, 2(\operatorname{Hess} f(XX^T)[X\eta^T + \eta X^T]X + \operatorname{grad} f(XX^T)\eta)(X^TX)^{-1}\right\rangle_X . \end{aligned} \tag{4.36}$$

To enforce that the approximate Hessian is a linear and symmetric mapping of $\mathcal{H}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ to $\mathcal{H}_X\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, we will apply the orthogonal projector onto $\mathcal{H}_X\mathbb{R}_*^{n\times rr}/\mathcal{O}_r$ (4.20) to (4.36). Thus, the horizontal lift of the action of the Hessian of (4.5) at $\pi(X)$ along $\eta_{\pi(X)} \in \mathrm{T}_{\pi(X)}\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ is

$$\begin{aligned}\left(\operatorname{Hess} F_r(\pi(X))[\eta_{\pi(X)}]\right)_{\uparrow X} =& \mathrm{P}_X^h\left(2\left(\operatorname{Hess} f(XX^T)[X\eta_{\uparrow X}^T + \eta_{\uparrow X}X^T]X\right.\right.\\ &\left.\left. + \operatorname{grad} f(XX^T)\eta_{\uparrow X}\right)(X^TX)^{-1}\right). \quad \blacksquare\end{aligned}$$

4.2 Phase II: Extract Role Partition with K-means Clustering and Silhouette Statistic

In order to avoid computing the full approximate similarity matrix $S^{(r)}$ for the second phase, we propose using k-means clustering [AV07, HTF09] on the scaled low-rank factor $\overline{X} \in \mathbb{R}_*^{n\times r}$ of the similarity matrix, where X is scaled by its unscaled self-similarity scores. This allows us to exploit the low-rank structure we found for the similarity matrix in the first phase, maintaining

efficiency in time and complexity. Scaling the low-rank factor ensures that nodes are more similar to themselves than other nodes, which improves the quality of the clustering algorithm.

To determine the optimal number of roles, we propose to use the silhouette statistic [Rou87, KR05] to evaluate the clustering assignment at an assumed number of clusters. We test the silhouette statistic for an assumed list of roles $\{2, \ldots, r+1\}$, where r is the rank of the approximate similarity matrix. We choose to test the silhouette statistic to $r+1$ because the rank of the similarity matrix can be less than the number of roles in the network (see Theorem 3.0.4).

While the gap statistic is more robust than the silhouette statistic, the gap statistic is more computationally expensive [TWH01]. We show, empirically, in Section 6.1.4, that the silhouette statistic is able to determine the correct number of roles for simulated random networks.

The main issue of using the silhouette statistic to extract the role structure is that if r is large, then the computational complexity in time and space increases because we have to test for several numbers of roles. We discuss in Chapter 7 a way to use Riemannian rank-adaptive techniques to fix this issue.

CHAPTER 5

BROWET'S FULL-RANK SIMILARITY ALGORITHM VIEWED AS EUCLIDEAN OPTIMIZATION

In this chapter we describe Browet and Van Dooren's algorithms for the neighborhood pattern based similarity measure as Euclidean optimization with respect to the cost function (4.1).

5.1 The Euclidean Gradient Projection Algorithm

The gradient projection method is a popular Euclidean method to solve the constrained optimization problem

$$\min f(x), \quad \text{subject to } x \in K \subseteq \mathbb{R}^n \tag{5.1}$$

where K is a nonempty and convex set, $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function over K, and $\mathbb{R}^n$ denotes an n-dimensional vector space with an unspecified metric (5.1).

The gradient projection method given in [Ber99] generates a sequence of iterates $\{x_k\}$ given by

$$x_{k+1} = x_x + \alpha_k(\bar{x}_x - x_k), \tag{5.2}$$

where

$$\bar{x}_k = [x_k - t_k \operatorname{grad} f(x_k)]^+, \tag{5.3}$$

$\alpha_k \in (0, 1]$, $t_k > 0$ is a scalar, and $[\cdot]^+$ denotes the projection onto the set K, which is defined as

$$[x]^+ = \arg\min_{z \in K} \|z - x\|_2^2. \tag{5.4}$$

The convexity of K guarantees that the projection $[x]^+$ is unique for all $x \in \mathbb{R}^n$. However, in practice, $\arg\min_{z \in K} \|z - x\|_2^2$ may be expensive to compute, unless K has a structure that can be exploited to reduce the computational cost.

There are several stepsize selection procedures for (5.3) where either α_k or t_k can be viewed as the stepsize. If t_k is some fixed number and α_k is the stepsize, then the gradient projection method

is a feasible direction method with feasible direction $d_k = \bar{x}_k - x_k$. If $\alpha_k = 1$ for all k and t_k is taken to be the stepsize, then the update for the gradient projection method is

$$x_{k+1} = [x_k - t_k \operatorname{grad} f(x_k)]^+ . \tag{5.5}$$

This means that x_{k+1} is determined by an Armijo-like search on the projected arc $\{x_k(s) \mid t > 0\}$. The convergence analysis of the gradient projection method is given in [Ber99, Section 2.3.2].

Proposition 5.1.1 *(Armijo Rule along the Projection Arc) [Ber99, Proposition 2.3.3]*

(a) For every $x \in K$ there exists scalars $\bar{t} > 0$, $\theta \in (0,1)$, and $\sigma \in (0,1)$, and we set $t_k = \theta^{m_k}\bar{t}$, where m_k is the first nonnegative integer m for which

$$(f(x_k) - f(\bar{x}_k)) \geq -\sigma\theta^m \operatorname{grad} f(x_k)^T(\bar{x}_k - x_k), \tag{5.6}$$

where $\bar{x}_k = [x_k - \theta^m \bar{t} \operatorname{grad} f(x_k)]^+$ and $\bar{t} > 0$ is the minimal initial stepsize

(b) Let $\{x_k\}$ be a sequence generated by the gradient projection method with $\alpha_k \equiv 1$ for all k and with stepsize t_k chosen by the Armijo rule along the projection arc. Then every point of $\{x_k\}$ is stationary.

Suppose that $t_k = t$ for some fixed constant $t > 0$ and $\alpha_k = 1$ for all k. Then, it is possible to show that the limit point of a sequence generated by the gradient projection method is a critical point (or stationary point) if t is sufficiently small and the gradient satisfies the Lipschitz continuity condition [Ber99].

Proposition 5.1.2 *(Constant Stepsize) [Ber99, Proposition 2.3.2] Let $\{x_k\}$ be a sequence generated by the projection method with $\alpha_k = 1$ and $t_k = t$ for all k. Assume that for some constant $L > 0$, that*

$$\|\operatorname{grad} f(x) - \operatorname{grad} f(y)\| \leq L\|x - y\|, \quad \forall x, y \in K. \tag{5.7}$$

Then, if $0 < t < 2/L$, every limit point of $\{x_k\}$ is stationary.

Lastly, we determine a bound for all stepsizes t that satisfy the Armijo condition (5.6). To prove this bound, we use two theorems. The first is the descent lemma given in [Ber99, Proposition A.24].

Lemma 5.1.3 *(Descent Lemma) [Ber99, Proposition A.24] Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and let x and y be two vectors in $\mathbb{R}^n$. Suppose that*

$$\|\operatorname{grad} f(x + sy) - \operatorname{grad} f(x)\| \leq Ls\|y\|, \quad \forall s \in [0, 1]$$

where L is some scalar. Then

$$f(x + y) \leq f(y) + y^T \operatorname{grad} f(x) + \frac{L}{2}\|y\|^2. \tag{5.8}$$

The second theorem we use is a result of the projection theorem [Ber99, Proposition 2.1.3]. That is, for $x \in K$, if $\bar{x} = [x - t \operatorname{grad} f(x)]^+$ for $t > 0$, then

$$(y - \bar{x})^T (x - \bar{x}) \leq t(y - \bar{x})^T \operatorname{grad} f(x)$$

and

$$\|x - \bar{x}\| \leq t(x - \bar{x})^T \operatorname{grad} f(x). \tag{5.9}$$

With these theorems, we can prove the following lemma.

Lemma 5.1.4 *If $\operatorname{grad} f(x)$ is Lipschitz continuous with constant $L > 0$, then the Armijo rule along the projection arc (5.6) is satisfied for all t such that*

$$t \leq \frac{2(1 - \sigma)}{L}. \tag{5.10}$$

There is a lower bound on the stepsize t

$$t_- = \frac{2\theta(1 - \sigma)}{L} \leq \theta^m \bar{t} = t, \tag{5.11}$$

where $\bar{t} > 0$ is the minimal initial stepsize and $\theta \in (0, 1)$ is the reduction factor used in the Armijo line search.

Proof By the descent lemma 5.1.3, we have

$$f(x_{k+1}) - f(x_k) = f(\bar{x}_k) - f(x_k) \leq (\operatorname{grad} f(x_k))^T (\bar{x}_k - x_k) + \frac{L}{2}\|\bar{x}_k - x_k\|^2. \tag{5.12}$$

Subtracting $\sigma(\operatorname{grad} f(x_k))^T (\bar{x}_k - x_k)$ from both sides of the inequality, we get

$$\begin{aligned} f(\bar{x}_k) - f(x_k) - \sigma(\operatorname{grad} f(x_k))^T (\bar{x}_k - x_k) &\leq (1 - \sigma)(\operatorname{grad} f(x_k))^T (\bar{x}_k - x_k) + \frac{L}{2}\|\bar{x}_k - x_k\|^2 \\ &\leq -\frac{(1 - \sigma)}{t}\|\bar{x}_k - x_k\|^2 + \frac{L}{2}\|\bar{x}_k - x_k\|^2 \\ &\leq \left(\frac{L}{2} - \frac{(1 - \sigma)}{t}\right)\|\bar{x}_k - x_k\|^2. \end{aligned} \tag{5.13}$$

If $t \leq (2(1-\sigma))/L$ for all t, then the right hand side of (5.13) is nonpositive and the Armijo condition along the projection arc (5.6) is satisfied for $\sigma \in (0,1)$.

The search of stepsize t will terminate either when t is large, or when

$$\theta^m \bar{t} \leq \frac{2(1-\sigma)}{L} \leq \theta^{m-1} \bar{t}.$$

Therefore, we have a lower bound on stepsize t by applying an additional θ, i.e.,

$$t_- = \frac{2\theta(1-\sigma)}{L} \leq \theta^m \bar{t} = t. \quad \blacksquare$$

5.2 Browet's Full-Rank Similarity Algorithm viewed as a Gradient Projection Method

In this section, we reformulate Browet and Van Dooren's full-rank similarity algorithm as a Euclidean optimization problem for our cost function (4.1). Then, we show that their full-rank iterative algorithm is a Euclidean gradient projection method along a projection arc with respect to the cost function (4.1) for constant stepsize $t_k = 1$ for all k (Section 5.2.1). Lastly, in Section 5.2.2, we add an Armijo stepsize to the gradient projection method and use Lemma 5.1.4 to show which stepsizes always satisfy Proposition 5.1.1 with respect to the neighborhood pattern similarity parameter β.

5.2.1 Euclidean Gradient Projection Method with Constant Stepsize

First, we show that Browet and Van Dooren's full-rank similarity algorithm (2.22) is a gradient projection method along a projection arc for the cost function (4.1) with constant stepsize $t_k = 1$ for all k, and by Proposition 5.1.2, it converges to the stationary point.

Let $\mathcal{S}_+^n$ denote the set of $n \times n$ symmetric positive semidefinite matrices. Consider the gradient projection method along a projection arc (5.5) for an ascent direction. If the stepsize is a fixed constant $t_k = 1$ for all k, then the Euclidean updated is give by

$$S_{k+1} = S_k + \operatorname{grad} f(S_k) = S_k + S_1 - S_k + \beta^2(AS_kA^T + A^TS_kA) = S_1 + \beta^2 \Gamma_A[S_k], \tag{5.14}$$

which is equivalent to the full-rank iterative update (2.22) in Browet and Van Dooren's algorithm.

Next, consider the Euclidean projection operator. The Euclidean projection onto the convex set $\mathcal{S}_+^n$ with the Frobenius norm $\|\cdot\|_F$ is given by

$$[S]^+ = \sum_{i=1}^{n} \max\{0, \lambda_i\} u_i u_i^T, \tag{5.15}$$

where $S = \sum_{i=1}^{n} \lambda_i u_i u_i^T$ is the eigenvalue decomposition of S [BV04].

Lastly, we must show that for fixed stepsize $t_k = 1$, the gradient projection method for the cost function satisfies the convergence Proposition 5.1.2 [Ber99]. To do this, we must show that the gradient is Lipschitz continuous. Take $S, R \in \mathcal{S}_n^+$ such that $\operatorname{grad} f(S) = S_1 - S + \beta^2(ASA^T + A^TSA)$ and $\operatorname{grad} f(R) = R_1 - R + \beta^2(ARA^T + A^TRA)$, where $S_1 = AA^T + A^TA = R_1$ by equation (2.23). Then, using properties of the Frobenius norm [HJ90, Section 5.6], we have

$$\begin{aligned}
\|\operatorname{grad} f(S) - \operatorname{grad} f(R)\|_F &= \|\big(S_1 - S + \beta^2(ASA^T + A^TSA)\big) - \big(R_1 - R + \beta^2(ARA^T + A^TRA)\big)\|_F \\
&= \|(R - S) - \beta^2(A(R - S)A^T + A^T(R - S)A)\|_F \\
&= \|\big(I - \beta^2\left(A \otimes A + A^T \otimes A^T\right)\big)\, vec(R - S)\|_2 \\
&= \|\big(I - \beta^2\Gamma_A\big)\, vec(R - S)\|_2 \\
&\leq \left(1 - \beta^2\rho\left(A \otimes A + A^T \otimes A^T\right)\right)\|R - S\|_F \\
&\leq \left(1 - \beta^2\left(\|A\|^2 + \|A^T\|^2\right)\right)\|R - S\|_F \\
&= \left(1 - 2\beta^2\|A\|^2\right)\|R - S\|_F, \qquad (5.16)
\end{aligned}$$

where $\|\cdot\|$ is the spectral norm. For the inequality above to be true, $L = 1 - 2\beta^2\|A\|^2$ must be greater than zero. Hence, $L > 0$ when

$$\beta^2 < \frac{1}{2\|A\|^2}. \qquad (5.17)$$

Therefore, the Euclidean gradient of the cost function (4.1) is Lipschitz continuous for $L = 1 - 2\beta^2\|A\|^2 > 0$ if β satisfies the bound (5.17). Observe also that L is less than 1.

By Proposition 5.1.2, every limit point $\{S_k\}$ of the gradient projection method is stationary if $0 < t < 2/L$. For $L = 1 - 2\beta^2\|A\|^2$ where β satisfies (5.17), the inequality $0 < t < 2/L$ is true for $t = 1$. Therefore, for stepsize t in the interval $0 < t < 2/(1 - 2\beta^2\|A\|^2)$ where β is bounded above by (5.17), every limit point of a sequence of iterates $\{S_k\}$ for convex optimization problem solved by the gradient projection method along a projection arc with a fixed stepsize is a stationary point.

Observe that the bound (5.17) is looser than (2.26) if the adjacency matrix A is nonnegative (i.e., $A \geq 0$). To prove Theorem 5.2.1, we need the following properties [HJ90]:

(i) $\|AA^T\| = \|A^TA\| = \|A\|^2$ for all $A \in \mathbb{R}^{n\times n}$;

(ii) Let $A, B \in \mathbb{R}^{n\times n}$. If $|A| \leq B$, then $\rho(A) \leq \rho(|A|) \leq \rho(B)$.

Theorem 5.2.1 *Let A be a nonnegative matrix in $\mathbb{R}^{n\times n}$. Then*

$$\frac{1}{\rho(A+A^T)^2} \leq \frac{1}{2\|A\|^2} \leq \frac{1}{\rho(A\otimes A + A^T \otimes A^T)}. \tag{5.18}$$

Proof Observe that $2A^TA \leq (A+A^T)^2$ since A is nonnegative. Therefore,

$$\rho(A+A^T)^2 = \rho((A+A^T)^2) \geq \rho(2A^TA) = 2\|A^TA\| = 2\|A\|^2. \tag{5.19}$$

Also, recall that $\rho(A\otimes A + A^T\otimes A^T) \leq 2\|A\|^2$. Therefore, (5.18) follows. ∎

Therefore, when the adjacency matrix A is nonnegative, by Theorem 5.2.1, we have a range for the similarity parameter β with stepsize $t_k = 1$ for which the Euclidean gradient projection method along a projection arc converges to a stationary point.

5.2.2 Euclidean Gradient Projection Method with Armijo Stepsize

Next we include a stepsize and Armijo line-search method to compute the neighborhood pattern similarity matrix. That is, the Euclidean update with stepsize t_k is

$$S_{k+1} = [S_k + t_k \operatorname{grad} f(S_k)]^+ = \left[S_k + t_k\left(S_1 - S_k + \beta^2(AS_kA^T + A^TS_kA)\right)\right]^+, \tag{5.20}$$

where $[\cdot]^+$ is the Euclidean projection operator (5.15) onto the set $\mathcal{S}_+^n$ with the Frobenius norm. Therefore, if the stepsize t_k is chosen by the Armijo rule (5.6), then by Proposition 5.1.1, the Euclidean gradient projection method converges to a stationary point.

Note that by 5.1.4, we have, for all k, that the stepsize

$$t_k \leq \frac{2(1-\sigma)}{L} < \frac{2}{L}, \tag{5.21}$$

where $\sigma \in (0,1)$ and $L = 1-2\beta^2\|A\|^2$ is positive when β satisfies (5.17). In practice, σ is chosen to be 10^{-4} [Ber99]. Therefore, the choice of the Armijo stepsize is dependent upon the parameter β. If β is chosen close to (5.17), then the right hand side of (5.21) goes to infinity and large stepsizes satisfy the Armijo rule (5.6). If β is chosen close to zero, then the right hand side of (5.21) is 2. Therefore, $t_k = 1$ satisfies the Armijo condition, implying the Armijo condition need not be performed and the gradient projection algorithm converges. So, including a stepsize and Armijo line-search method may not improve the performance of Browet and Van Dooren's algorithm because a stepsize of 1 guarantees convergence to a stationary point if the similarity metric is well-defined.

Even if we reframe Browet and Van Dooren's iterative algorithm into the Euclidean optimization framework, the computational complexity of the Euclidean update (5.20) is $O(n^3)$, which is the computational complexity of Browet and Van Dooren's full-rank iterative algorithm. So, the full-rank Euclidean optimization approach is inefficient for large networks. Therefore, more aggressive low-rank optimization approaches, such as Riemannian optimization, are needed to solve the full-rank optimization approach efficiently.

CHAPTER 6

UNSIGNED NETWORK EXPERIMENTS FOR TWO-PHASE ALGORITHMS

In this chapter, we evaluate empirically our two-phase indirect approach to the role extraction problem. In Section 6.1, using random unweighted Erdós-Rényi graphs containing a structural block distribution of their nodes, we compare our Riemannian optimization approach to Browet and Van Dooren's low-rank neighborhood pattern similarity iterative algorithm (Section 2.4.3) and the low-rank similarity measure of Cheng et al. (Section 2.4.4). To extract the roles from the similarity approximations, we compare k-means clustering with Browet et al.'s community detection algorithm. The community detection algorithm uses either the configuration null model (CNM) or the Constant Potts Model (CPM) as the cost function, and k-means clustering uses either the silhouette statistic or the gap statistic. We compare the normalized mutual information (NMI) of the extracted role structure with the exact role structure to assess the effectiveness of the algorithms.

Lastly, in Section 6.2 we investigate our approach on a weighted network defined by trade data from several manufactures of metal among 80 countries in 1994.

6.1 Erdós-Rényi Graphs: Unweighted Networks

To assess the quality of our Riemannian optimization approach to the low-rank neighborhood pattern similarity approximation in comparison with Browet and Van Dooren's iterative algorithm, we analyze its accuracy on synthetic networks with known role structure. Our indirect approach to the role extract problem consists of two phases:

(1) From the adjacency matrix of the input graph, compute the low-rank factor $X^{(r)}$ of the neighborhood patterns similarity approximation $S^{(r)} = X^{(r)}(X^{(r)})^T$ from the optimization problem (4.7) for suitable values of β and r, and suitable optimization parameters.

(2) Extract the roles from the similarity approximation using k-means clustering with the silhouette statistic on the low-rank factor $X^{(r)}$.

(a) (b)

Figure 6.1: Erdós-Rényi Role Graphs: (a) block cycle role structure; and (b) almost isomorphic role structure.

Additionally, to asses specifically the quality of the silhouette statistic in the second phase of our role extraction approach, we also use Browet et al.'s community detection algorithm with either CNM or CPM as the cost function on the full similarity approximation $S^{(r)}$, and k-means clustering with the gap statistic on the low-rank factor $X^{(r)}$.

As in [Bro14], we compute the similarity measure of Erdós-Rényi random graphs containing a block structure (see Figure 6.1). To build our graphs, we first select a directed role graph $G_B(V_B, E_B)$ where each node in G_B defines a role. We consider two different role structures: the block cycle structure (Figure 6.1a) and the almost isomorphic role structure (Figure 6.1b).

Given the role graph G_B, we build our random graph $G_A(V_A, E_A)$ in the same manner used in [Bro14] where each node in G_A has a corresponding role in G_B. Edges are added to E_A according to two probability parameters p_{in} and p_{out}. For every pair of nodes $i, j \in V_A$, an edge $(i, j) \in E_A$ is added with probability p_{in} if an edge between the corresponding roles exists in G_B. If the edge does not exist between the corresponding roles in G_B, then the edge is added with probability p_{out}. If p_{in} is much larger than p_{out} ($p_{in} \gg p_{out}$), then the role graph G_B is an accurate representation of the different roles in the graph G_A and it is expected that the pairwise similarity measure S^* between the vertices V_A should allow the roles to be extracted. On the other hand, if p_{out} is much larger than p_{in} ($p_{out} \gg p_{in}$), then the different roles in G_A are closely represented by the complement graph of G_B represented by the adjacency matrix $\mathbf{1}\mathbf{1}^T - B$. However, the role structure strongly exists in the complement graph and it is expected that the pairwise similarity measure S^* should differentiate between them. When p_{in} and p_{out} are close to each other, then it is difficult to extract the roles since this graph is close to a uniform Erdós-Rényi graph, which is known to be free of any structure [New10].

Once we create the random graph, we randomly permute the rows and columns of the adjacency matrix by the permutation matrix P to create the matrix PAP^T. The matrix PAP^T is the adjacency matrix to which the algorthims are applied to compute the low-rank neighborhood similarity approximation. Lastly, we apply either community detection or k-means clustering to extract the permutation P and retrieve the original adjacency matrix A.

The Riemannian algorithms, Browet and Van Dooren's low-rank iterative algorithm, and the Cheng et al.'s low-rank similarity measure computation were written in Matlab. The k-means clustering, silhouette statistic and gap statistic algorithms are internal Matlab functions located within the statistics library and Browet et al.'s community detection algorithm and cost functions are C++ functions. All algorithms are called within Matlab R2015a on a 64 bit Mac platform with 2.5 GHz and 4 GB of memory.

The Riemannian optimization methods used in the experiments are steepest descent (RSD) [AMS08], Newton (RNewton) [AMS08], and limited-memory BFGS (LRBFGS) [HGA15]. Due to the assumption that ideally the networks are large, we compare the Riemannian algorithms for the symmetric positive semidefinite fixed-rank manifold as a quotient space with respect to our special metric (Section 4.1.2). We represent the tangent vectors by their intrinsic representation and use vector transport by parallelization (4.25).

All optimization algorithms are line-search algorithms and use the back tracking line search algorithm to determine an Armijo point [AMS08, Definition 4.2.2]. The initial stepsize is taken to be 1, the coefficient c_1 in the Armijo condition is 10^{-4}, and the ratio ρ for the decreasing the stepsize is 0.25. For LRBFGS, the number of rank-1 updates is $m = 4$ [HGA15]. The linear system in the RNewton method is solved using a truncated conjugate gradient (CG) [Ste83, CGT00] and requires the action of Hessian. The θ and κ parameters in the inner iteration stopping criterion [AMS08, (7.10)] of the truncated CG were set to 1 and 0.1.

The optimization algorithms start from two different initial conditions. The first initial condition (IC1) is the initial iterate of Browet and Van Dooren's low-rank algorithm (lines 1 and 2 in Algorithm 1). The second initial condition (IC2) is generated by a random subspace iteration in the subspace of $[A \mid A^T]$ with oversampling parameter $l = r + 5$ [HMT11]. The factors U and Σ of $[A \mid A^T]$ are truncated to be rank r, i.e., $X_1 = U_r \Sigma_r$ where U_r is an $n \times r$ orthogonal matrix and Σ_r is an $r \times r$ diagonal matrix of the singular values.

Table 6.1: Notation for reporting experimental results.

$iter$	number of iterations
nf	number of function evaluations
ng	number of gradient evaluations
nH	number of of operations of the form $\mathcal{H}\eta$
nV	number of vector transport operations
nR	number of retraction evaluations
f_f	final function value
gf_f	Riemannian metric value of the final gradient
$Stop\ value$	stopping criterion gf_f for Riemannian and (2.31) for Browet LR
$time$	time (seconds)
NMI	normalized mutual information

The stopping criterion for the optimization methods requires the norm of the Riemannian final gradient to be less than 10^{-6}, and for the algorithm of Browet and Van Dooren $\epsilon = 10^{-6}$ in (2.31). The maximum number of iterations allowed for the algorithm of Browet and Van Dooren and outer iterations for the optimization algorithms is 5000. Table 6.1 summarizes the notation used when describing the experimental results.

6.1.1 Measuring the Quality of a Partition

To compare the results of each algorithm, we need a quantitative criterion to measure how close the extracted partitions are to the exact partitions. In community detection, it is a common practice to use measures based on information theory. Since clustering problems are a generalization of community detection problems, we use the normalized mutual information (NMI) to measure the quality of the extracted partition compared to the exact partition.

Assuming X is an event that may occur with probability $p(x)$, the information contained in event X is defined by

$$I(X) = -\log(p(x)). \tag{6.1}$$

The joint entropy measures the uncertainty of the joint probability distribution $p(x, y)$ to observe $X = x$ and $Y = y$ and is defined by

$$H(X, Y) = -\sum_{X=x, Y=y} p(x, y) \log(p(x, y)) = H(Y, X). \tag{6.2}$$

If we observe a particular realization of the random variable $Y = y$, then the entropy of the probability distribution of X is given by

$$H(X|Y = y) = -\sum_{X=x} p(x|y)\log(p(x|y)), \tag{6.3}$$

and the conditional entropy is defined as

$$H(X|Y) = -\sum_{X=x,Y=y} p(x,y)\log\left(\frac{p(x,y)}{p(y)}\right), \tag{6.4}$$

where $p(x,y) = p(y)p(x|y) = p(x)p(y|x)$. The probability that a node randomly taken belongs to community x in partition X is

$$p(x) = \frac{n_x}{n}, \tag{6.5}$$

where n_x is the number of nodes in community x in partition X, and

$$p(x,y) = \frac{n_{xy}}{n}, \tag{6.6}$$

where n_{xy} is the number of nodes that belong to community x in partition X and to community y in partition Y. The mutual information between X and Y is the difference between the entropy of X and the condition entropy of X given Y, i.e.,

$$I(X,Y) = H(X) - H(X|Y) = \sum_{X=x,Y=y} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right). \tag{6.7}$$

Observe that $I(X,Y) = 0$ when X and Y are independent. Also, the closer X and Y are to each other, then the larger $I(X,Y)$ becomes. However, it has been shown that the mutual information cannot accurately distinguish between different sub-partitions. Thus, Danon et al. introduced in [DDGDA05] the normalized mutual information

$$NMI(X,Y) = \frac{I(X,Y)}{\sqrt{H(X)}\sqrt{H(Y)}}, \tag{6.8}$$

which is bounded between 0 and 1. NMI equal to 1 indicates identical partitions, while NMI equal to 0 (in the limit) indicates independent partitions.

6.1.2 Low-Rank Neighborhood Pattern Based Similarity Approximation Relative Error Comparison

For our first set of experiments, we compare the similarity matrices of the low-rank approaches for unweighted random Erdós-Rényi graphs with the two different role structures shown in Figure 6.1. Figures 6.2 and 6.3 display the relative error for both Erdós-Rényi graphs, Each figure is

divided into four panels corresponding to the p_{in} and p_{out} pairs $(0.9, 0.1)$, $(0.8, 0.2)$, $(0.7, 0.3)$, and $(0.6, 0.4)$. The relative error for each fixed-rank r is computed as

$$\frac{\|S^{(r)} - S^*\|_F}{\|S^*\|_F} \tag{6.9}$$

where $S^{(r)} = XX^T$ is the approximate low-rank similarity matrix computed from either the algorithm of Browet and Van Dooren or our optimization approach and S^* is the fixed point given by (2.24). This reveals the minimum rank required for $S^{(r)}$ to be a qualitatively good approximation of S^*.

In each panel, we show the adjacency matrix A of the random graph G_A, the random permutation of the adjacency matrix PAP^T, the role assignment of each nodes as extracted by k-means clustering applied to the low-rank factor $X \in \mathbb{R}_*^{n \times r}$ when r equals the number of roles for both graphs, and the relative error trend as r increases from 1 to 15.

From the figures, observe that the relative error for all of the Riemannian optimization methods is the same as Browet and Van Dooren's low-rank iterative method for both role graphs. Also, the minimal rank necessary for the low-rank similarity matrix to be close to the full-rank similarity matrix is revealed to be equal to the number of roles when $(p_{in}, p_{out}) = (0.9, 0.1)$. However, the minimal rank becomes less obvious as more noise is added to the network by changing p_{in} and p_{out} to be closer to each other. For Figure 6.3, there is a larger drop in the relative error between $r = 1$ and $r = 2$ than between $r = 2$ and $r = 3$, and, as the noise in the graph increases, the minimal numerical rank necessary to approximate the neighborhood pattern similarity measure reduces from 3 to 2. Therefore, when using a low-rank approximation method for the similarity measure, it is better to assume that the rank is larger than the number of roles because the low-rank approximation is closer the full-rank similarity measure.

In addition, notice that the relative error for $r \geq 3$ increases as the roles become less distinct in as p_{in} and p_{out} get closer in value. Thus, in graphs free of any structure, determining an appropriate rank for the fixed-rank neighborhood pattern similarity algorithms is difficult. For problems where there exists some type of structure within the network, using a low-rank method to approximate the neighborhood pattern similarity measure should be appropriate but the feasibility of techniques for detecting the likelihood that no significant role structure exists is an open question.

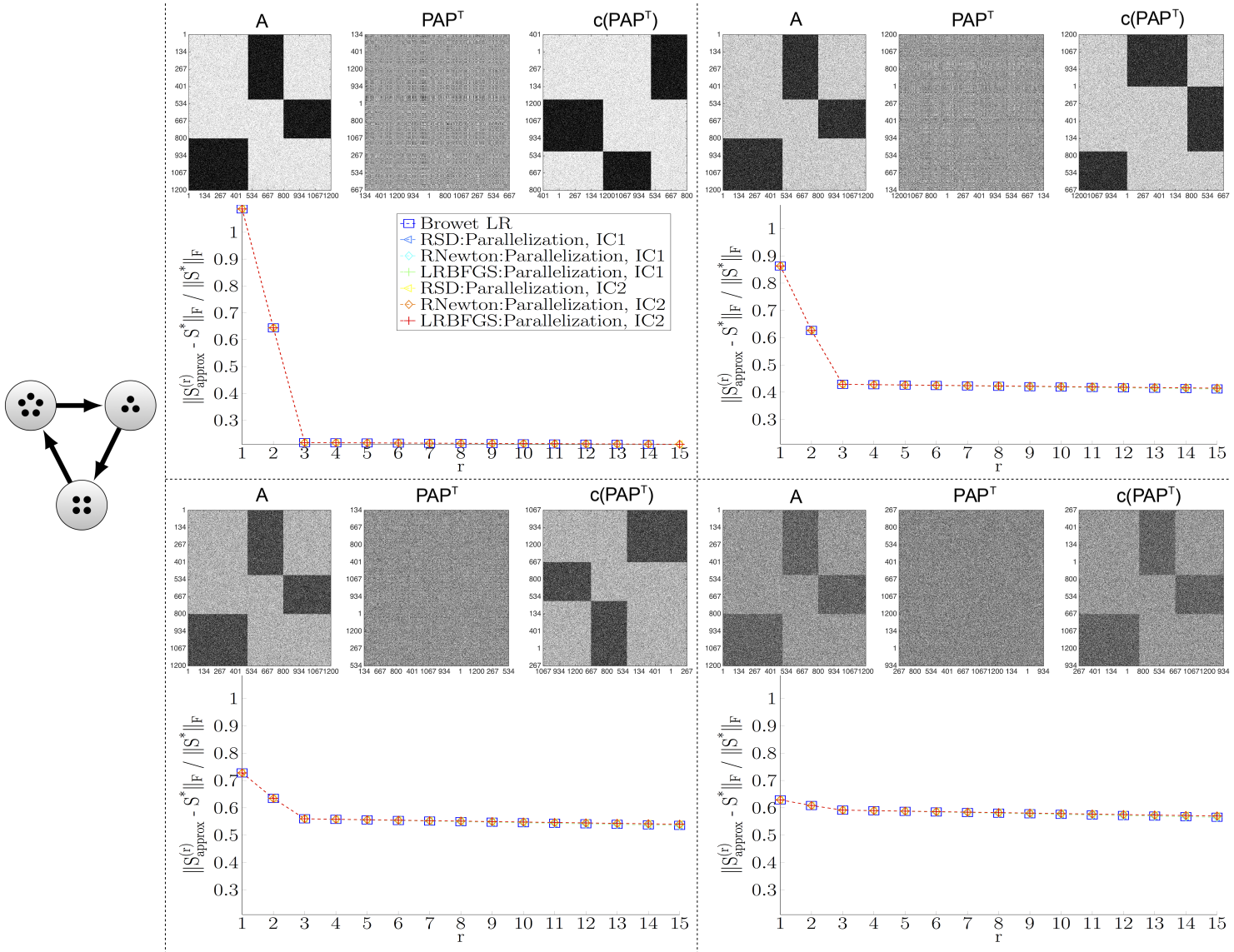


Figure 6.2: Results for role graph (a). Top Left: $(p_{in}, p_{out}) = (0.9, 0.1)$. Top Right: $(p_{in}, p_{out}) = (0.8, 0.2)$. Bottom Left: $(p_{in}, p_{out}) = (0.7, 0.3)$. Bottom Right: $(p_{in}, p_{out}) = (0.6, 0.4)$.

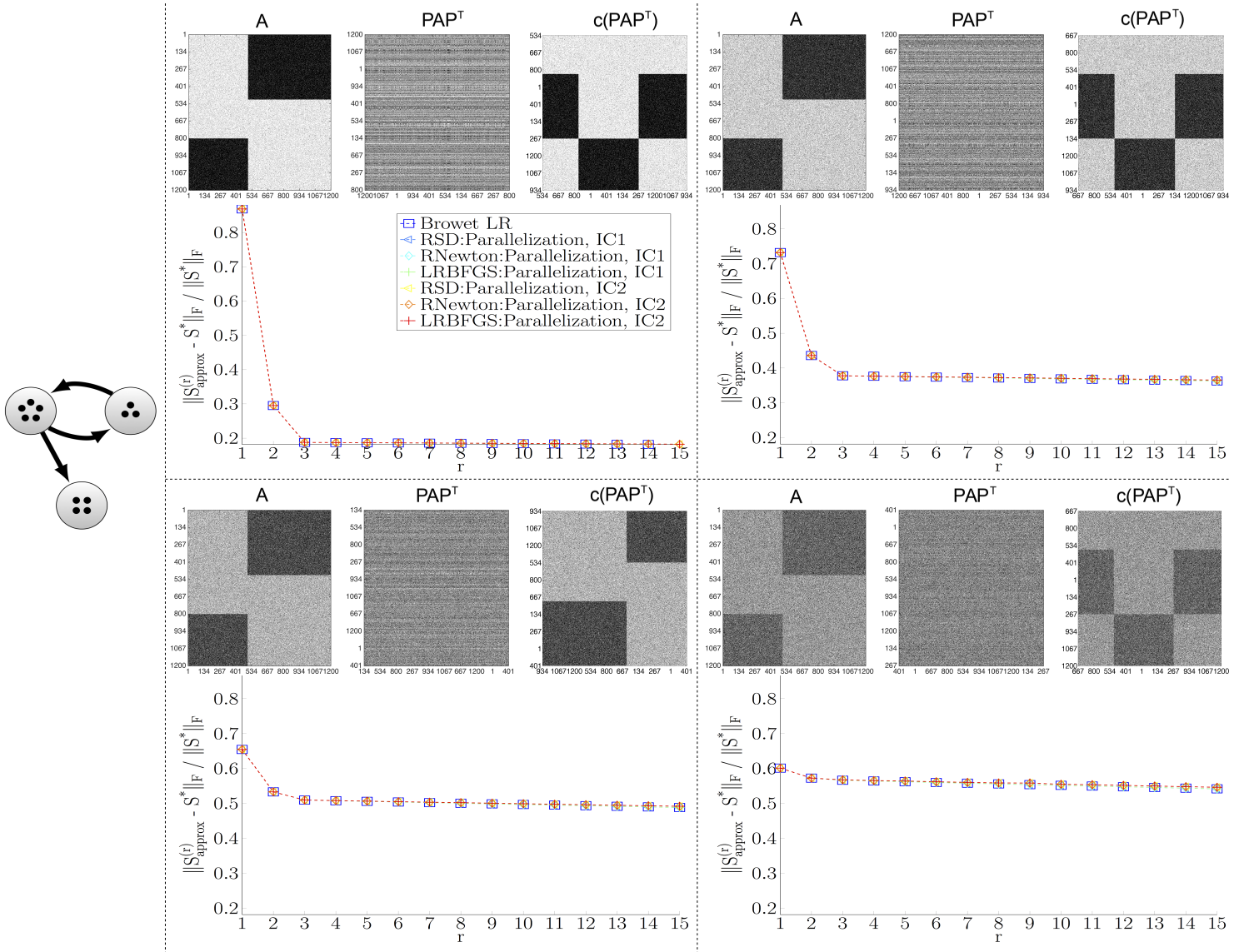


Figure 6.3: Results for role graph (b). Top Left: $(p_{in}, p_{out}) = (0.9, 0.1)$. Top Right: $(p_{in}, p_{out}) = (0.8, 0.2)$. Bottom Left: $(p_{in}, p_{out}) = (0.7, 0.3)$. Bottom Right: $(p_{in}, p_{out}) = (0.6, 0.4)$.

6.1.3 Low-Rank Similarity Approximation Time Comparison

Next, we compare the average time (out of 5 runs) to complete Browet and Van Dooren's low-rank algorithm, Riemannian optimization approaches, and the low-rank similarity measure of Cheng et al. for the unweighted random Erdós-Rényi graphs. We compare the time for both role structures in Figure 6.1 for p_{in} and p_{out} pairs $(0.9, 0.1)$ and $(0.7, 0.3)$. The size of the adjacency matrix is fixed to $n = 1200$ where there are 500 nodes in the first role, 300 nodes in the second role, and 400 nodes in the third role.

Table 6.2 compares the time to compute the two initial conditions for both role graphs as r increases from 1 to 15. Observe that as r increases, the difference in time between IC1 and IC2 also increases, where IC1 takes more time to compute than IC2. This difference in time is noticeable for both p_{in} and p_{out} pairs in both examples. However, while computing the initial condition can be costly if n and r are both large, we are mainly concerned about the computational time it takes to approximate the low-rank similarity matrix. Therefore, when we compare the computational time of the neighborhood pattern similarity matrices, we isolate the computation time of the initial condition from the time it takes for the low-rank algorithms for the neighborhood pattern similarity matrices. However, when we compare the computational time of the neighborhood pattern similarity matrix with the similarity matrix of Cheng et al., we include the initial condition time, which is labeled as *Total time* in the tables.

Recall that the parameter β determines the weight of the long neighborhood patterns for the neighborhood pattern similarity measure. Also, by Theorem 5.2.1, we have for nonnegative matrices

Table 6.2: Table of times (seconds) to compute the initial condition (iterate) for role graphs (a) and (b). The subscript ν indicates a scale of 10^{ν}

Graph	(p_{in}, p_{out})	r	1	2	3	4	5	10	15
(a)	$(0.9, 0.1)$	IC1 time	5.88_{-1}	1.17_{-1}	8.33_{-1}	1.98	2.10	1.47	1.06
		IC2 time	1.71_{-1}	5.77_{-2}	5.53_{-2}	6.27_{-2}	8.39_{-2}	7.99_{-2}	9.33_{-2}
	$(0.7, 0.3)$	IC1 time	1.33_{-1}	1.17_{-1}	5.14_{-1}	9.82_{-1}	1.45	2.56	2.02
		IC2 time	5.68_{-2}	6.40_{-2}	6.34_{-2}	7.50_{-2}	7.24_{-2}	9.46_{-2}	1.10_{-1}
(b)	$(0.9, 0.1)$	IC1 time	1.12_{-1}	1.09_{-1}	3.67_{-1}	7.86_{-1}	1.38	1.09	1.37
		IC2 time	5.39_{-2}	5.95_{-2}	5.95_{-2}	6.89_{-2}	7.12_{-2}	9.55_{-2}	1.04_{-1}
	$(0.7, 0.3)$	IC1 time	1.23_{-1}	1.20_{-1}	3.96_{-1}	7.82_{-1}	1.01	1.62	1.89
		IC2 time	5.82_{-2}	6.65_{-2}	6.48_{-2}	7.50_{-2}	7.57_{-2}	1.00_{-1}	1.22_{-1}

that

$$\beta^2 \leq \frac{1}{\rho(A + A^T)^2} \leq \frac{1}{2\|A\|^2} < \frac{1}{\rho((A \otimes A) + (A \otimes A)^T)}. \tag{6.10}$$

Therefore, we compare the average low-rank algorithm times for three different β values:

$$\beta_1 = \frac{0.95}{\rho(A + A^T)}, \quad \beta_2 = 0.5\left(\frac{1}{\rho(A + A^T)} + \frac{1}{\sqrt{2\|A\|^2}}\right), \quad \text{and } \beta_3 = \frac{0.95}{\sqrt{2\|A\|^2}}. \tag{6.11}$$

In Figures 6.4 and 6.5, we compare the times of low-rank similarity algorithms for the neighborhood pattern similarity measure for the three different β's as the rank r increases from 1 to 15. Tables 6.3 and 6.4 displays the number of iterations, computational time of the iteration (or optimization) (in seconds), final stopping criterion value, and overall total time (in seconds) for all of the algorithms, and the number of computations for the Riemannian algorithms.

When β is less than the lower bound, the computational times of all the algorithms are approximately the same for all r. However, the time for RNewton for $(p_{in}, p_{out}) = (0.9, 0.1)$ when $r \geq 7$ for both role graphs is increased noticeably. The increase in time as r is chosen larger than the number of roles is probably due to some type of degeneracy in the Hessian, which cause the number of Hessian computations during the truncated conjugate gradient line search to increase (see Tables 6.3 and 6.4). Thus, if we knew the number of roles, or could assume a rank very close to the number of roles, all of the algorithms would converge rapidly to a solution.

For β_2, observe that RSD and Browet both slow down, while RNewton (except for $(p_{in}, p_{out}) = (0.9, 0.1)$) and LRBFGS are approximately the same time to approximate a solution. The degradation in time for RSD may be due to the scaling of the problem and applying a scaling matrix to RSD may improve performance. While RSD is competitive in time with Browet, observe from Tables 6.3 and 6.4 that RSD converges in fewer iterations. Also, observe that LRBFGS is the fastest algorithm for all fixed ranks. In particular, for role graph (b) and $(p_{in}, p_{out}) = (0.7, 0.3)$, observe in Figure 6.5 that the difference in time increases between Browet and LRBFGS as r increases. Since, in practice, real data graphs tend to be noisy and r is unknown, this implies that LRBFGS is superior to Browet and Van Dooren's low-rank algorithm when approximating the similarity matrix for large networks, especially for problems where we must consider several values of r to determine an acceptable set of roles.

Lastly, we use longer neighborhood patterns by setting β equal to β_3. From Figure 6.4, observe that RSD slowed down significantly after $r = 3$ (the number of roles). Also, note that for LRBFGS

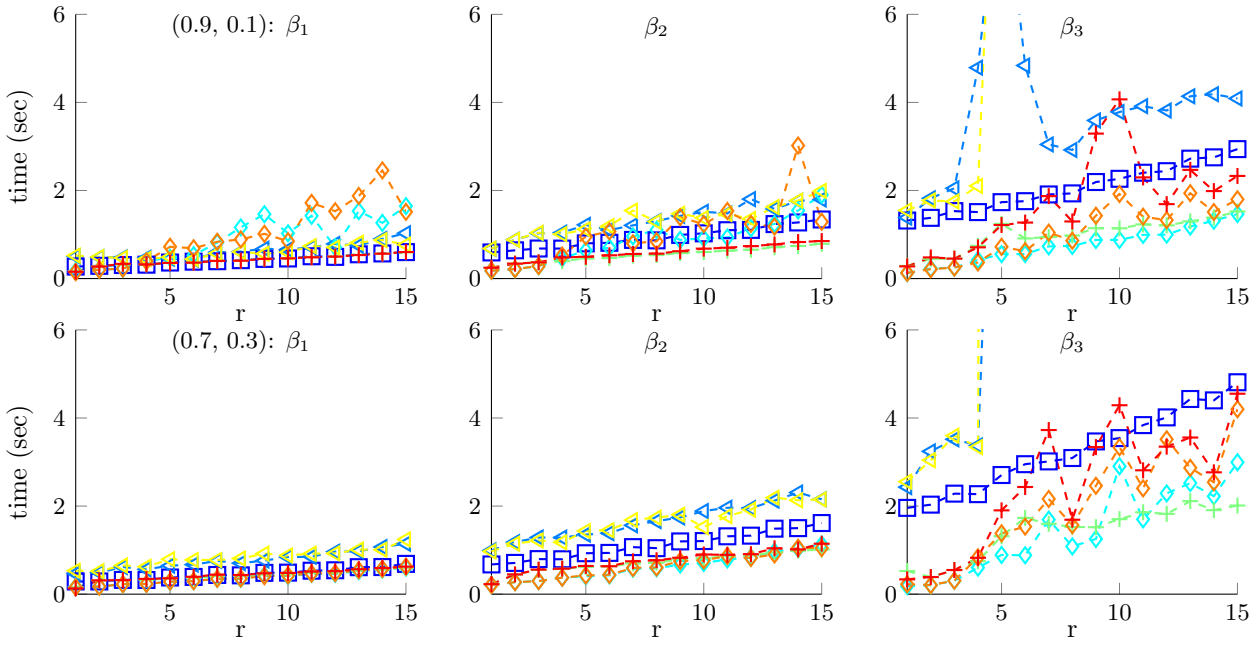


Figure 6.4: Average time to compute the similarity approximation for role graph (a).

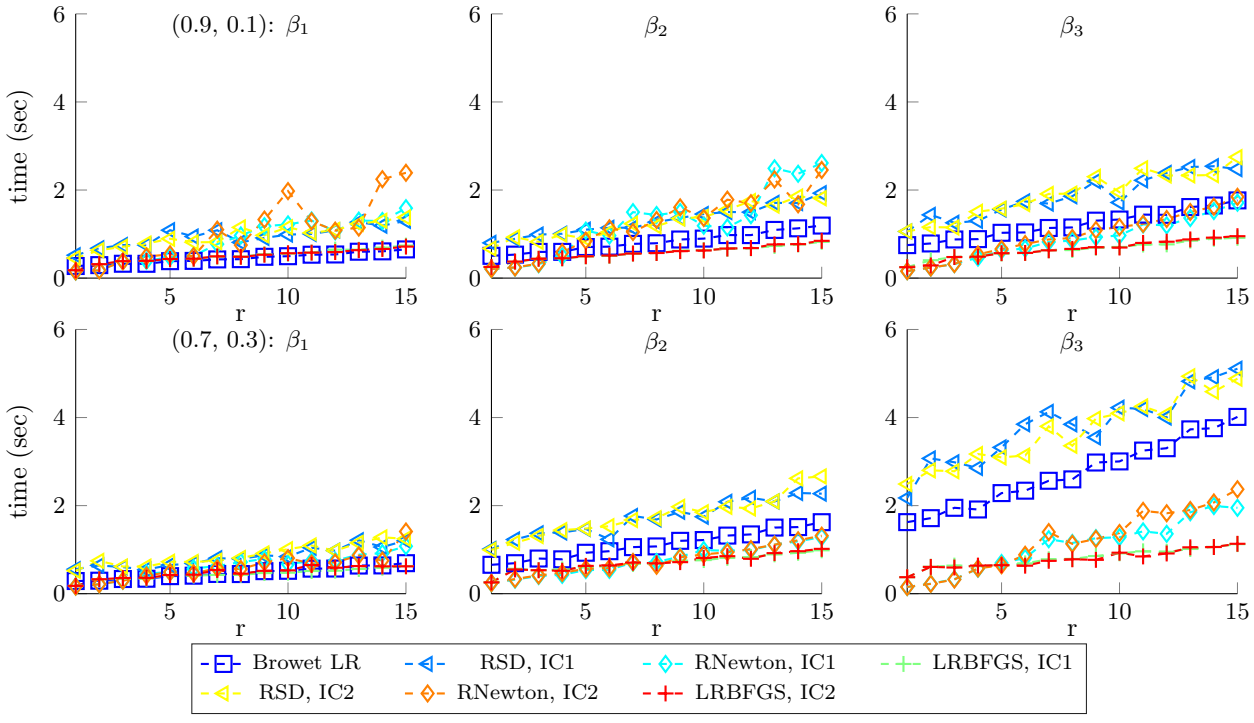


Figure 6.5: Average time to compute the similarity approximation for role graph (b).

Table 6.3: Results for role graph (a) and $(p_{in}, p_{out}) = (0.9, 0.1)$. The subscript ν indicates a scale of 10^{ν}.

Algorithm	r	2			3			4			10			15		
		β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
Browet LR	*iter*	34	77	172	34	77	172	34	77	172	34	77	172	34	77	172
	Stop val	6.90_{-7}	9.04_{-7}	8.62_{-7}	6.88_{-7}	9.14_{-7}	8.76_{-7}	6.90_{-7}	9.11_{-7}	8.73_{-7}	6.90_{-7}	9.09_{-7}	8.77_{-7}	6.90_{-7}	9.11_{-7}	8.71_{-7}
	Iter time	2.89_{-1}	6.41_{-1}	1.38	3.08_{-1}	6.82_{-1}	1.53	3.15_{-1}	6.81_{-1}	1.51	4.56_{-1}	1.03	2.27	6.00_{-1}	1.34	2.94
	Total time	4.07_{-1}	6.42_{-1}	1.38	1.14	6.83_{-1}	1.53	2.29	6.81_{-1}	1.51	1.93	1.03	2.27	1.66	1.34	2.94
RSD IC1	*iter*	31	58	120	28	62	124	27	62	256	26	63	163	31	57	138
	nf	51	79	140	48	80	144	47	81	273	45	80	184	49	75	158
	ng	32	59	121	29	63	125	28	63	257	27	64	164	32	58	139
	nR	50	78	139	47	79	143	46	80	272	44	79	183	48	74	157
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	8.18_{-7}	9.61_{-7}	8.58_{-7}	4.96_{-7}	7.30_{-7}	9.25_{-7}	4.58_{-7}	7.44_{-7}	9.55_{-7}	5.13_{-7}	7.15_{-7}	9.46_{-7}	4.84_{-7}	8.17_{-7}	9.47_{-7}
	Opt time	4.84_{-1}	8.84_{-1}	1.82	4.82_{-1}	1.03	2.05	4.69_{-1}	1.04	4.79	6.58_{-1}	1.52	3.78	1.03	1.80	4.09
	Total time	4.84_{-1}	8.85_{-1}	1.82	4.82_{-1}	1.03	2.05	4.69_{-1}	1.04	4.79	6.59_{-1}	1.52	3.78	1.03	1.80	4.09
RSD IC2	*iter*	32	58	117	28	63	107	26	60	126	23	57	2724	22	63	490
	nf	52	78	135	48	80	124	45	79	144	44	74	2748	41	81	513
	ng	33	59	118	29	64	108	27	61	127	24	58	2725	23	64	491
	nR	51	77	134	47	79	123	44	78	143	43	73	2747	40	80	512
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	5.15_{-7}	9.74_{-7}	9.16_{-7}	5.26_{-7}	9.60_{-7}	9.43_{-7}	8.05_{-7}	8.27_{-7}	9.42_{-7}	6.98_{-7}	8.18_{-7}	9.97_{-7}	5.38_{-7}	8.01_{-7}	9.72_{-7}
	Opt time	4.91_{-1}	8.82_{-1}	1.77	4.70_{-1}	1.04	1.75	4.44_{-1}	1.01	2.10	5.98_{-1}	1.38	2.37_{2}	7.68_{-1}	1.99	1.53_{1}
	Total time	4.91_{-1}	8.82_{-1}	1.77	4.70_{-1}	1.04	1.75	4.45_{-1}	1.01	2.10	5.98_{-1}	1.38	2.38_{2}	7.69_{-1}	1.99	1.53_{1}
RNewton IC1	*iter*	10	10	9	9	10	9	16	16	12	20	17	12	21	20	12
	nf	12	12	10	11	19	10	23	28	13	33	32	13	37	35	13
	ng	11	11	10	10	11	10	17	17	13	21	18	13	22	21	13
	nH	23	29	33	24	32	30	32	45	58	49	44	59	55	73	68
	nR	11	11	9	10	18	9	22	27	12	32	31	12	36	34	12
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	9.53_{-10}	1.11_{-9}	2.90_{-9}	5.20_{-10}	2.42_{-9}	4.81_{-9}	2.01_{-8}	1.12_{-7}	5.96_{-9}	8.82_{-7}	8.12_{-9}	4.30_{-9}	1.28_{-8}	2.05_{-8}	4.75_{-9}
	Opt time	2.03_{-1}	2.14_{-1}	2.28_{-1}	2.26_{-1}	2.82_{-1}	2.48_{-1}	4.01_{-1}	4.61_{-1}	4.22_{-1}	1.03	9.13_{-1}	8.79_{-1}	1.64	1.90	1.46
	Total time	2.03_{-1}	2.14_{-1}	2.28_{-1}	2.26_{-1}	2.82_{-1}	2.48_{-1}	4.01_{-1}	4.61_{-1}	4.23_{-1}	1.03	9.13_{-1}	8.79_{-1}	1.65	1.90	1.46
RNewton IC2	*iter*	10	10	9	9	10	9	15	18	11	15	23	16	17	17	13
	nf	12	12	10	11	19	10	24	31	12	21	38	18	25	33	14
	ng	11	11	10	10	11	10	16	19	12	16	24	17	18	18	14
	nH	23	29	33	24	32	30	35	50	47	46	61	136	57	41	85
	nR	11	11	9	10	18	9	23	30	11	20	37	17	24	32	13
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	6.79_{-10}	1.33_{-9}	2.07_{-9}	5.34_{-10}	3.36_{-9}	2.40_{-9}	6.11_{-9}	7.07_{-8}	8.31_{-9}	4.72_{-9}	3.30_{-9}	8.71_{-9}	9.92_{-7}	6.93_{-7}	3.36_{-8}
	Opt time	1.98_{-1}	2.15_{-1}	2.13_{-1}	2.26_{-1}	2.85_{-1}	2.50_{-1}	3.94_{-1}	5.22_{-1}	3.66_{-1}	8.48_{-1}	1.24	1.92	1.51	1.29	1.80
	Total time	1.99_{-1}	2.15_{-1}	2.13_{-1}	2.27_{-1}	2.85_{-1}	2.50_{-1}	3.95_{-1}	5.22_{-1}	3.66_{-1}	8.48_{-1}	1.24	1.92	1.51	1.29	1.80
LRBFGS IC1	*iter*	18	22	29	19	22	27	19	23	43	19	24	47	19	24	49
	nf	20	25	30	21	25	28	21	26	45	21	27	50	21	27	51
	ng	19	23	30	20	23	28	20	24	44	20	25	48	20	25	50
	nV	140	172	228	148	172	212	148	180	340	148	188	372	148	188	388
	nR	19	24	29	20	24	27	20	25	44	20	26	49	20	26	50
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	9.48_{-7}	9.52_{-7}	9.84_{-7}	8.55_{-8}	9.86_{-7}	6.11_{-7}	2.82_{-7}	9.57_{-7}	3.07_{-7}	2.02_{-7}	6.50_{-7}	4.86_{-7}	2.35_{-7}	7.74_{-7}	6.63_{-7}
	Opt time	2.80_{-1}	3.39_{-1}	4.49_{-1}	3.16_{-1}	3.67_{-1}	4.51_{-1}	3.20_{-1}	3.94_{-1}	7.42_{-1}	4.60_{-1}	6.04_{-1}	1.14	5.96_{-1}	7.89_{-1}	1.52
	Total time	2.80_{-1}	3.39_{-1}	4.49_{-1}	3.16_{-1}	3.68_{-1}	4.52_{-1}	3.20_{-1}	3.95_{-1}	7.42_{-1}	4.60_{-1}	6.04_{-1}	1.14	5.97_{-1}	7.89_{-1}	1.52
LRBFGS IC2	*iter*	18	22	31	19	22	27	19	28	41	19	28	160	19	27	75
	nf	20	25	32	21	25	28	21	31	43	21	31	165	21	30	77
	ng	19	23	32	20	23	28	20	29	42	20	29	161	20	28	76
	nV	140	172	244	148	172	212	148	220	324	148	220	1276	148	212	596
	nR	19	24	31	20	24	27	20	30	42	20	30	164	20	29	76
	f_f	-1.59_{11}	-2.75_{11}	-5.33_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}	-1.87_{11}	-3.16_{11}	-5.87_{11}
	Stop val	9.47_{-7}	9.54_{-7}	1.63_{-7}	8.60_{-8}	9.87_{-7}	6.02_{-7}	3.53_{-7}	1.55_{-7}	7.86_{-7}	3.69_{-7}	2.59_{-7}	7.34_{-7}	7.13_{-7}	6.07_{-7}	7.35_{-7}
	Opt time	2.74_{-1}	3.39_{-1}	4.80_{-1}	3.34_{-1}	3.69_{-1}	4.53_{-1}	3.21_{-1}	4.79_{-1}	7.09_{-1}	4.59_{-1}	6.80_{-1}	4.07	6.02_{-1}	8.52_{-1}	2.33
	Total time	2.74_{-1}	3.39_{-1}	4.80_{-1}	3.35_{-1}	3.69_{-1}	4.53_{-1}	3.22_{-1}	4.80_{-1}	7.09_{-1}	4.59_{-1}	6.80_{-1}	4.07	6.03_{-1}	8.52_{-1}	2.33
Cheng LR	*time*		1.68_{-1}			5.59_{-1}			1.21			1.52			1.25	

Table 6.4: Results for role graph (b) and $(p_{in}, p_{out}) = (0.9, 0.1)$. The subscript ν indicates a scale of 10^{ν}.

Algorithm	r	2			3			4			10			15		
		β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
Browet LR	*iter*	35	65	96	35	65	96	35	65	96	35	65	96	35	65	96
	Stop val	4.82_{-7}	7.59_{-7}	7.62_{-7}	5.08_{-7}	8.29_{-7}	8.56_{-7}	5.06_{-7}	8.25_{-7}	8.56_{-7}	5.07_{-7}	8.27_{-7}	8.58_{-7}	5.05_{-7}	8.26_{-7}	8.59_{-7}
	Iter time	2.95_{-1}	5.33_{-1}	7.87_{-1}	3.26_{-1}	5.95_{-1}	8.78_{-1}	3.24_{-1}	5.96_{-1}	8.85_{-1}	4.97_{-1}	8.97_{-1}	1.33	6.52_{-1}	1.19	1.76
	Total time	4.05_{-1}	5.34_{-1}	7.87_{-1}	6.93_{-1}	5.95_{-1}	8.78_{-1}	1.11	5.96_{-1}	8.85_{-1}	1.59	8.97_{-1}	1.33	2.02	1.19	1.77
RSD IC1	*iter*	38	51	82	39	52	66	38	52	67	35	53	63	35	54	71
	nf	55	71	101	58	74	87	57	71	86	55	75	85	54	77	92
	ng	39	52	83	40	53	67	39	53	68	36	54	64	36	55	72
	nR	54	70	100	57	73	86	56	70	85	54	74	84	53	76	91
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	7.28_{-7}	8.39_{-7}	9.73_{-7}	5.83_{-7}	6.55_{-7}	8.35_{-7}	7.43_{-7}	6.60_{-7}	8.82_{-7}	5.96_{-7}	7.33_{-7}	9.39_{-7}	9.08_{-7}	7.99_{-7}	8.73_{-7}
	Opt time	6.78_{-1}	8.93_{-1}	1.44	7.51_{-1}	9.89_{-1}	1.26	7.48_{-1}	1.01	1.30	9.88_{-1}	1.45	1.72	1.30	1.93	2.48
	Total time	6.78_{-1}	8.93_{-1}	1.44	7.51_{-1}	9.89_{-1}	1.26	7.49_{-1}	1.01	1.30	9.88_{-1}	1.45	1.72	1.30	1.93	2.48
RSD IC2	*iter*	36	53	66	37	43	60	40	51	78	41	50	72	38	51	79
	nf	52	72	85	53	64	81	58	71	96	57	73	91	55	75	98
	ng	37	54	67	38	44	61	41	52	79	42	51	73	39	52	80
	nR	51	71	84	52	63	80	57	70	95	56	72	90	54	74	97
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	5.50_{-7}	8.96_{-7}	9.96_{-7}	8.27_{-7}	8.43_{-7}	9.08_{-7}	7.15_{-7}	8.04_{-7}	9.10_{-7}	9.02_{-7}	6.84_{-7}	8.78_{-7}	9.21_{-7}	9.15_{-7}	9.37_{-7}
	Opt time	6.33_{-1}	9.28_{-1}	1.16	7.12_{-1}	8.24_{-1}	1.15	7.84_{-1}	9.88_{-1}	1.51	1.16	1.39	1.95	1.38	1.81	2.75
	Total time	6.33_{-1}	9.28_{-1}	1.16	7.12_{-1}	8.24_{-1}	1.15	7.84_{-1}	9.89_{-1}	1.51	1.16	1.39	1.95	1.38	1.81	2.75
RNewton IC1	*iter*	8	10	10	13	10	10	14	16	13	19	21	12	18	23	13
	nf	10	14	11	16	13	11	19	19	14	27	31	13	27	35	14
	ng	9	11	11	14	11	11	15	17	14	20	22	13	19	24	14
	nH	18	29	28	35	33	34	28	45	51	60	52	58	51	98	71
	nR	9	13	10	15	12	10	18	18	13	26	30	12	26	34	13
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	2.67_{-8}	5.45_{-9}	8.41_{-7}	1.08_{-9}	3.26_{-9}	3.66_{-9}	8.07_{-7}	4.51_{-8}	7.42_{-9}	2.29_{-8}	1.20_{-8}	5.38_{-8}	1.31_{-7}	1.90_{-8}	3.48_{-9}
	Opt time	1.77_{-1}	2.40_{-1}	2.38_{-1}	3.83_{-1}	3.16_{-1}	3.18_{-1}	3.92_{-1}	4.98_{-1}	4.65_{-1}	1.22	1.19	9.67_{-1}	1.59	2.61	1.71
	Total time	1.77_{-1}	2.40_{-1}	2.39_{-1}	3.84_{-1}	3.17_{-1}	3.18_{-1}	3.92_{-1}	4.98_{-1}	4.65_{-1}	1.22	1.19	9.68_{-1}	1.59	2.61	1.71
RNewton IC2	*iter*	8	10	10	13	10	10	17	18	15	26	24	14	21	25	14
	nf	10	14	11	16	13	11	25	24	18	35	35	15	27	39	15
	ng	9	11	11	14	11	11	18	19	16	27	25	15	22	26	15
	nH	18	29	28	35	33	34	37	53	58	111	60	70	91	83	77
	nR	9	13	10	15	12	10	24	23	17	34	34	14	26	38	14
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	2.68_{-8}	1.26_{-9}	8.42_{-7}	8.85_{-10}	1.57_{-9}	5.44_{-9}	2.07_{-9}	1.98_{-8}	7.30_{-9}	1.92_{-8}	1.96_{-8}	1.28_{-8}	2.19_{-8}	7.43_{-9}	2.89_{-9}
	Opt time	1.74_{-1}	2.42_{-1}	2.39_{-1}	3.80_{-1}	3.28_{-1}	3.19_{-1}	4.89_{-1}	5.84_{-1}	5.38_{-1}	1.98	1.36	1.16	2.39	2.46	1.85
	Total time	1.74_{-1}	2.42_{-1}	2.39_{-1}	3.80_{-1}	3.28_{-1}	3.19_{-1}	4.90_{-1}	5.84_{-1}	5.38_{-1}	1.98	1.36	1.16	2.39	2.46	1.85
LRBFGS IC1	*iter*	18	21	23	20	23	25	20	23	26	20	23	26	20	23	26
	nf	21	25	26	23	26	26	23	26	27	23	26	27	23	26	27
	ng	19	22	24	21	24	26	21	24	27	21	24	27	21	24	27
	nV	140	164	180	156	180	196	156	180	204	156	180	204	156	180	204
	nR	20	24	25	22	25	25	22	25	26	22	25	26	22	25	26
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	1.25_{-7}	3.39_{-7}	4.87_{-7}	7.27_{-8}	3.84_{-7}	4.14_{-7}	1.34_{-7}	6.81_{-7}	7.95_{-7}	1.34_{-7}	6.77_{-7}	4.96_{-7}	1.36_{-7}	6.57_{-7}	4.51_{-7}
	Opt time	3.13_{-1}	3.68_{-1}	4.06_{-1}	3.78_{-1}	4.38_{-1}	4.78_{-1}	3.90_{-1}	4.47_{-1}	5.06_{-1}	5.56_{-1}	6.37_{-1}	7.07_{-1}	7.38_{-1}	8.23_{-1}	9.14_{-1}
	Total time	3.13_{-1}	3.69_{-1}	4.06_{-1}	3.78_{-1}	4.38_{-1}	4.78_{-1}	3.90_{-1}	4.47_{-1}	5.06_{-1}	5.57_{-1}	6.37_{-1}	7.08_{-1}	7.38_{-1}	8.23_{-1}	9.15_{-1}
LRBFGS IC2	*iter*	18	21	13	20	23	25	20	23	25	20	23	25	20	24	27
	nf	21	25	14	23	26	26	23	26	26	23	26	26	24	27	28
	ng	19	22	14	21	24	26	21	24	26	21	24	26	21	25	28
	nV	140	164	100	156	180	196	156	180	196	156	180	196	156	188	212
	nR	20	24	13	22	25	25	22	25	25	22	25	25	23	26	27
	f_f	-3.43_{11}	-5.22_{11}	-7.01_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}	-3.45_{11}	-5.26_{11}	-7.08_{11}
	Stop val	1.24_{-7}	3.37_{-7}	1.76	7.35_{-8}	3.82_{-7}	4.17_{-7}	1.46_{-7}	5.69_{-7}	9.96_{-7}	1.93_{-7}	7.28_{-7}	5.34_{-7}	1.20_{-7}	6.97_{-7}	4.75_{-7}
	Opt time	3.17_{-1}	3.75_{-1}	2.88_{-1}	3.85_{-1}	4.39_{-1}	4.78_{-1}	3.92_{-1}	4.51_{-1}	4.86_{-1}	5.64_{-1}	6.27_{-1}	6.94_{-1}	7.15_{-1}	8.46_{-1}	9.52_{-1}
	Total time	3.17_{-1}	3.76_{-1}	2.88_{-1}	3.85_{-1}	4.40_{-1}	4.79_{-1}	3.92_{-1}	4.51_{-1}	4.87_{-1}	5.65_{-1}	6.27_{-1}	6.95_{-1}	7.16_{-1}	8.47_{-1}	9.52_{-1}
Cheng LR	*time*		1.82_{-1}			3.81_{-1}			7.36_{-1}			1.30			9.11_{-1}	

starting from IC2, the time fluctuated as r increased, while the LRBFGS starting from IC1 had a smoother increase in time. This indicates a sensitivity to the initial point of the algorithm when searching longer neighborhood patterns. Regardless of sensitivity, the optimization algorithms were still able to converge to a solution. In addition, RNewton and LRBFGS converged in significantly fewer iterations compared to the low-rank algorithm of Browet and Van Dooren (see Tables 6.3 and 6.4).

Comparing to total time of the low-rank neighborhood pattern similarity algorithms with the low-rank similarity measure of Cheng et al., the RNewton and LRBFGS are faster or competitive with Cheng et al.'s similarity measure in time. However, we will show later for role graph (b) that, while the low-rank similarity measure of Cheng et al. is competitive with the Riemannian algorithms in time, the clustering algorithms have difficulty extracting the role partition from Cheng et al.'s similarity measure compared to the low-rank neighborhood pattern similarity matrices.

Therefore, we see that for β small, i.e., $\beta = \beta_1$, all of the low-rank algorithms for the neighborhood pattern similarity measure are competitive in time. However, as we allow for longer neighborhood patterns and larger rank, LRBFGS starting from IC1 is significantly faster than Browet and Van Dooren's low-rank iterative algorithm and the other Riemannian algorithms. This speed and robustness is particularly important for problems where an appropriate beta is not known and several values must be considered.

6.1.4 Low-Rank Similarity Approximation Normalized Mutual Information Comparison

Next, from the similarity measures computed in Section (6.1.3), we compare the quality of the role partitions are extracted using NMI.

Tables 6.5 and 6.6 display the NMI and time of k-means clustering with the gap and silhouette statistics and community detection with CNM as the cost function. CPM is used for role graph (b), when two of the three roles are almost isomorphic, meaning it is difficult to distinguish between the two roles. For k-means clustering with the gap and silhouette statistics, we test the cluster list $\{1, \ldots, r+1\}$ and $\{2, \ldots, r+1\}$, respectively, and choose the optimal number of clusters based on the discussions in Sections 2.6.1 and 2.6.2. Note that the k-means algorithm is a Matlab function while the community detection algorithm is a C++ algorithm called within Matlab. Thus, while

we report the time of the algorithms, we do not compare the times between k-means clustering and community detection only the times for each individual algorithm as the rank increases.

Since k-means clustering starts with a random initial centroid and Browet et. al's community detection algorithm is a greedy hierarchical algorithm, we run both clustering algorithms 5 times, where k-means clustering chooses the partition with the smallest sum of distance and community detection chooses the partition with maximum modularity and compare the NMI value of the chosen partition. The time (in seconds) is the total time of the 5 runs. In addition, the time to compute XX^T is added to the algorithm time for community detection. Since all of the Riemannian algorithms approximate the same low-rank similarity matrix and LRBFGS is the fastest Riemannian algorithm, we only consider LRBFGS in the tables and compare it with Browet and Van Dooren's low-rank similarity algorithm and Cheng et al. low-rank similarity matrix.

From the tables, observe that for k-means clustering the NMI was 1.00 when r is equal to the number of roles. For graph (a), both clustering statistics are able to determine the optimal role partition from the neighborhood pattern similarity measures for $r \geq 2$ and all β's. However, for graph (b), the silhouette statistic fails to extract the role partition for $(p_{in}, p_{out}) = (0.7, 0.3)$ for the low-rank neighborhood similarity approximations when $r = 15$ and $\beta = \beta_1$. When β is equal to β_2 or β_3, the silhouette statistic extracts the role partition (see Table 6.6). For more complicated role structures, such as role graph (b), longer neighborhood patterns are preferred for the silhouette statistic to extract the role partition when r is much larger than the number of roles.

For role graph (b), the silhouette statistic only extracts a 2 role partition from Cheng et al. low-rank similarity matrix for $r = 2$ and 15 for $(p_{in}, p_{out}) = (0.9, 0.1)$ and $r = 2$, 10 and 15 when $(p_{in}, p_{out}) = (0.7, 0.3)$. This is because their similarity matrix is a low-rank approximation of a pairwise node self-similarity measure and has trouble extracting the two isomorphic roles. Thus, for this example, the longer neighborhood patterns are necessary to extract the role structure, favoring the Riemannian algorithms for the first phase.

The gap statistic for the low-rank neighborhood similarity approximations is able to extract the role partition for $r = 15$. It takes about 80 seconds to extract the role partition while the time to determine the optimal partition for the silhouette statistic is closer to 3 seconds. Thus, while the gap statistic may be more robust than the silhouette statistic, its computation time is significantly longer and it is not recommend to use the gap statistic on large datasets.

Table 6.5: Comparison of NMI and time (seconds) between k-means clustering and community detection with CNM for role graph (a). SC is the silhouette coefficient. The subscript ν indicates a scale of 10^{ν}.

(p_{in}, p_{out})	Algorithm		r	2			3			4			10			15		
				β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
$(0.9, 0.1)$	Browet LR	Gap	*time*	3.94	3.75	3.73	7.25	6.95	7.03	1.05_1	1.05_1	1.03_1	4.27_1	4.25_1	4.32_1	8.06_1	8.05_1	8.14_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.43_{-1}	1.23_{-1}	1.21_{-1}	2.15_{-1}	2.20_{-1}	2.18_{-1}	3.22_{-1}	3.23_{-1}	3.39_{-1}	1.61	1.62	1.63	3.14	3.12	3.16
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.99_{-1}	9.99_{-1}	9.99_{-1}	1.00	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.94_{-1}	9.95_{-1}	9.96_{-1}	9.90_{-1}	9.92_{-1}	9.93_{-1}
		CNM	*time*	4.68	4.81	4.92	4.74	4.90	4.95	4.60	4.70	4.71	4.62	4.88	5.12	4.63	4.71	4.68
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	LRBFGS IC1	Gap	*time*	4.03	3.77	3.75	7.11	7.15	7.17	1.05_1	1.04_1	1.06_1	4.25_1	4.30_1	4.28_1	8.10_1	8.07_1	8.03_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	2.32_{-1}	1.21_{-1}	1.21_{-1}	2.24_{-1}	2.15_{-1}	2.15_{-1}	3.24_{-1}	3.31_{-1}	3.29_{-1}	1.64	1.63	1.77	3.68	3.14	3.67
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.99_{-1}	9.99_{-1}	9.99_{-1}	1.00	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.94_{-1}	9.95_{-1}	9.96_{-1}	9.90_{-1}	9.92_{-1}	9.93_{-1}
		CNM	*time*	4.48	4.68	4.63	4.48	4.62	4.65	4.50	4.85	4.69	4.45	5.29	4.62	4.48	5.31	4.70
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	LRBFGS IC2	Gap	*time*	3.72	3.79	3.76	7.07	7.11	7.07	1.04_1	1.03_1	1.06_1	4.25_1	4.27_1	4.25_1	8.08_1	8.10_1	8.06_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.22_{-1}	1.21_{-1}	1.22_{-1}	2.17_{-1}	2.18_{-1}	2.17_{-1}	3.28_{-1}	3.21_{-1}	3.23_{-1}	1.62	1.62	1.61	3.10	3.14	3.12
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.99_{-1}	9.99_{-1}	9.99_{-1}	1.00	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.95_{-1}	9.96_{-1}	9.96_{-1}	9.92_{-1}	9.93_{-1}	9.94_{-1}
		CNM	*time*	4.88	4.71	4.56	5.02	4.57	4.65	4.90	4.84	4.58	4.92	4.64	4.76	4.60	4.59	4.64
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Cheng LR	Gap	*time*		3.73			7.18			1.02_1			4.22_1			8.01_1	
			NMI		1.00			1.00			1.00			1.00			1.00	
		Silhouette	*time*		1.24_{-1}			2.20_{-1}			3.29_{-1}			1.61			3.14	
			NMI		1.00			1.00			1.00			1.00			1.00	
			SC		9.98_{-1}			1.00			9.99_{-1}			9.92_{-1}			9.87_{-1}	
		CNM	*time*		5.73			5.42			5.43			5.94			5.99	
			NMI		7.73_{-1}			1.00			1.00			1.00			1.00	
$(0.7, 0.3)$	Browet LR	Gap	*time*	3.74	3.75	3.76	7.10	7.12	6.82	1.04_1	1.05_1	1.02_1	4.40_1	4.38_1	4.36_1	8.12_1	8.01_1	8.13_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	8.06_{-1}	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.22_{-1}	1.22_{-1}	1.25_{-1}	2.15_{-1}	2.13_{-1}	2.16_{-1}	3.28_{-1}	3.28_{-1}	3.20_{-1}	1.64	1.61	1.58	3.14	3.14	3.12
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.90_{-1}	9.90_{-1}	9.83_{-1}	9.96_{-1}	9.96_{-1}	9.94_{-1}	9.87_{-1}	9.87_{-1}	9.83_{-1}	9.36_{-1}	9.38_{-1}	9.37_{-1}	8.98_{-1}	9.02_{-1}	9.02_{-1}
		CNM	*time*	4.52	5.06	5.09	4.61	5.03	4.72	4.87	4.65	4.91	5.37	5.28	4.88	5.41	5.33	5.30
			NMI	7.73_{-1}	7.73_{-1}	6.75_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	LRBFGS IC1	Gap	*time*	3.74	3.76	3.76	7.24	7.18	6.83	1.05_1	1.03_1	1.03_1	4.39_1	4.38_1	4.38_1	8.09_1	8.06_1	8.10_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.24_{-1}	1.23_{-1}	1.27_{-1}	2.21_{-1}	2.17_{-1}	2.14_{-1}	3.24_{-1}	3.27_{-1}	3.19_{-1}	1.62	1.61	1.56	3.15	3.58	3.58
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.90_{-1}	9.90_{-1}	9.83_{-1}	9.96_{-1}	9.96_{-1}	9.94_{-1}	9.87_{-1}	9.87_{-1}	9.83_{-1}	9.36_{-1}	9.38_{-1}	9.37_{-1}	8.98_{-1}	9.02_{-1}	9.02_{-1}
		CNM	*time*	4.63	4.60	4.70	4.64	5.01	4.71	4.87	4.73	4.60	5.36	5.04	4.69	5.19	5.23	4.97
			NMI	7.73_{-1}	7.73_{-1}	6.75_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	LRBFGS IC2	Gap	*time*	3.71	3.71	3.74	7.10	6.99	6.91	1.02_1	1.03_1	1.02_1	4.33_1	4.36_1	4.35_1	7.97_1	8.05_1	8.10_1
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	8.06_{-1}	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.25_{-1}	1.22_{-1}	1.24_{-1}	2.16_{-1}	2.20_{-1}	2.15_{-1}	3.26_{-1}	3.25_{-1}	3.28_{-1}	1.62	1.62	1.59	3.13	3.18	3.09
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	9.90_{-1}	9.90_{-1}	9.83_{-1}	9.96_{-1}	9.96_{-1}	9.94_{-1}	9.88_{-1}	9.88_{-1}	9.83_{-1}	9.46_{-1}	9.48_{-1}	9.45_{-1}	9.14_{-1}	9.17_{-1}	9.16_{-1}
		CNM	*time*	4.96	4.84	4.63	4.86	5.03	4.61	4.97	4.73	4.55	5.11	4.91	4.76	5.36	5.08	4.88
			NMI	7.73_{-1}	7.73_{-1}	6.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Cheng LR	Gap	*time*		3.66			7.26			1.05_1			4.45_1			8.34_1	
			NMI		1.00			1.00			8.07_{-1}			1.00			1.00	
		Silhouette	*time*		1.22_{-1}			2.16_{-1}			3.28_{-1}			1.63			3.14	
			NMI		1.00			1.00			1.00			1.00			1.00	
			SC		9.86_{-1}			9.96_{-1}			9.86_{-1}			9.32_{-1}			8.92_{-1}	
		CNM	*time*		5.59			5.68			5.44			5.99			6.41	
			NMI		7.73_{-1}			1.00			1.00			1.00			1.00	

Table 6.6: Comparison of NMI and time (seconds) between k-means clustering and community detection with CNM for role graph (b). SC is the silhouette coefficient. The subscript ν indicates a scale of 10^{ν}.

(p_{in}, p_{out})	Algorithm		r	2			3			4			10			15		
				β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
$(0.9, 0.1)$	Browet LR	Gap	*time*	3.81	3.76	3.77	7.75	7.55	7.58	1.11_1	1.09_1	1.09_1	4.27_1	4.26_1	4.29_1	8.08_1	8.08_1	8.05_1
			NMI	7.97_{-1}	7.73_{-1}	9.21_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.27_{-1}	1.29_{-1}	1.24_{-1}	2.24_{-1}	2.13_{-1}	2.16_{-1}	3.19_{-1}	3.26_{-1}	3.27_{-1}	1.56	1.61	1.59	3.12	3.10	3.55
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.97_{-1}	9.98_{-1}	9.98_{-1}	9.88_{-1}	9.91_{-1}	9.93_{-1}	9.80_{-1}	9.85_{-1}	9.88_{-1}
		CNM	*time*	4.71	4.63	4.53	4.67	4.61	4.56	4.75	4.82	4.57	4.98	5.50	4.64	5.01	5.33	5.07
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	LRBFGS IC1	Gap	*time*	3.77	3.73	3.75	7.39	7.78	7.57	1.11_1	1.10_1	1.09_1	4.30_1	4.30_1	4.23_1	8.07_1	8.06_1	8.07_1
			NMI	7.97_{-1}	7.73_{-1}	9.21_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.27_{-1}	1.29_{-1}	1.26_{-1}	2.17_{-1}	2.15_{-1}	2.14_{-1}	3.24_{-1}	3.32_{-1}	3.21_{-1}	1.60	1.58	1.59	3.09	3.10	3.56
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.97_{-1}	9.98_{-1}	9.98_{-1}	9.88_{-1}	9.91_{-1}	9.93_{-1}	9.80_{-1}	9.85_{-1}	9.88_{-1}
		CNM	*time*	4.74	4.66	4.56	4.71	4.66	4.55	4.99	4.98	4.64	4.88	4.67	4.63	5.74	5.13	5.04
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	LRBFGS IC2	Gap	*time*	3.71	3.75	3.69	7.67	7.69	7.55	1.11_1	1.09_1	1.08_1	4.27_1	4.26_1	4.24_1	8.17_1	8.12_1	8.18_1
			NMI	7.97_{-1}	7.73_{-1}	9.21_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.26_{-1}	1.30_{-1}	1.26_{-1}	2.24_{-1}	2.13_{-1}	2.15_{-1}	3.23_{-1}	3.25_{-1}	3.23_{-1}	1.61	1.62	1.61	3.17	3.12	3.10
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			SC	1.00	1.00	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.99_{-1}	9.98_{-1}	9.98_{-1}	9.98_{-1}	9.90_{-1}	9.92_{-1}	9.94_{-1}	9.83_{-1}	9.88_{-1}	9.90_{-1}
		CNM	*time*	5.54	4.82	4.59	5.23	4.77	4.62	6.13	4.90	4.55	4.74	4.54	4.96	4.85	4.61	4.72
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	Cheng LR	Gap	*time*		3.81			7.23			1.11_1			5.01_1			9.24_1	
			NMI		1.00			9.26_{-1}			8.89_{-1}			8.07_{-1}			7.74_{-1}	
		Silhouette	*time*		1.22_{-1}			2.13_{-1}			3.20_{-1}			1.50			2.87	
			NMI		7.73_{-1}			1.00			1.00			1.00			7.73_{-1}	
			SC		9.94_{-1}			9.98_{-1}			9.86_{-1}			9.26_{-1}			9.06_{-1}	
		CNM	*time*		5.74			5.70			6.01			6.19			6.27	
			NMI		7.73_{-1}			7.73_{-1}			7.73_{-1}			7.73_{-1}			7.73_{-1}	
$(0.7, 0.3)$	Browet LR	Gap	*time*	3.68	3.76	3.68	7.61	7.23	7.46	1.11_1	1.06_1	1.11_1	4.44_1	4.40_1	4.33_1	8.43_1	8.05_1	7.93_1
			NMI	8.97_{-1}	7.73_{-1}	9.81_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.25_{-1}	1.29_{-1}	1.25_{-1}	2.15_{-1}	2.16_{-1}	2.17_{-1}	3.20_{-1}	3.26_{-1}	3.26_{-1}	1.69	1.61	1.60	3.61	3.12	3.15
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	7.73_{-1}	1.00	1.00
			SC	9.95_{-1}	9.98_{-1}	9.79_{-1}	9.91_{-1}	9.90_{-1}	9.87_{-1}	9.71_{-1}	9.76_{-1}	9.78_{-1}	8.79_{-1}	9.07_{-1}	9.33_{-1}	8.38_{-1}	8.59_{-1}	9.00_{-1}
		CNM	*time*	4.63	4.80	5.03	4.66	4.81	4.83	4.56	4.85	4.94	5.07	4.92	5.13	5.21	5.29	5.40
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	LRBFGS IC1	Gap	*time*	3.80	3.71	3.74	7.59	7.22	7.53	1.13_1	1.05_1	1.08_1	4.48_1	4.40_1	4.28_1	8.40_1	8.04_1	8.01_1
			NMI	8.97_{-1}	7.73_{-1}	9.81_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.26_{-1}	1.32_{-1}	1.25_{-1}	2.18_{-1}	2.15_{-1}	2.16_{-1}	3.21_{-1}	3.37_{-1}	3.32_{-1}	1.62	1.62	1.62	3.13	3.11	3.56
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	7.73_{-1}	1.00	1.00
			SC	9.95_{-1}	9.98_{-1}	9.79_{-1}	9.91_{-1}	9.90_{-1}	9.87_{-1}	9.71_{-1}	9.76_{-1}	9.78_{-1}	8.79_{-1}	9.07_{-1}	9.33_{-1}	8.38_{-1}	8.59_{-1}	9.00_{-1}
		CNM	*time*	4.61	5.04	4.66	4.59	4.63	4.60	4.78	4.93	4.60	5.00	4.89	5.15	5.15	5.51	5.20
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	LRBFGS IC2	Gap	*time*	3.76	3.84	3.73	7.71	7.29	7.59	1.11_1	1.06_1	1.09_1	4.46_1	4.35_1	4.28_1	8.12_1	8.04_1	7.98_1
			NMI	8.97_{-1}	7.73_{-1}	9.81_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		Silhouette	*time*	1.26_{-1}	1.26_{-1}	1.24_{-1}	2.17_{-1}	2.17_{-1}	2.15_{-1}	3.21_{-1}	3.31_{-1}	3.22_{-1}	1.60	1.61	1.58	3.12	3.19	3.66
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	7.73_{-1}	1.00	1.00
			SC	9.95_{-1}	9.98_{-1}	9.79_{-1}	9.90_{-1}	9.90_{-1}	9.87_{-1}	9.75_{-1}	9.79_{-1}	9.80_{-1}	8.96_{-1}	9.19_{-1}	9.42_{-1}	8.49_{-1}	8.78_{-1}	9.13_{-1}
		CNM	*time*	4.85	4.56	4.59	4.63	4.83	4.83	5.38	4.84	4.67	5.19	4.90	4.62	5.13	5.31	4.94
			NMI	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}	7.73_{-1}
	Cheng LR	Gap	*time*		3.73			7.15			1.18_1			5.32_1			9.96_1	
			NMI		1.00			1.00			8.47_{-1}			9.26_{-1}			1.00	
		Silhouette	*time*		1.22_{-1}			2.16_{-1}			3.26_{-1}			1.61			3.54	
			NMI		7.73_{-1}			1.00			1.00			7.73_{-1}			7.73_{-1}	
			SC		9.82_{-1}			9.87_{-1}			9.48_{-1}			8.70_{-1}			8.32_{-1}	
		CNM	*time*		5.91			5.91			6.53			6.42			5.93	
			NMI		7.73_{-1}			7.73_{-1}			7.73_{-1}			7.73_{-1}			7.73_{-1}	

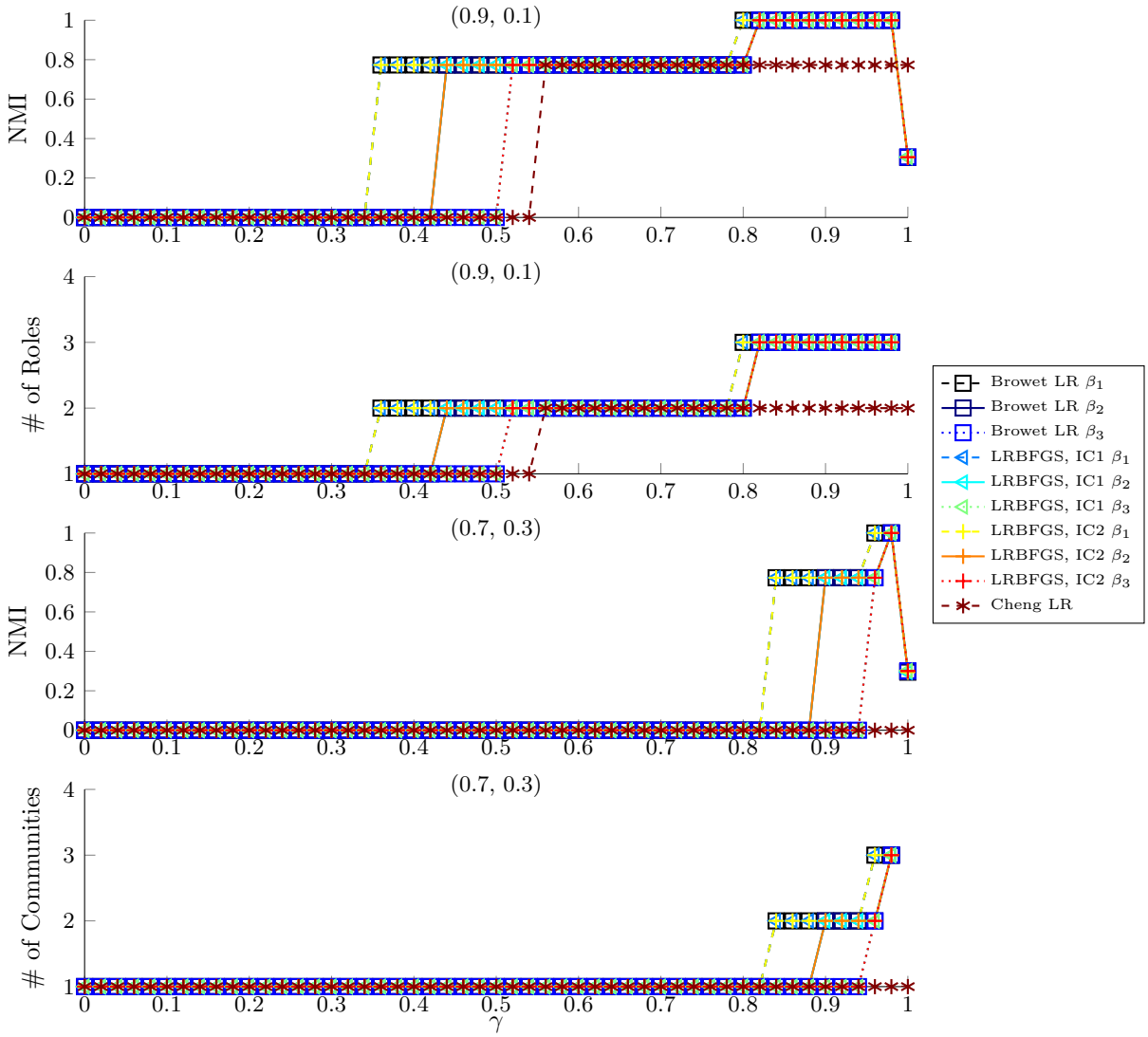


Figure 6.6: NMI and number of roles as resolution parameter γ varies in $[0, 1]$ for role graph (b).

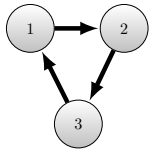
Community detection with CNM is able to extract the role structure of graph (a) when r is greater than or equal to the number of roles. However, for role graph (b), the NMI is 0.773 for $r \geq 2$ (see Table 6.6). This difficulty to extract the role structure is possibly due to either a resolution limit phenomenon or to the almost isomorphic behavior of the two roles. Thus, we consider CPM with community detection for graph (b) and vary the resolution parameter γ in $[0, 1]$ by 0.02 to determine the γ that extracts the role structure. Figure 6.6 displays the NMI and the number of roles for $(p_{in},\ p_{out}) = (0.9, 0.1)$ and $(0.7, 0.3)$. For the low-rank algorithms, we display the results when $r = 3$. Note that when $(p_{in}, p_{out}) = (0.9, 0.1)$, CPM is able to extract the role structure

from the neighborhood pattern similarity measure when γ is in the interval $[0.8, 0.98]$ for β_1 and $[0.82, 0.98]$ for β_2 and β_3. Unfortunately, note that that the length of this interval is less than the length of the interval where CPM extracts a 2 role structure (which was $[0.36, 0.78]$ for β_1, $[0.44, 0.8]$ for β_2, $[0.52, 0.8]$). Thus, the 2 role partition is the more stable (or natural) partition than the 3 role partition. Observe that as we consider longer neighborhood patterns, the plateau that finds a 2 role structure decreases in length, while the plateau corresponding to the 3 role structure does not. So the longer neighborhood patterns help determine the 3 role structure of the network. However, as p_{in} decreases and p_{out} increases, the interval of resolution parameter values that extract the correct role structure decreases and it is impossible to extract the roles. This is different from the silhouette statistic, which is able to extract the roles from noisy graphs when using longer neighborhood patterns.

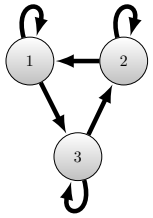
While CPM on the low-rank neighborhood pattern similarity measure is able to determine the role structure, CPM on the low-rank similarity measure of Cheng et al. is only able to extract a 2 role partition for $(p_{in}, p_{out}) = (0.9, 0.1)$ and is unable to extract the role partition for $(p_{in}, p_{out}) = (0.7, 0.3)$. Therefore, CPM is not a good community detection cost function for Cheng et al.'s low-rank similarity measure.

6.1.5 Robustness Comparison

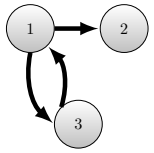
Finally, we compare the robustness of k-means clustering with the silhouette statistic and Browet et al.'s community detection algorithm with CNM as the cost function for extracting the roles from the low-rank similarity approximations. The neighborhood pattern similarity algorithms we compare are Browet and Van Dooren's low-rank iterative algorithm and LRBFGS starting from both IC1 and IC2. We compare the neighborhood patterns similarity algorithms with Cheng et al.'s low-rank similarity measure and assume that the rank is fixed at $r = 10$. For each reduced graph, we generate 20 random realizations for each pair of probability parameters p_{in} and p_{out} in the interval $[0, 1]$ for step size 0.05 and compute the average NMI between the true role partition and the extracted role partition. For k-means clustering with the silhouette statistic, we assume that the number of clusters is from 2 to r. We implement both algorithms for three different β values given by (6.11). Note that the low-rank similarity measure of Cheng et al. does not use the parameter β.



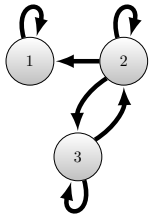
(a) Block Cycle Role Graph



(b) Complement Cycle Role Graph



(c) Almost Isomorphic Role Graph



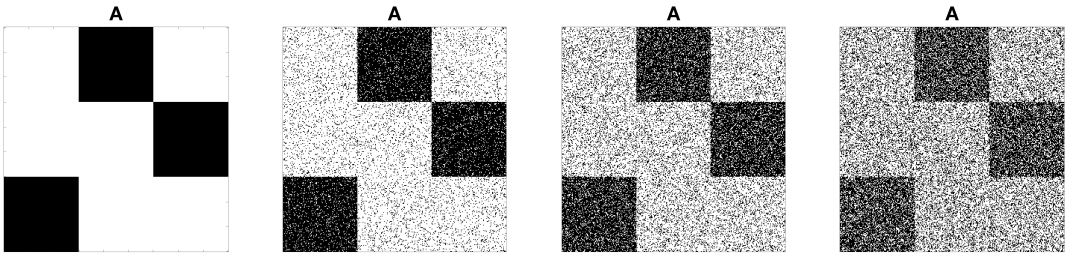
(d) Complement Almost Isomorphic Role Graph

Figure 6.7: Complement Erdós-Rényi role graphs.

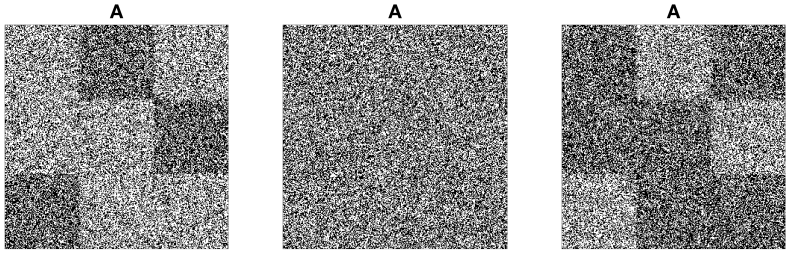
For our first example, we consider homogeneous and nonhomogeneous role sizes for role graph (a). For the homogeneous role size case, there are 100 nodes in each role, and for the nonhomogeneous role sizes, there are 320 nodes in the first role, 80 in the second role, and 20 in the third role. Note that for $p_{in} \gg p_{out}$, we have role graph (a), but for $p_{in} \ll p_{out}$, we have the complement role graph (see Figure 6.7). The complement role graph has a cycle in the opposite direction and every role has a self-loop. While the role structure in the complement role graph is different than role graph (a) in Figure 6.1, the nodes in the roles are partitioned the same, so the role extraction algorithms should extract similar role partitions.

Figures 6.8 and 6.9 display the adjacency matrices of the homogeneous and nonhomogeneous role size cases, respectfully, as (p_{in}, p_{out}) goes from the noiseless case $(1, 0)$ of role graph (a) to the noiseless case of the complement role graph $(0, 1)$. Note that for the homogeneous role sizes, the adjacency matrix has a distinct role structure for role graph (a) and its complement for the noiseless case. As we add noise to the graphs, the role structures are still distinct until $p_{in} \approx p_{out}$. Thus, we expect the role extraction algorithms to extract the exact role partitions until $p_{in} \approx p_{out}$.

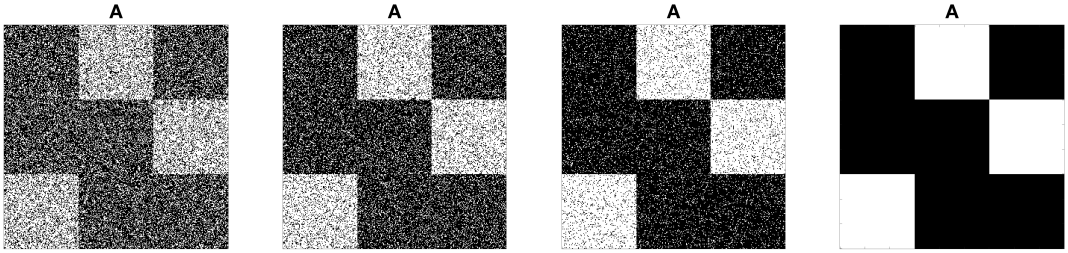
For the nonhomogeneous role sizes, the role structures are noticeable for the noiseless case, but become indistinguishable as noise is added to the graphs. In particular, role **3** in both graphs is difficult to distinguish from the other roles since there are 20 nodes in the role. Therefore, adding



(a) $(1, 0)$ (b) $(0.9, 0.1)$ (c) $(0.8, 0.2)$ (d) $(0.7, 0.3)$



(e) $(0.6, 0.4)$ (f) $(0.5, 0.5)$ (g) $(0.4, 0.6)$

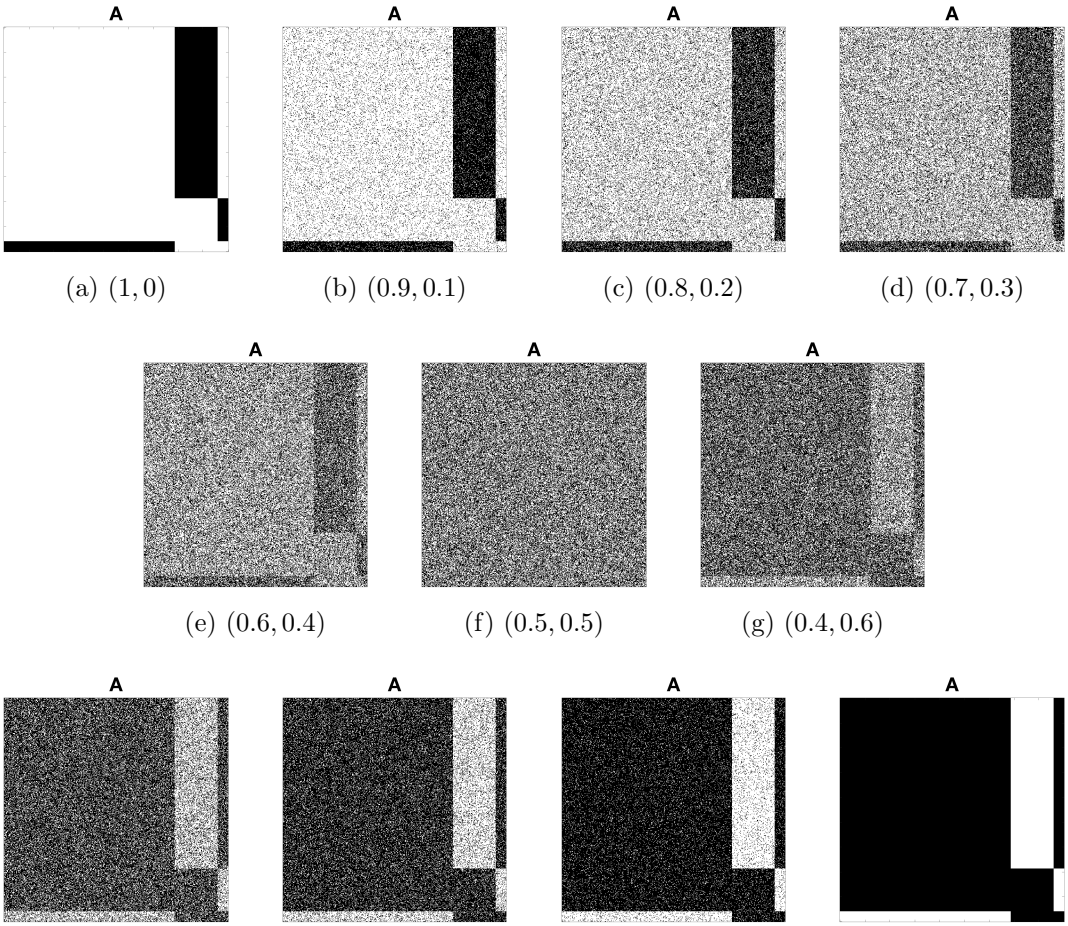


(h) $(0.3, 0.7)$ (i) $(0.2, 0.8)$ (j) $(0.1, 0.9)$ (k) $(0, 1)$

Figure 6.8: Adjacency matrices of the Erdós-Rényi graphs with role graph (a) and homogeneous role sizes, where (p_{in}, p_{out}) goes from $(1, 0)$ to $(0, 1)$.

(removing) edges between role **3** and roles **1** and **2** and, for the complement role graph, adding (removing) edges between the nodes within the role will change the node distribution between the roles and change the role partition. Thus, we expect the role extraction algorithms to struggle when extracting the exact role partitions for this example because adding (removing) edges from small roles changes the role structure greater than adding (removing) edges from larger roles.

Figures 6.10 and 6.11 display the average NMI values for role graph (a) with homogeneous (Figure 6.10) and nonhomogeneous (Figure 6.11) role sizes. For the neighborhood pattern similarity measure, k-means clustering with the sillhouette statistic (rows (1), (3), and (5) in Figure 6.10) and community detection with CNM (rows (2), (4), and (6) in Figure 6.10) have similar results



(a) $(1, 0)$ (b) $(0.9, 0.1)$ (c) $(0.8, 0.2)$ (d) $(0.7, 0.3)$

(e) $(0.6, 0.4)$ (f) $(0.5, 0.5)$ (g) $(0.4, 0.6)$

(h) $(0.3, 0.7)$ (i) $(0.2, 0.8)$ (j) $(0.1, 0.9)$ (k) $(0, 1)$

Figure 6.9: Adjacency matrices of the Erdós-Rényi graphs with role graph (a) and nonhomogeneous role sizes, where (p_{in}, p_{out}) goes from $(1, 0)$ to $(0, 1)$.

for homogeneous role sizes. Both algorithms are able to extract the role partition exactly when $p_{in} \gg p_{out}$ (role graph (a)) and $p_{in} \ll p_{out}$ (complement role graph of role graph (a)), and fail to extract the role partition when $p_{in} \approx p_{out}$. Also, there was little change between β_1, β_2 and β_3, except for LRBFGS when $\beta = \beta_3$ and $p_{in} \approx p_{out}$. Both the silhouette statistic and CNM extracted different role partitions that were still close to the original role partition. A possible explanation is that the optimization algorithm may have found a different local minimizer of the low-rank neighborhood pattern similarity matrix when $\beta = \beta_3$ compared to the low-rank similarity matrix found by Browet and Van Dooren's algorithm. Thus, the clustering algorithms extracted a different, but close, role partition.

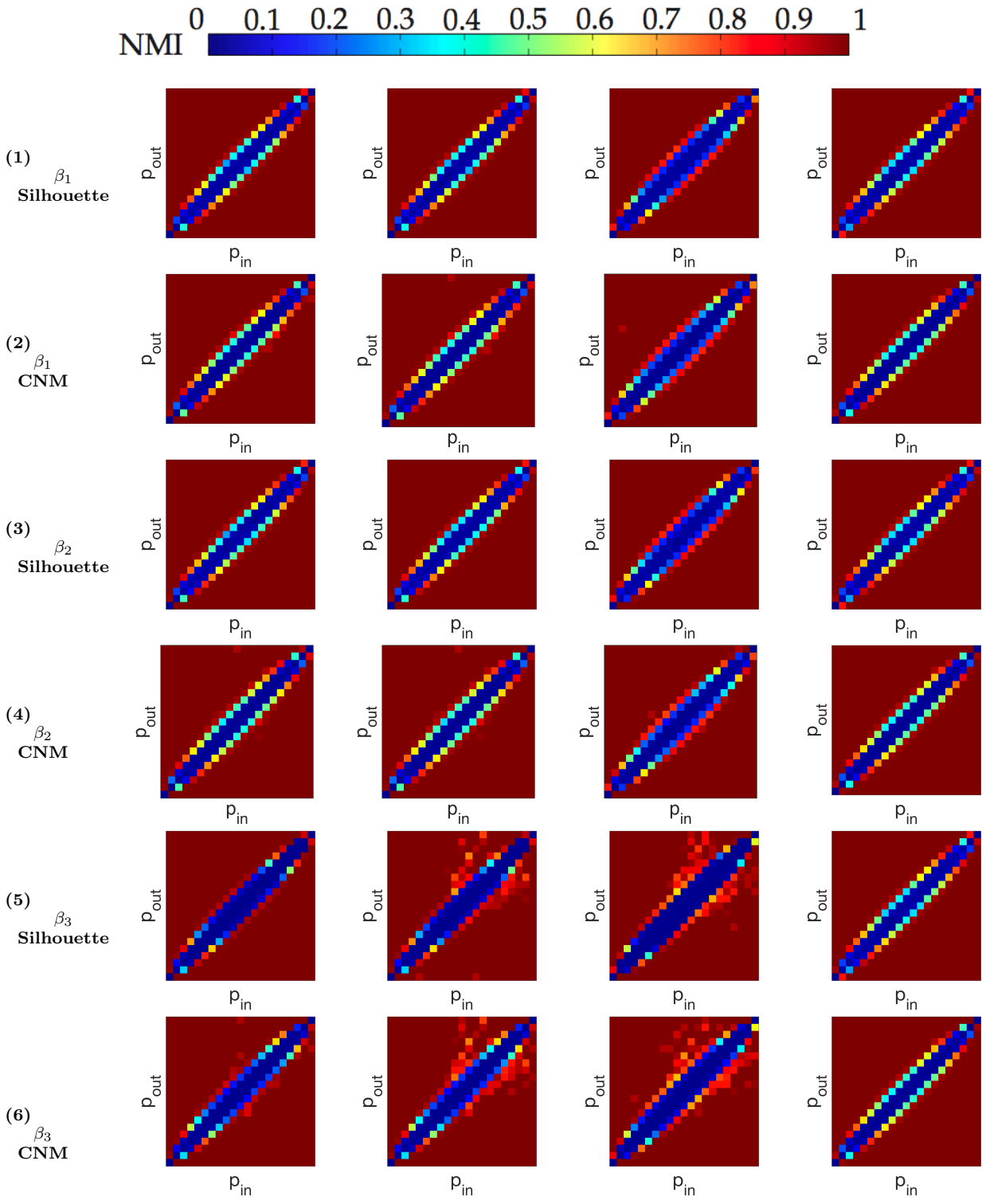
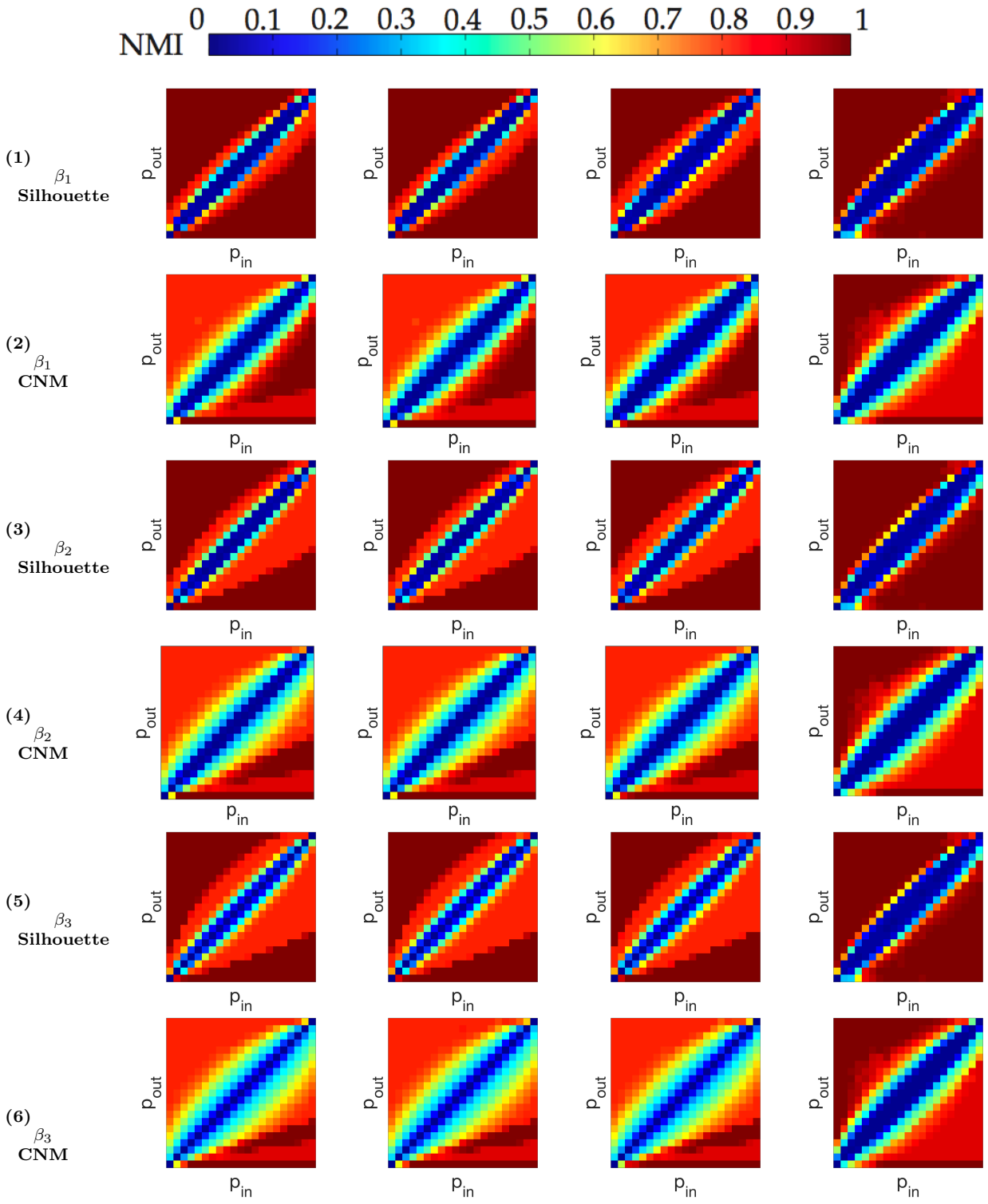


Figure 6.10: NMI for k-means clustering with silhouette statistic and community detection algorithm with CNM for role graph (a) and homogeneous role sizes.



(a) Browet LR (b) LRBFGS, IC1 (c) LRBFGS, IC2 (d) Cheng LR

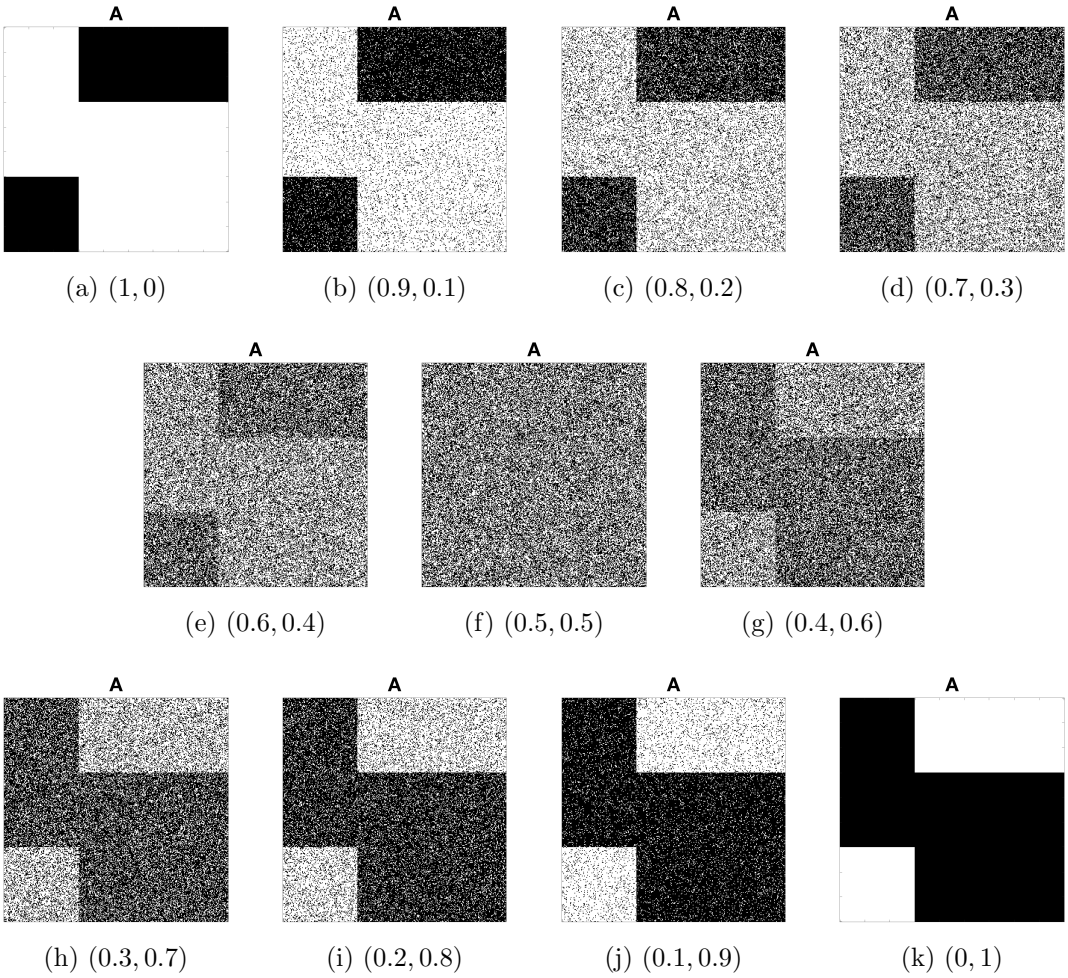
Figure 6.11: NMI for k-means clustering with silhouette statistic and community detection algorithm with CNM for role graph (a) and nonhomogeneous role sizes.

When the role sizes are different (Figure 6.11), k-means clustering with the silhouette statistic was able to extract the role structure from the neighborhood pattern similarity approximation for $p_{in} \gg p_{out}$ (and vice versa) for β_1. For β_2 and β_3, the algorithm began to struggle to extract the role partition when $p_{in} \gg p_{out}$, but did not struggle for its complement role graph. The asymmetry in the NMI values for β_2 and β_3 is due to how the noise was distributed for role graph (a), where the longer neighborhood patterns detected a different role partition for the noisy graphs compared to the exact role partition. However, the noise in the complement role graph did not affect the silhouette statistic from extracting the exact role partition. The average NMI values for community detection with CNM was the opposite. CNM fails to extract the exact role partition when $p_{in} \ll p_{out}$ for all three β values. For some p_{in} and p_{out} parameters when $p_{in} \gg p_{out}$, CNM also fails to extract the exact role partition. This is probably due to the resolution limit of CNM. Overall, the NMI values for the silhouette is higher than CNM.

For the low-rank similarity measure of Cheng et al., when $p_{in} \approx p_{out}$, both role extraction algorithms fail to extract the role structure sooner, i.e., the blue band in the middle is wider, than the neighborhood pattern similarity measure. The reason for the higher NMI results is that the neighborhood pattern similarity measure gains extra information about the role structure from the longer neighborhood patterns while Cheng et al.'s similarity measure only considers the immediate neighbors of a node. Thus, for empirical data, the neighborhood pattern similarity measure may be preferable because it is more robust in the presence of noise and nonhomogeneous roles sizes.

For our second example, we consider homogeneous role sizes for role graph (b), where there are 100 nodes in each role. Similar to role graph (a), when $p_{in} \gg p_{out}$, the Erdós-Rényi graphs have the role structure given by role graph (b), but when $p_{in} \ll p_{out}$, they have the complement role graph structure (see Figure 6.7). However, the roles are partitioned the same, so the role extraction algorithms should extract similar role partitions.

Figure 6.12 displays the adjacency matrices as (p_{in}, p_{out}) transitions from the noiseless case $(1, 0)$ of role graph (b) to the noiseless case of the complement role graph $(0, 1)$. For the noiseless cases, the rank of the adjacency matrix is 2; however, the rank of the neighborhood pattern similarity matrix is 3, which is the number of roles. Therefore, the role extraction algorithms should extract the role partitions from the similarity matrix for both noiseless cases. As edges are added (or removed) from the graphs, it becomes difficult to distinguish between the 3 role structure. Also,



(a) $(1, 0)$ (b) $(0.9, 0.1)$ (c) $(0.8, 0.2)$ (d) $(0.7, 0.3)$

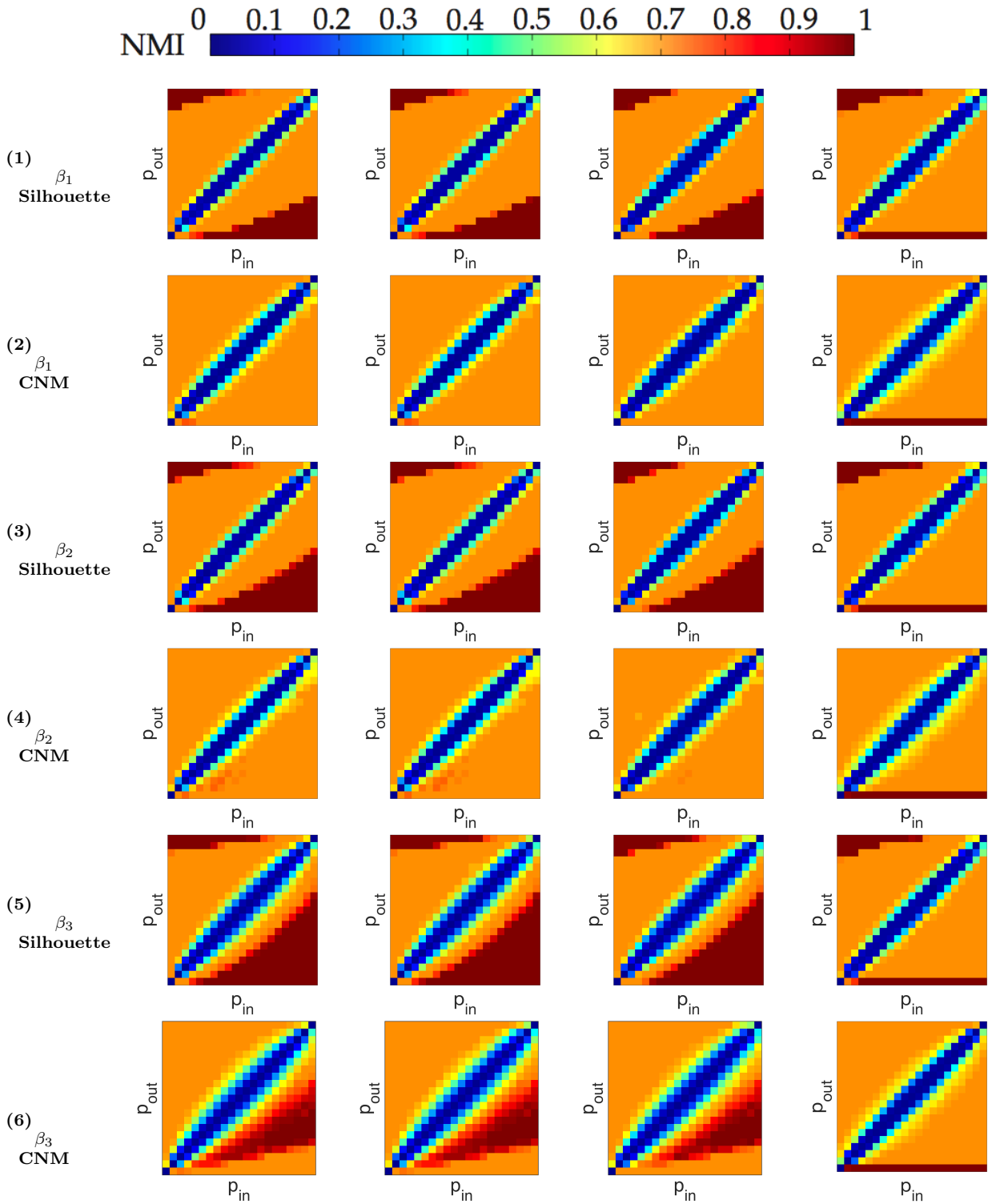
(e) $(0.6, 0.4)$ (f) $(0.5, 0.5)$ (g) $(0.4, 0.6)$

(h) $(0.3, 0.7)$ (i) $(0.2, 0.8)$ (j) $(0.1, 0.9)$ (k) $(0, 1)$

Figure 6.12: Adjacency matrices of the Erdós-Rényi graphs with role graph (b), where (p_{in}, p_{out}) goes from $(1, 0)$ to $(0, 1)$.

note that the noise is not symmetric between role graph (b) and its complement. That is, adding edges to null blocks in the complement role graph distorts the role graph sooner than adding edges to the null blocks in role graph (b). Therefore, the role extraction algorithms may extract the exact role partitions more often for role graph (b) than its complement.

Figure 6.13 displays the NMI for the Erdós-Rényi graphs with almost isomorphic role structure where the roles are assumed to all be the same size. Comparing the silhouette statistic (rows (1), (3), and (5)) with CNM (rows (2), (4), and (6)), we see that the silhouette statistic extracts the role partition better for the neighborhood pattern similarity measure when $p_{in} \gg p_{out}$ and $p_{out} \ll p_{in}$, while the NMI values for CNM is around 0.773 for most p_{in} and p_{out} combinations. As



(a) Browet LR (b) LRBFGS, IC1 (c) LRBFGS, IC2 (d) Cheng LR

Figure 6.13: NMI for k-means clustering with the silhouette statistic and community detection algorithm with CNM for role graph (b).

the neighborhood pattern similarity measure searched longer neighborhood patterns, the silhouette statistic was able to determine the role structure for the role graph, but not its complement. The asymmetry of the NMI values is expected since the noise distribution for the graph is not symmetric. Community detection was only able to extract the role structure from a small subset of probability pairs for β_3. This is probably due to the fact that β_3 allows the similarity algorithms to search longer neighborhood patterns for similarity between the nodes, which provided the extract information CNM needed to extract the role partition.

Compared with the previous role structure example, both clustering methods fail to extract the role partition more often. This is probably due to the almost isomorphic behavior of two of the roles. For the low-rank similarity measure of Cheng et al., the NMI for k-means clustering with the silhouette statistic and CNM are also around 0.773. However, the silhouette statistic extracts the role partition for more p_{in} and p_{out} pairs from the neighborhood pattern similarity measure than from Cheng et al.'s low-rank similarity measure. Thus, for more complicated role graphs, the indirect approach using the neighborhood pattern similarity measure and k-means clustering is a better approach to extract the role structure.

6.1.6 Summary

Overall, for the role extraction problem, the neighborhood pattern similarity measure is preferred over the similarity measure of Cheng et al. because the longer neighborhood patterns are significant for examples where there are two roles that mainly interact with each other or there is a difference in role sizes, both of which are common in real world applications.

Comparing the algorithms to compute the neighborhood pattern similarity measure, the low-rank algorithm by Browet and Van Dooren is competitive in time with Riemannian optimization algorithms when the parameter β is small. However, a small β means we are not looking at long neighborhood patterns in the network. In practice, we do not know the role structure is and observing longer neighborhood patterns helps determine an accurate role structure. When β is large (but still satisfies the existence bounds), LRBFGS is significantly faster than Browet and Van Dooren's low-rank algorithm and RSD for all r. When r is less than or equal to the number of roles, RNewton and LRBFGS are competitive in time, but when r is larger than the number of roles, LRBFGS is faster than RNewton. Therefore, LRBFGS is preferable to compute a low-rank

approximation of the similarity matrix for networks with a large number of roles because we would assume that r is also large.

When comparing the clustering methods for the second part of the role extraction process, k-means clustering with the silhouette statistic and community detection with the CNM have similar NMI results for role graph (a), but different NMI results for role graph (b). The silhouette statistic extracts the exact role partition more often than community detection with CNM as p_{in} and p_{out} vary. However, the silhouette statistic fails to extract the exact role partition when the rank is assumed to be much larger than the number of roles for $\beta = \beta_1$. In practice, we do not know the number of roles necessary to represent the network. Therefore, we should consider longer neighborhood pattern for the silhouette statistic in case we have overestimated the rank with respect to the number of roles.

Also, k-means clustering with the silhouette statistic requires an assumed index list of number of roles to compute k-means clustering with and choosing the optimal number of clusters by maximizing the silhouette value over the data set, which can be costly when the index list is not chosen carefully. Community detection does not require the user to assume the number of roles; however, it requires the computation of the similarity matrix $S^{(r)} = XX^T$, which is $O(n^2 r)$ computations and is costly when either n or r are large. So, both clustering methods suffer for large n and r and a more sophisticated approach is needed to extract the role structure.

Fortunately, there is a relationship between the smallest rank necessary to approximate the similarity measure and the number of roles in the graph. If the role blocks are complete, then the rank of the neighborhood pattern similarity measure is equal to the number of roles (see Theorem 3.0.5). If the blocks are regular, then the rank is not equal to the number of roles, but there is still a noticeable gap between the singular values (see Figure 3.4b). Thus, in Chapter 7, we combine the current methods for determining the low-rank approximation of a matrix using Riemannian optimization (see [JBAS10, MS13, Zho15, ZHG$^+$16, HGZ16]) to develop a Riemannian optimization algorithm that determine the numerical rank of the neighborhood pattern similarity measure and optimal number of roles using k-means clustering with the silhouette statistic simultaneously, i.e., mixing the functions of the previously defined two phase into a single phase. The list of assumed clusters to test for the silhouette statistic is determined by the rank and assumed number of clus-

ters and is adjusted until convergence. This approach combines the two-step process of the indirect approach into a single, low-rank iterative process.

6.2 Metal World Trade Network

Next, we analyze a real world example. We classify countries based on 1994 trade data from several manufactures of metal[1]. This dataset is from [NMB11, Chapter 2], where Nooy et al. used it to analyze global relationships between countries in trade systems. The authors use the capitalist world trade concept first introduced by Wallerstein in [Wal74], where the world economy is classified into three categories: core, semiperiphery, and periphery. The core consists of countries that profit by successfully exploiting the periphery and the sempheriphery, while the semiperiphery consists of countries that profit by being a link between the core and the periphery [Chapter 2] [NMB11]. Also, core countries specialize in capital-intensive and high-tech production, while peripheral countries specialize in low-valued, labor-intensive products or raw materials. Thus, core countries import raw materials from periphery countries, transform them into high-tech products, then export them to core, semiperiphery, and periphery countries [NMB11, Chapter 2]. For the metal trade network, we expect the network to be dominated by trade from the core countries and see very little trade by periphery countries.

The metal trade dataset contains all countries with entries in the paper version of the Commodity Trade Statistics published by the United Nations, except for countries Austria, Seychelles, Bangladesh, Croatia, and Barbados, which used the 1993 statistics, and South Africa and Ecuador, which used the 1995 statistics, because the information for 1994 was unavailable [Nat,Nat94,BM06, NMB11]. Countries that are not sovereign are excluded because additional economic data is not available, e.g., Faeroe Islands and Greenland belong to Denmark and Macau belongs to Portugal. Note that most of the missing countries are located in central Africa and the Middle East, or belonged to the former Union of Soviet Socialist Republics (USSR).

In the dataset, a directed edge represents imports by one country (source) from another (target) for the class of commodities classified as "miscellaneous manufactures of metal", which represents high technology products or heavy manufactures. The absolute value of imports (in $1,000$ US \$) is used; but imports with values less than 1% of the country's total imports are omitted from the

[1] `http://vlado.fmf.uni-lj.si/pub/networks/data/esna/metalWT.htm`

Table 6.7: Countries partitioned by their world system positions in 1994.

World Position	Countries
Core	Austria, Belgium /Lux., China, France Mon., Germany, Italy, Japan, Netherlands, Sweden, Switzerland, United Kingdom, United States
Semiperiphery	Algeria, Argentina, Australia, Barbados, Brazil, Canada, Chile, Colombia, Croatia, Cyprus, Czech Rep., Denmark, Ecuador, Egypt, Finland, Greece, Hong Kong, Hungary, Iceland, India, Indonesia, Ireland, Israel, Jordan, Rep. of Korea, Kuwait, Latvia, Madagascar, Malaysia, Mauritius, Morocco, New Zealand, Norway, Oman, Pakistan, Peru, Philippines, Poland, Portugal, Romania, Singapore, Slovenia, Southern Africa, Spain, Sri Lanka, Thailand, Trinidad and Tobago, Tunisia, Turkey, Uruguay, Venezuela
Periphery	Bangladesh, Belize, Bolivia, El Salvador, Fiji, French Guiana, Guadeloupe, Guatemala, Honduras, Martinique, Mexico, Nicaragua, Panama, Paraguay, Rep. of Moldova, Reunion, Seychelles

dataset. The only preprocessing applied to the data is to take the logarithm of the imports, which preserves the relative strength of the imports, but prevents countries with large trade values from dominating the quality of the role model. Table 6.7 partitions the countries by their world position in 1994 [Nat, Nat94, BM06, NMB11].

For our experiments, we compare the low-rank similarity algorithms with $r = 10$. For the Riemannian optimization methods, vector transport is by parallelization, and k-means clustering with the silhouette statistic extracts the role partition. The cluster list for the silhouette statistic was $\{2, \ldots, 11\}$. We compare our role extraction approach with the approach by Browet and Van Dooren, which uses their low-rank algorithm to approximate the similarity matrix and community detection with CPM to extract the role partition. For both approaches that use the neighborhood pattern similarity measures, we compare results for the three β given by (6.11).

Lastly, we compare the extracted role partition from the low-rank neighborhood pattern similarity algorithms with the low-rank similarity measure of Cheng et al.. For Cheng et al.'s similarity measure, we use both k-means clustering with the silhouette statistic and community detection with CPM to extract the role partition.

For CPM, we vary the resolution parameter γ in the interval $[0, 1]$ incrementally by 0.02 and look for the longest plateau where the number of roles does not vary. In Figure 6.14 for β_1, notice that longest plateau for the low-rank neighborhood pattern similarity measures is for the interval

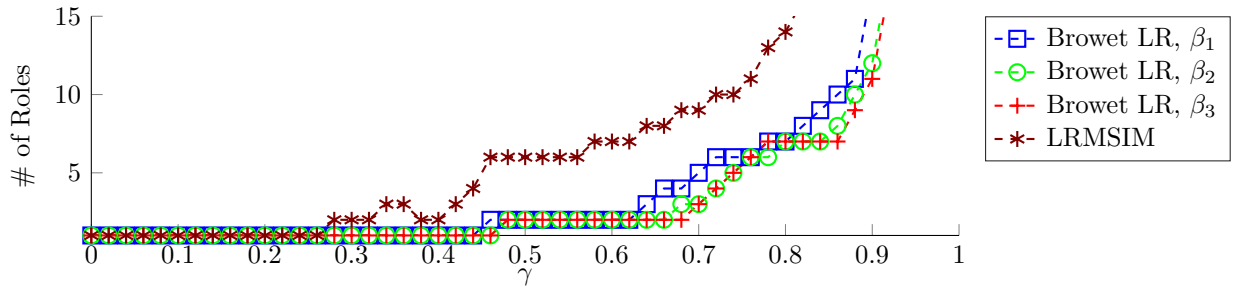


Figure 6.14: Number of roles as resolution parameter γ varies in $[0, 1]$.

Table 6.8: Countries partitioned by CPM.

Algorithm	Countries
Browet LR (all β) $\gamma = 0.56$	Algeria, Egypt, Madagascar, Mauritius, Morocco, Reunion, Seychelles, Southern Africa, Tunisia, Bangladesh, **INDIA**, Indonesia, Israel, Jordan, Kuwait, Malaysia, Oman, Pakistan, Philippines, Singapore, Sri Lankauy, Thailand, Croatia, Cyprus, Czech Rep., **DENMARK**, Finland, Greece, Hungary, Iceland, Ireland, Latvia, Norway, Poland, Portugal, Rep. of Moldova, Romania, Slovenia, **SPAIN**, Turkey, Belize, Canada, El Salvador, Guatemala, Honduras, Mexico, Nicaragua, Panama, Australia, Fiji, New Zealand, Argentina, Barbados, Bolivia, Brazil, Chile, Colombia, Ecuador, French Guiana, Guadeloupe, Martinique, Paraguay, Peru, Trinidad and Tobago, Uruguay, Venezuela
	China, Hong Kong, Japan, Rep. Of Korea, Austria, Belgium /Lux., France Mon., Germany, Italy, Netherlands, Sweden, Switzerland, United Kingdom, United States
LRMSIM $\gamma = 0.52$	Rep. of Moldova
	Hong Kong, Indonesia, Malaysia, Singapore, Thailand, Australia, New Zealand
	Canada, El Salvador, Guatemala, Honduras, Mexico, Panama, Argentina, Brazil, Chile, Colombia, Ecuador, Peru, Venezuela
	Algeria, Madagascar, Mauritius, Morocco, Reunion, Seychelles, Bangladesh, China, Japan, Jordan, Kuwait, Oman, Pakistan, Philippines, Sri Lanka, Cyprus, France Mon., Germany, Iceland, Italy, Latvia, United Kingdom, Belize, Nicaragua, United States, Fiji, French Guiana, Guadeloupe, Martinique, Paraguay, Uruguay
	Barbados, Bolivia, Trinidad and Tobago
	Egypt, Southern Africa, Tunisia, India, Israel, Rep. Of Korea, Austria, Belgium /Lux., Croatia, Czech Rep., Denmark, Finland, Greece, Hungary, Ireland, Netherlands, Norway, Poland, Portugal, Romania, Slovenia, Spain, Sweden, Switzerland, Turkey

Table 6.9: Number of roles and silhouette coefficient (SC). The asterisk and plus indicates that the role structure was different than the others. The subscript ν indicates a scale of 10^{ν}.

Algorithm	Silhouette					
	# Roles			SC		
	β_1	β_2	β_3	β_1	β_2	β_3
RSD, IC1	5	2	2	5.58_{-1}	5.99_{-1}	6.11_{-1}
RNewton, IC1	5	2	2	5.58_{-1}	5.99_{-1}	6.11_{-1}
LRBGS, IC1	5	2	2	5.58_{-1}	5.99_{-1}	6.11_{-1}
RSD, IC2	5	2	2	5.43_{-1}	5.95_{-1}	6.10_{-1}
RNewton, IC2	2	2	2	5.41_{-1}	5.95_{-1}	6.10_{-1}
LRBFGS, IC2	2	2	2	5.41_{-1}	5.95_{-1}	6.10_{-1}
Cheng LR	8			6.70_{-1}		

$[0.46, 0.62]$ and the number of roles for these resolution parameters is 2. For β_2 and β_3, the plateau is slightly longer, indicating that the plateau is more stable; however, the number of roles is still 2. For the similarity measure of Cheng et al., the longest plateau is $[0.46, 0.56]$ with 6 roles. Table 6.8 displays the partitions of the countries extracted from the similarity measures using community detection.

For k-means clustering with the silhouette statistic, the ideal number of clusters varies for the Riemannian algorithms and the parameter β. When $\beta = \beta_1$, the silhouette statistic extracts either 2 or 5 roles (see Table 6.9). The silhouette coefficients for these roles range from 5.41×10^{-1} and 5.58×10^{-1}, indicating reasonable role structures for the network (see Section 2.6.1). However, as we search longer neighborhood patterns in the network, i.e., loosened the parameter β, the silhouette statistic extracts 2 roles for all of the algorithms. Also, the silhouette coefficients for β_2 and β_3 are larger than those for β_1.

For the role structure extracted for $\beta = \beta_2$ and $\beta = \beta_3$, k-means clustering with the silhouette statistic extracts 8 roles from Cheng et al.'s low-rank similarity measure and the silhouette coefficient for this role structure is 6.70×10^{-1}, which is a reasonable role structure. Table 6.10 displays the partitions of the countries extracted from the similarity measures using k-means clustering.

Observe that both CPM and the silhouette statistic for the neighborhood pattern similarity measure detects 2 roles within the data set (see Tables 6.8 and 6.10). Also, note that while the partitions for k-means clustering and community detection differ slightly by the assignments of three countries – Denmark, India, and Spain – for the neighborhood pattern similarity approximations,

Table 6.10: Countries partitioned by silhouette statistic.

Algorithm	Countries
Riemannian Opt β_2 β_3	Algeria, Egypt, Madagascar, Mauritius, Morocco, Reunion, Seychelles, Southern Africa, Tunisia, Bangladesh, Indonesia, Israel, Jordan, Kuwait, Malaysia, Oman, Pakistan, Philippines, Singapore, Sri Lanka, Thailand, Croatia, Cyprus, Czech Rep., Finland, Greece, Hungary, Iceland, Ireland, Latvia, Norway, Poland, Portugal, Rep. of Moldova, Romania, Slovenia, Turkey, Belize, Canada, El Salvador, Guatemala, Honduras, Mexico, Nicaragua, Panama, Australia, Fiji, New Zealand, Argentina, Barbados, Bolivia, Brazil, Chile, Colombia, Ecuador, French Guiana, Guadeloupe, Martinique, Paraguay, Peru, Trinidad and Tobago, Uruguay, Venezuela
	China, Hong Kong, **INDIA**, Japan, Rep. Of Korea, Austria, Belgium /Lux., **DENMARK**, France Mon., Germany, Italy, Netherlands, **SPAIN**, Sweden, Switzerland, United Kingdom, United States
Cheng LR	China, Hong Kong, India, Indonesia, Japan, Rep. Of Korea, Malaysia, Singapore, Thailand, United States, Australia, New Zealand
	Algeria, Madagascar, Mauritius, Morocco, Reunion, Seychelles, Bangladesh, Jordan, Kuwait, Oman, Pakistan, Philippines, Sri Lanka, Cyprus, Iceland, Latvia, Belize, Nicaragua, Fiji, French Guiana, Guadeloupe, Martinique, Paraguay, Uruguay
	Canada, Mexico, Argentina, Brazil, Chile, Colombia, Ecuador, Peru, Venezuela
	Egypt, Southern Africa, Tunisia, Greece, Romania, Turkey, Israel
	Austria, Belgium /Lux., Denmark, Finland, France Mon., Germany, Ireland, Italy, Netherlands, Norway, Poland, Portugal, Spain, Sweden, Switzerland, United Kingdom
	El Salvador, Guatemala, Honduras, Panama
	Croatia, Czech Rep., Hungary, Rep. of Moldova, Slovenia
	Barbados, Bolivia, Trinidad and Tobago

the reduced graphs are the same (see Figure 6.15[2]). When considering how the data is partitioned by k-means clustering compared to CPM, the main difference is that k-means clustering included India, Denmark, and Spain in role **B** while the CPM places them in role **A**. However, the two role graphs groups the core countries (with a few semiperphery countries) together in one role and the semiperiphery and periphery countries in the other role. Also, the self-loop in role **B** and the arc from **B** to **A** indicate that the core countries dominate the dataset (which is true) and that the

[2]Maps were colored using the open source website: `http://www.paintmaps.com/index.php?` [pai14]

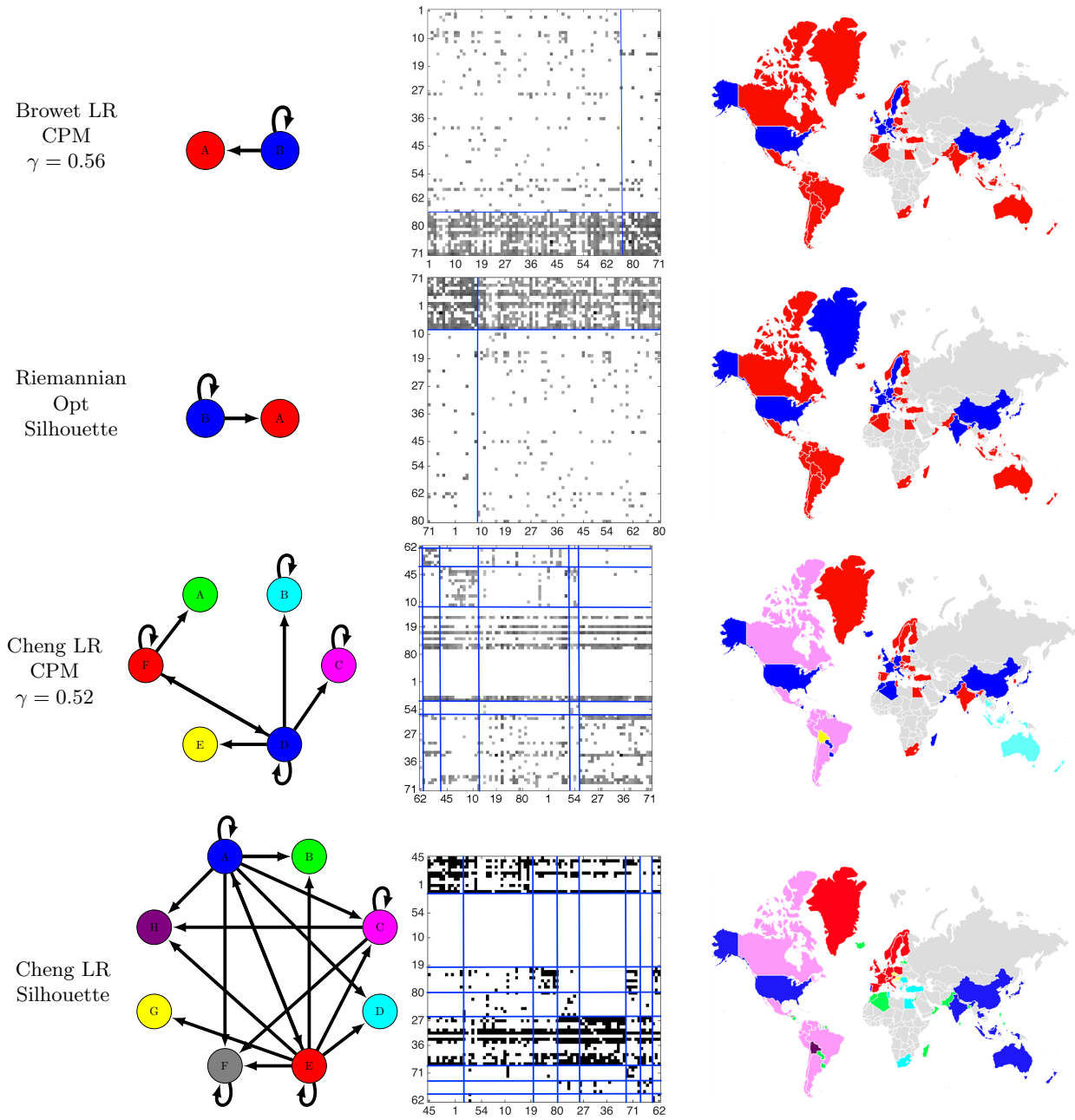


Figure 6.15: Role graphs for metal world trade network.

periphery and semiperiphery countries imported very little (or not at all) between themselves and from **B**. Therefore, these role graphs are an adequate representation of the metal trade data.

Lastly for Cheng et al.'s low-rank similarity measure, CPM and the silhouette statistic determine

that the network contains 6 and 8 roles, respectively. Observe that some blocks in the permuted adjacency matrices, while much denser than others, have entire rows of zeros. Recall that these types of blocks are called row-regular blocks and are still considered a role in the reduced graph [DBF05]. However, most of the row-regular blocks are sparse, which indicates an inconsistency in the role partition, since an arc may or may not need to be added to represent the data. These inconsistencies indicate that the role graphs may not be a good representation of the data.

For the low-rank similarity measure of Cheng et al. using the silhouette statistic, there are obvious null blocks within its permuted adjacency matrix. Thus, this partitioning may not be considered unreasonable and cannot be dismissed immediately. However, if we consider the image matrix of the role graph

$$B = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

note that the roles **B** and **D** are structurally equivalent, which violates the role constraint. So, the countries in roles **B** and **D** should be grouped together into a single role. Thus, this role graph may be over partitioned.

Also, the country partitioning for the low-rank similarity measure of Cheng et al. makes little sense from a global trade point-of-view. For example, when using CPM to group the roles, the approach placed several of the countries classified as core countries (e.g., United States, China, etc.) in the same roles as periphery countries (e.g., Fiji, Belize, etc.). When using k-means clustering with the silhouette statistic, the core countries are not grouped with any periphery countries, but they have been divided and placed into roles with semipherphery countries. Therefore, k-means clustering with the silhouette statistic gave better results than community detection with CPM for the Cheng et al.'s similarity measure, but the results are still difficult to interpret.

In conclusion, the neighborhood patterns of the arcs provide extra information in the similarity measure that allow us to accurately represent the metal world trade data in the reduced graphs, in comparison to Cheng et al.'s similarity measure that provided block structures with several inconsistencies. Thus, only considering pairwise nodes in the similarity measure is not sufficient.

Browet's approach and our approaches extract 2 roles and partition the network similarly (excluding India, Denmark, and Spain), where countries that dominate the trade network were placed into one role and everyone else was place in the other role. CPM is able to extract the 2 roles when the parameter β was tight, while the silhouette statistic requires a looser β bound to extract the 2 roles. Regardless, the 2 role structures were good representations of the network.

6.3 Summary

In this chapter, we evaluate our two-phase approach on unsigned networks for both unweighted and weighted graphs. From our experiments in Section 6.1.3, we show that using LRBFGS to compute a low-rank approximation of the neighborhood pattern similarity measure is robust in time when there exists graph noise, when we search for longer neighborhood patterns in the graph, or when we assume that the rank of the similarity matrix is large. Being robust in the presence of noise is important since real world networks tend to be noisy, i.e., for unweighted networks there may exist edges in blocks that would be classified as null blocks in the adjacency matrix or there may be edges missing in dense blocks, and for weighted networks edges that are really small (or large) may lead to misleading partitions of the network. Robustness across the rank and parameter β is also ideal since, in practice, the optimal rank and β are unknown and we would need to probe across a range of values to be able to interpret the results. Therefore, we recommend LRBFGS as a generic first phase approach for computing a low-rank approximation of the similarity matrix.

In Section 6.1.4, we show that using k-means clustering with the silhouette statics is robust in quality when extracting the role partition from noisy graphs, when searching longer neighborhood patterns for ranks that are larger than the number of roles. In practice, the number of roles is unknown and we need to run k-means clustering on several assumed number of roles, which is inefficient for large n or r. Therefore, an approach to determine a better value for the bound on the number of roles is necessary to improve the second phase our role extraction algorithm.

Lastly, in Section 6.2, we show that we are able to achieve reasonable results for the world trade network using our role extraction algorithm. But, our results indicate that we should compute longer neighborhood patterns to get a stable role partition of the network or that the application user must provide additional information to assess the resulting roles for various betas. The community detection approach with CPM used by Browet and Van Dooren was able to detect the role

structures for β_1; however, searching across the resolution parameter in CPM for stable community partitions may be just as costly as searching across the index list of roles for the silhouette statistic. Therefore, while we have improved the second phase of the indirect approach to the role extraction problem, more efficient methods to determine the role partitions are needed.

CHAPTER 7

ONE-PHASE INDIRECT APPROACH: RIEMANNIAN RANK-ADAPTIVE ROLE EXTRACTION METHOD

In this chapter, we combine the two phases of our indirect role extraction method into a one-phase indirect approach that iteratively approximates the best low-rank solution of the neighborhood pattern similarity measure and determines the role structure of the network. To do this, we reformulate the problem as a rank-inequality constrained optimization problem.

7.1 Introduction

Recall in Chapter 4 that we solve the first phase of our indirect approach to the role extraction problem by approximating a low-rank solution of the neighborhood pattern similarity measure by solving (4.7) on $S_+(n,r)$. We consider the low-rank factorizatoin $S^{(r)} = X^{(r)}(X^{(r)})^T$ and represent $S_+(n,r)$ by the quotient manifold $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$. Then, we apply k-means clustering on the low-rank factor $X^{(r)}$ and group the rows of $X^{(r)}$ into k roles. We implement k-means clustering for an assumed index list of roles from 2 to $r+1$ and determine the optimal number of roles by the silhouette statistic (see Section 2.6.1).

We suggest the user set the role index list $\{2,3,\ldots,r+1\}$, where r is the rank of the similarity matrix, since, in practice, the rank of the similarity matrix and the number of roles are unknown. This suggestion is supported by our analytical results of the rank-role relationship between the neighborhood pattern similarity measure and the number of roles in the network for the ideal case where all of the nodes in the network are structurally equivalent (see Chapter 3). That is, by Therorem 3.0.4, the rank of the neighborhood pattern similarity matrix is less than or equal to the number of roles in the network and greater than or equal to the rank of the image matrix B.

If the graph is noisy, i.e., the nodes are regularly equivalent or there are edges in the null blocks of the adjacency matrix, then the rank of the similarity measure is not equal to the number or roles. Our empirical results in Section 6.1 indicated that if there exists enough of a spectral gap in the

singular values of the similarity matrix, the clustering algorithm can still extract the role partition up to some threshold (see Section 6.1.4). Our experiments also indicated that the minimal rank necessary to extract the role structure is related to the number of roles.

Therefore, rather than choosing a particular value for the expected rank or number of roles, in this chapter we consider extracting the role partition for the graph by optimizing the cost function (4.5) under the rank inequality constraints to approximate a low-rank solution to the similarity measure. That is, we solve the optimization problem

$$\max_{S \in \mathcal{D}_{\leq p}} f(S) = \operatorname{trace}\left(S\left(S_1 - \frac{1}{2}S + \beta^2 ASA^T\right)\right) \tag{7.1}$$

for a low-rank approximation to the similarity measure where $\mathcal{D}_{\leq p} = \{S \in \mathcal{S}^n \mid S \succeq 0,\ \operatorname{rank}(S) \leq p\}$. We use Riemannian rank-adaptation techniques from [Zho15, HGZ16, ZHG$^+$16] and heuristics derived from our observations in Chapter 3 and Section 6.1 to define our one-phase role extraction process.

In Section 7.2, we give an overview of our one-phase role extraction algorithm. In Sections 7.3, 7.4, and 7.5, we describe the main components in our approach and define the Riemannian objects required. In Section 7.6, we state the full algorithm and summarize the heuristics for choosing the parameters of the algorithm. Lastly, in Section 7.7, we compare the time and NMI values of one-phase algorithm with our two-phase algorithm to assess the effectiveness and robustness of the algorithm.

7.2 Overview of One-Phase Algorithm

Due to work by Browet and Van Dooren in [Bro14,BD14] and our work in the previous chapters, the two-phase indirect approach to the role extraction problem has become efficient in time and space and it can determine a representative role partition of a network. However, both two-phase methods suffer from parameter and complexity issues.

Both algorithms are very dependent on the choice of the rank r used to compute a low-rank approximation of the similarity measure. In practice, the optimal choice of r is unknown. Also, we have shown empirically in Section 6.1.4 that if r is assumed to be less than the number of roles, then the algorithms cannot extract the roles. However, if r is assumed to be too large compared to the number of roles, then we may not extract the role partition and the algorithm suffers in time and

computational complexity. Thus, we propose to combine the two-phase indirect approach into a one-phase approach to solve the role extraction problem by optimizing problem (7.1) to determine a low-rank solution of the similarity measure that we can use to extract the role partition.

Our Riemannian rank-adaptive role extraction method (RRAREM) repeats the following three tasks to determine the role structure of a network:

(1) *Riemannian update*: Given the initial guess $\hat{X}_k^{(r)}$ on $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$, we solve problem (4.7) and return a point $X_k^{(r)}$ when a suitable stopping criterion is satisfied. This is equivalent to the first phase in our two-phase role extraction approach for a fixed rank r (Section 4.1).

(2) *Role extraction update*: We extract the role partition from the iterate $X_k^{(r)}$ by k-means clustering with the silhouette statistic where the range of the index list of clusters is between r and the previous number of roles in the network. This is similar to the second phase in our two-phase role extraction approach (Section 4.2), except that we run k-means clustering on a narrower index list of roles and repeatedly update the partition as we converge towards a low-rank solution.

(3) *Rank-related update*: We generate new iterate $X_{k+1}^{(\tilde{r})} \in \mathcal{D}_{\leq p}$, where rank $\tilde{r}$ is the new rank, by either increasing or decreasing the rank. We use the rank-adaptive strategies in [Zho15,HGZ16, ZHG$^+$16] to update the rank.

These three steps are repeated until convergence, where we output a low-rank matrix and role structure of the network.

In the next three sections, we describe the implementation details of the role extraction update and rank-related update, including the rank reduction and rank increasing strategies.

7.3 Role Extraction Step

Given the factor $X_k^{(r)}$ on the k-th iteration of the algorithm, we must extract the role partition. In Section 6.1.4 we showed empirically that using k-means clustering with the silhouette statistic to extract the role structure from the low-rank similarity measure can be time consuming since we must determine the partition for each assumed number of roles using k-means clustering. We have shown analytically for the ideal case (Chapter 3) and empirically for practical cases (Section 6.1.4) that there is a relationship between the rank of the neighborhood pattern similarity measure and the number of roles. We exploit this relationship when choosing the index list of the number of clusters we use for k-means clustering.

Similar to the second phase in our two-phase approach, we first scale the low-rank factor $X_k^{(r)}$ by its unscaled self-similarity scores. Then, we use k-means clustering with the silhouette statistic defined in Section 2.6.1 for an assumed index list of roles. The list of roles is determined by the rank r of $X_k^{(r)}$ and the previously assumed number of roles q. If $r < q$, then the index list is given by $\{r-1, \ldots, q+1\}$; otherwise, it is $\{q-1, \ldots, r+1\}$. Recall that the silhouette value cannot be calculated for 1 cluster. Therefore, if either r, or q, equals 1, we start the index list from $r+1$, or $q+1$. After k-means clusters the role assignment for each index in the list and computed the silhouette value, the algorithm returns the optimal number of roles q^* and role partition c^* associated with the maximal silhouette value. Algorithm 2 summarizes the role extraction step.

Algorithm 2 Role Partition and Number of Roles Algorithm

Input: $X \in \mathbb{R}^{n \times r}$; number of roles q;

Output: number of roles q^*; partition c^*;

1: Scale X by diagonal matrix $D_{S^{(r)}}$ of unscaled self-similarity scores;
2: Determine cluster list $\{l, \ldots, u\}$ by using r and q as bounds;
3: $l \leftarrow \min\{r, q\}$; $u \leftarrow \max\{r, q\} + 1$;
4: **if** $l = 1$ **then**
5: $\quad l \leftarrow l + 1$;
6: **else if** $l > 2$ **then**
7: $\quad l \leftarrow l - 1$;
8: **end if**
9: Given X, compute k-means clustering for each assumed number of clusters in $\{l, \ldots, u\}$ and compute the silhouette measure for each data set partition;
10: Choose the number of clusters and partition that gives the maximum silhouette measure as the number of roles q^* and role partition c^*;

7.4 Adaptive Rank Reduction Strategy

We use the adaptive rank reduction strategy in [HGZ16] to reduce the rank of an iterate in our one-phase role extraction algorithm.

Recall that our cost function (4.5) is defined on the quotient manifold $\mathbb{R}_*^{n \times r}/\mathcal{O}_r$. Therefore, the domain of the cost function is not closed. That is, the sequence of iterates $\{X_k^{(r)}\}$ may have a limit point $X_*^{(r)}$ that has rank less than r. For this situation, we must be able to reduce the rank of $X_k^{(r)}$.

Consider the thin SVD of $X_k^{(r)}$, i.e., $X_k^{(r)} = U\Sigma V^T$ where $U \in \mathbb{R}^{n\times r}$ and $V \in \mathbb{R}^{r\times r}$ are matrices with orthonormal columns and $\Sigma = \text{Diag}(\sigma_1, \ldots, \sigma_r)$ where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$. Let $\hat{\sigma} = \|\Sigma\|_F/\sqrt{r}$. If there exists $p < r$ such that $\sigma_p/\hat{\sigma} > \delta$ and $\sigma_{p+1}/\hat{\sigma} < \delta$ for some given threshold δ, then we set the initial point for optimizing the cost function (4.5) over $\mathbb{R}_*^{n\times p}/\mathcal{O}_p$ to be $X_*^{(r)} = U(:.1:p)\,\text{Diag}(\sigma_1, \cdots \sigma_p)$, where $\text{Diag}(\sigma_1, \cdots \sigma_p)$ denotes a $p \times p$ diagonal matrix of the singular values. The details for reducing the rank are given in Algorithm 3.

The rank reducing parameter δ is initialized close to 1. Unlike in [HGZ16], we decrease the threshold parameter δ after each iterate. That is, for examples like Figure 3.4 in Chapter 3, the rank of the similarity matrix is large, but there exists a small, but noticeable, gap in the singular values that we have shown empirically is associated with the number of roles in the graph. Reducing the parameter δ allows us to detect this gap in the singular values.

Algorithm 3 is ideal when the true rank of the minimizer is known and the current iterate $X_k^{(r)}$ has a rank higher than the desired rank. In practice, we do not know the rank of the neighborhood pattern similarity measure; however, we know that there exists a relationship between the rank and the number of roles of the graph. So for this approach, a priori information about the number of roles means we can assume a rank slightly larger than the number of roles for our initial iterate and reduce the rank accordingly.

Algorithm 3 Adaptive Rank Reduction Strategy [HGZ16]

Input: $X_k^{(r)} \in \mathbb{R}^{n\times r}/\mathcal{O}_r$; threshold δ;
Output: $Y \in \mathbb{R}^{n\times p}$;

1: Take thin singular value decomposition for $X_k^{(r)}$, i.e., $X_k^{(r)} = U\,\text{Diag}(\sigma_1, \ldots, \sigma_r)V^T$, where $U \in St(r, n)$, $V \in St(r, r)$, and $\sigma_1 \geq \cdots \geq \sigma_r \geq 0$;
2: Set $\tilde{\sigma} = \|\text{Diag}(\sigma_1, \ldots, \sigma_r)\|_F/\sqrt{r}$;
3: **if** $\sigma_r/\tilde{\sigma} > \delta$ **then**
4: $\quad p \leftarrow r$, $Y \leftarrow X_k^{(r)}$ and return;
5: **else**
6: $\quad$ Find p such that $\sigma_p/\tilde{\sigma} > \delta$ and $\sigma_{p+1}/\tilde{\sigma} \leq \delta$;
7: $\quad$ Let $Y = U(:, 1:p)\,\text{Diag}(\sigma_1, \ldots, \sigma_p)$ and return;
8: **end if**

7.5 Adaptive Rank Increasing Strategy

In Section 6.1, we showed empirically that it was difficult to extract (or impossible if the solution was rank-1) the role structure from a solution with rank less than the number of roles. So, an adaptive rank increasing strategy is needed.

The adaptive rank increasing strategy we use is by Zhou et al. [Zho15, ZHG+16]. Their approach can increase the rank by an adaptively chosen amount during the iteration, unlike previous methods that increase the rank by a fixed number [Ye05, ABG07, JBAS10, LKLS13, MMBS13]. Their rank-update procedure is governed by parameters that can be adjusted to balance between saving space and time for the rank reduction and achieving higher accuracy for the rank increment. Additionally, a convergence analysis is provided and their algorithm is demonstrated on the weighted low-rank approximation problem and the low-rank graph similarity measure by Blondel et al. [Zho15, ZHG+16].

In this section, we define additional objects that are needed to increase the rank and give a brief overview of their rank increasing strategy.

7.5.1 Symmetric Positive Semidefinite Fixed-Rank Manifold as an Embedded Submanifold

The convergence analysis of the rank increasing strategy in [Zho15, ZHG+16] was on the fixed-rank manifold viewed as an embedded submanifold of $\mathbb{R}^{m\times n}$. Since the analysis does not support the quotient manifold, we define an equivalent cost function of (4.5) that maps $\mathcal{S}_+(n,r)$ viewed as an embedded submanifold of the general linear group, i.e., the set of all non-singular real $n \times n$ matrices, to $\mathbb{R}$. That is, for $S \in \mathcal{S}_+(n,r)$, we define the function $f_r : \mathcal{S}_+(n,r) \to \mathbb{R} : S \mapsto f_r(S)$ such that for the rank-inequality problem (7.1) we have that $f_r = f|_{\mathcal{S}_+(n,r)}$ where f is given by (4.1). Thus, when we want to increase the rank of the similarity matrix, we convert from the quotient manifold to the embedded submanifold and use the rank increasing strategy by Zhou et al. [Zho15, ZHG+16]. Therefore, we need to define the Riemannian objects of $S \in \mathcal{S}_+(n,r)$ as an embedded submanifold of the general linear group.

In [VAV09], Vandereycken et al. studied $S_+(n,r)$ as an embedded submanifold of the general linear group denoted GL_n, of dimension $nr - r(r-1)/2$. The authors considered $S = XX^T$ and derived some Riemannian objects for the embedded submanifold. For implementation purposes,

we consider the eigendecomposition (or SVD) of S, i.e., $S = U\Sigma U^T$, and derive the Riemannian objects for the manifold

$$S_+(n,r) = \left\{U\Sigma U^T \mid U \in St(r,n),\ \Sigma = \mathrm{Diag}(\sigma_1, \ldots, \sigma_r)\right\}, \tag{7.2}$$

where Σ is a diagonal matrix of the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$. Some of the objects of the manifold with this representation can be found in [VV10].

The tangent space is

$$\mathrm{T}_S S_+(n,r) = \left\{ZS + SZ^T \mid Z \in \mathbb{R}^{n\times n}\right\}. \tag{7.3}$$

However, this is an over-parametrization of the tangent space since the tangent space is of dimension $nr - r(r-1)/2$. Thus, if $S = U\Sigma U^T$ then the tangent space is given by

$$\mathrm{T}_{U\Sigma U^T} S_+(n,r) = \left\{ \begin{bmatrix} U & U_\perp \end{bmatrix} \begin{bmatrix} H & K^T \\ K & 0 \end{bmatrix} \begin{bmatrix} U^T \\ U_\perp^T \end{bmatrix} \middle|\ H = H^T \in \mathbb{R}^{r\times r}, K \in \mathbb{R}^{(n-r)\times r} \right\}, \tag{7.4}$$

where $U_\perp \in \mathbb{R}^{n\times(n-r)}$ is the orthogonal complement of U in GL_n, i.e., $U_\perp^T U = 0$.

The metric endowed from the Euclidean space is given by

$$g_S(\eta_S, \xi_S) = \mathrm{trace}\left(\eta_S^T \xi_S\right) \tag{7.5}$$

for all $\eta_S, \xi_S \in \mathrm{T}_S S_+(n,r)$. Taking $\eta_S = UH_1U^T + U_\perp K_1 U^T + UK_1U_\perp^T$ and $\xi_S = UH_2U^T + U_\perp K_2 U^T + UK_2U_\perp^T$, equation (7.5) becomes

$$g_{U\Sigma U^T}(\eta_S, \xi_S) = \mathrm{trace}\left(H_1^T H_2 + 2K_1^T K_2\right). \tag{7.6}$$

The orthogonal projection onto $\mathrm{T}_S\mathcal{S}_+(n,r)$ is given by

$$\mathrm{P}_S^t(\eta_S) = \mathrm{P}_{U\Sigma U^T}^t(\eta_S) = \frac{1}{2}\left(\mathrm{P}_U(\eta_S + \eta_S^T)\mathrm{P}_U + \mathrm{P}_U^\perp(\eta_S + \eta_S^T)\mathrm{P}_U + \mathrm{P}_U(\eta_S + \eta_S^T)\mathrm{P}_U^\perp\right), \tag{7.7}$$

where $P_U = UU^T$ and $P_U^\perp = I - P_U$.

Since $\mathcal{S}_+(n,r)$ is an embedded submanifold, the Riemannian gradient of f_r is equal to the projection of the gradient of f on $\mathbb{R}^{n\times n}$ onto $T_S\mathcal{S}_+(n,r)$ [Section 3.6.1] [AMS08], i.e., the Riemannian gradient of f_r is given by

$$\mathrm{grad}\, f_r(S) = \mathrm{P}_S\left(\mathrm{grad}\, f(S)\right) \tag{7.8}$$

where $\mathrm{P}_S^t(\eta_S)$ is defined by (7.7) and $\mathrm{grad}\, f(S)$ is the Euclidean gradient (4.2).

7.5.2 Retractions on the Embedded Submanifold

In this section, we define a retraction on $\mathcal{S}_+(n,r)$ that uses the two factor representation (U,Σ) for S defined on the embedded submanifold $\mathcal{S}_+(n,r)$. That is, given $\dot{S} \in \mathrm{T}_S\mathcal{S}_+(n,r)$, we want to find $R_S(\dot{S}) = U_+\Sigma_+U_+^T$.

In [KL07, Section 2], Koch and Lubich showed for $m \times n$ real matrices on the fixed-rank manifold rewritten in the form $X = U\Sigma V^T$, where $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{n\times r}$ have orthonormal columns and $\Sigma \in \mathbb{R}^{r\times r}$ is nonsingular, that there is a unique representation in the tangent space of the fixed-rank manifold.

These results can be easily applied to the symmetric positive semidefinite fixed-rank manifold since for any $n \times n$ symmetric positive semidefinite matrix S, the (not unique) eigendecomposition is $S = U\Sigma U^T$ where $U \in \mathbb{R}^{n\times r}$ has orthonormal columns and $\Sigma \in \mathbb{R}^{r\times r}$ is nonsingular and diagonal. Thus, given a pair (U,Σ) for $S \in \mathcal{S}_+(n,r)$, then there exists a unique representation $(\dot{U},\dot{\Sigma})$ of any $\dot{S} \in \mathrm{T}_S\mathcal{S}_+(n,r)$ that can be computed efficiently and satisfies

$$\dot{S} = \dot{U}\Sigma U^T + U\dot{\Sigma}U^T + U\Sigma\dot{U}^T, \tag{7.9}$$

$$U^T\dot{U} = 0. \tag{7.10}$$

The computation of $(\dot{U},\dot{\Sigma})$ is similar to the computation of $(\dot{U},\dot{\Sigma},\dot{V})$ in [Zho15, Section 4.3.4] on the fixed-rank manifold. The computations have been simplified since, due to the symmetry of S and $\dot{S}$, $V = U$ and $\dot{V} = \dot{U}$. Since $\dot{S} = \dot{S}^T$ implies that $\dot{\Sigma} = \dot{\Sigma}^T$, substituting $V = U$ into equations (4.20), (4.21), and (4.22) in [Zho15, Section 4.3.4], we get the following equations for $\dot{U}$ and $\dot{\Sigma}$:

$$\dot{U} = (I - UU^T)\dot{S}U\Sigma^{-1}, \tag{7.11}$$

$$\dot{\Sigma} = U^T\dot{S}U. \tag{7.12}$$

Once we have $\dot{U}$ and $\dot{\Sigma}$, we can use them to derive a retraction onto $\mathcal{S}_+(n,r)$. Absil and Oseledets in [AO15] analyzed retractions on the fixed-rank manifold. Most of the retractions in their paper preserve symmetry, i.e., $R_S(\dot{S})$ is a symmetric matrix if S and $\dot{S}$ are symmetric matrices. Using retractions related to the compact Stiefel manifold and projections, we can define a retraction on $\mathcal{S}_+(n,r)$ based on the two factor representation of S [AM12, AO15]. The two-factor SVD-type

retraction is the projection-like retraction defined as

$$R_S(\dot{S}) = \underset{Y \in \mathcal{S}_+(n,r)}{\arg\min} \|Y - (S + \dot{S})\|_F,$$

where $\|\cdot\|_F$ denotes the Frobenius norm [AM12, AO15]. Therefore, simplifying the SVD-type retraction in [Van13] on the fixed-rank manifold, the two-factor SVD-type retraction onto $\mathcal{S}_+(n,r)$ can be computed as follows:

$$R_S(\dot{S}) = U_+ \Sigma_+ U_+^T, \tag{7.13}$$

where

$$\begin{aligned} \dot{U}\Sigma &= QR, \\ U_s \Sigma_s U_s^T &= \begin{bmatrix} \Sigma + \dot{\Sigma} & R^T \\ R & 0 \end{bmatrix} \text{ by the SVD,} \\ U_+ &= \begin{bmatrix} U & Q \end{bmatrix} U_s(:, 1:r), \\ \Sigma_+ &= \Sigma_s(1:r, 1:r). \end{aligned}$$

This algorithm requires the QR factorization of the $n \times r$ matrix $\dot{U}\Sigma$, the SVD computation of a $2r \times 2r$ matrix, and $4nr^2$ operations in the matrix multiplication yielding a total of $O(nr^2) + O(r^3)$ operations where the coefficient of the first term depends upon the method used to compute the QR factorization.

7.5.3 Tangent Cone

Note that the set $\mathcal{D}_{\leq p}$ is not a manifold, but the union of symmetric positive semidefinite fixed-rank manifolds, i.e.,

$$\mathcal{D}_{\leq p} = \bigcup_{r \leq p} \mathcal{S}_+(n,r). \tag{7.14}$$

So, for any $S \in \mathcal{D}_{\leq p}$ with rank less than p, the set $\mathcal{D}_{\leq p}$ does not have a tangent space, but there exists a tangent cone (see Section 1.3.7). Therefore, we can define directions that increase rank in the tangent cone $\mathrm{T}_S\mathcal{D}_{\leq p}$.

Huang et al. [HGZ16] derived the tangent cone for the union of Hermitian positive semidefinite fixed-rank manifolds and the derivation in Lemma 7.5.1 is modified for the real case.

Lemma 7.5.1 *Let $S \in \mathcal{D}_{\leq p}$ with rank $r \leq p$. Then the tangent cone to $\mathcal{D}_{\leq p}$ at a point S is*

$$\begin{aligned}\mathrm{T}_S\mathcal{D}_{\leq p} = &\left\{ \begin{bmatrix} U^{(r)} & (U^{(r)})_\perp \end{bmatrix} \begin{bmatrix} H & W^T \\ W & R \end{bmatrix} \begin{bmatrix} (U^{(r)})^T \\ (U^{(r)})_\perp^T \end{bmatrix} \middle| \; H = H^T \in \mathbb{R}^{r\times r}, W \in \mathbb{R}^{(n-r)\times r}, \right. \\ &\left. R = R^T \in \mathbb{R}^{(n-r)\times(n-r)}, \;\; \mathrm{rank}(R) \leq p - r \right\} \\ = &\left\{ U^{(r)} H (U^{(r)})^T + U^{(r)} W^T (U^{(r)})_\perp^T + (U^{(r)})_\perp W (U^{(r)})^T + (U^{(r)})_\perp R (U^{(r)})_\perp^T \;\middle|\; \right. \\ &\left. H = H^T \in \mathbb{R}^{r\times r}, W \in \mathbb{R}^{(n-r)\times r}, R = R^T \in \mathbb{R}^{(n-r)\times(n-r)}, \;\; \mathrm{rank}(R) \leq p - r \right\}, \end{aligned} \tag{7.15}$$

where $U^{(r)} \in St(r,n)$ and $U_\perp^{(r)} \in \mathbb{R}^{n\times(n-r)}$ is the orthogonal complement of $U^{(r)}$ in $\mathbb{R}^{n\times n}$.

7.5.4 Rank-Related Retractions

In order to change the rank, we must derive a rank-related retraction, where the rank-related retraction maps a tangent vector on the tangent cone back to $\mathcal{D}_{\leq p}$. We define a SVD-type rank-related retraction, which is similar to the SVD-type rank-related retraction [Zho15, Section 4.3.5].

Let f_F denote a cost function that maps from $\mathcal{S}_+^n$ to $\mathbb{R}$, where $f = f_F|_{\mathcal{D}_{\leq p}}$ and $f_F|_{\mathcal{S}_+(n,r)} = f_r = f|_{\mathcal{S}_+(n,r)}$. Then the gradient of f_F (which is called the full gradient) is the projection of the Euclidean gradient (4.2) onto S_+^n, i.e.,

$$\mathrm{grad}\, f_F(S) = [\mathrm{grad}\, f(S)]^+, \tag{7.16}$$

where $[\cdot]^+$ is given by (5.15).

The full gradient $\mathrm{grad}\, f_F(S^*)$ of a point $S^* = U_r \Sigma_r U_r^T$ on $\mathcal{S}_+^n$ can be written as

$$\mathrm{grad}\, f_F(S^*) = \begin{bmatrix} U_r & (U_r)_\perp \end{bmatrix} \begin{bmatrix} H & K^T \\ K & R \end{bmatrix} \begin{bmatrix} U_r^T \\ (U_r)_\perp^T \end{bmatrix}, \tag{7.17}$$

where $H = H^T \in \mathbb{R}^{r\times r}$, $R = R^T \in \mathbb{R}^{(n-r)\times(n-r)}$, and $K \in \mathbb{R}^{(n-r)\times r}$ Therefore, increasing the rank depends upon the symmetric matrix R.

Let $\tilde{r} = r + \Delta r$, where Δr is the amount the rank has been increased. Then a search direction on the tangent cone satisfies [Zho15]

$$\eta^* = \underset{\eta \in \mathrm{T}_S\mathcal{D}_{\leq \tilde{r}}}{\arg\min} \|\mathrm{grad}\, f_F(S^*) - \eta\|_F. \tag{7.18}$$

Due to the symmetry of η^*, we can simplify the three-factored SVD-type rank-related retraction for the union of fixed-rank manifolds in [Zho15, Section 4.3.5] to a two-factored SVD-type rank-related retraction

$$\tilde{R}_S(\eta^*) = \tilde{U}_+ \tilde{\Sigma}_+ \tilde{U}_+^T, \tag{7.19}$$

given by

$$\begin{aligned}
\dot{U}_{\tilde{r}}\Sigma &= QR, \\
U_s\Sigma_s U_s^T &= \begin{bmatrix} \Sigma_{\tilde{r}} + \dot{\Sigma}_{\tilde{r}} & R^T \\ R & 0 \end{bmatrix} \text{ by the SVD,} \\
\tilde{U}_+ &= \begin{bmatrix} U_{\tilde{r}} & Q \end{bmatrix} U_s(:, 1:r), \\
\tilde{\Sigma}_+ &= \Sigma_s(1:r, 1:r)
\end{aligned}$$

where $\tilde{r} = r + \Delta r$, $U_{\tilde{r}} = \begin{bmatrix} U_{\tilde{r}} & (U_{\tilde{r}})_\perp \end{bmatrix}$, $\Sigma_{\tilde{r}} = \begin{bmatrix} \Sigma_{\tilde{r}} & 0^{\tilde{r}\times(n-\tilde{r})} \\ 0^{(n-\tilde{r})\times\tilde{r}} & 0^{(n-\tilde{r})\times(n-\tilde{r})} \end{bmatrix}$, and the two factors $(\dot{U}_{\tilde{r}}, \dot{\Sigma}_{\tilde{r}})$ are defined as

$$\dot{U}_{\tilde{r}} = (I - U_{\tilde{r}}U_{\tilde{r}}^T)\eta^* U_{\tilde{r}}\Sigma_{\tilde{r}}^{-1}, \tag{7.20}$$

$$\dot{\Sigma}_{\tilde{r}} = U_{\tilde{r}}^T \eta^* U_{\tilde{r}}. \tag{7.21}$$

7.5.5 Overview of Adaptive Rank Increasing Strategy

Zhou et al.'s rank increasing strategy uses the first order information provided by grad f_F and grad f_r to increase the rank. In their Riemannian rank-adaptive method, the authors first check if the angle and difference between grad f_F and grad f_r at the current iterate $X_k^{(r)}$ are both large. If so, then it is assumed that the maximal rank has not been reached and their method increases the rank of $X_k^{(r)}$ [Zho15, ZHG$^+$16].

As in Zhou et al., we use the idea of the angle and difference between the full gradient grad f_F and grad f_r (equations (7.16) and (7.8), respectively) to determine if we have reached the maximal rank or if we need to increase the rank. Parameters ϵ_1 and ϵ_2 in Algorithm 5 are the parameters associated with this check and were analyzed in [Zho15, ZHG$^+$16]. If they are small, then the algorithm is more likely to increase the rank. Zhou et al. in [Zho15, ZHG$^+$16] discuss choices of ϵ_1 and ϵ_2 for their Riemannian rank-adaptive method to work efficiently.

If the rank is allowed to increase, then the algorithm determines how much by incrementing $\tilde{r}$ from r to at most p until the tangent of the angle between $-\operatorname{grad} f_F(S_k^{(r)})$ and its projection η^* onto $\mathrm{T}_{S_r^{(k)}}\mathcal{D}_{\leq\tilde{r}}$ is smaller than some parameter ϵ_4 where $0 < \epsilon_3 \ll 1$. Assuming that $\tilde{r} \neq p$, it is assumed that the update rank of η^* is $\tilde{r} > r$. Then, η^* is updated by the rank-related retraction and a stepsize is chosen by an Armijo-type backtracking procedure (Algorithm 4) to yield the next iterate $S_{k+1}^{(\tilde{r})}$, whose rank is less than or equal to $\tilde{r}$ but not necessarily equal to $\tilde{r}$. The details of rank increasing strategy are given in Algorithms 4 and 5.

Algorithm 4 Rank-related Armijo backtracking [ZHG+16]

Inherit $\tilde{R}$, $S_k^{(r)}$, β, $\bar{\alpha}$, η^*, $\mathcal{S}_+(n,\tilde{r})$, f, an d σ from Algorithm 6 where Algorithm 4 is called;
Compute the smallest nonnegative integer m such that

(i) $\tilde{R}_{S_k^{(r)}}(\beta^m\bar{\alpha}\eta^*)$ belongs to $\mathcal{D}_{\leq\tilde{r}}$, and

(ii) $f_r(S_k^{(r)}) - f_r(\tilde{R}_{S_k^{(r)}}(\beta^m\bar{\alpha}\eta^*)) \geq \sigma \left\langle -\operatorname{grad} f_F(S_k^{(r)}), \beta^m\bar{\alpha}\eta^* \right\rangle_{S_k^{(r)}}$;

Return $t^* \leftarrow \beta^m\bar{\alpha}$;

Algorithm 5 Adaptive Rank Increase Strategy [Zho15, ZHG+16]

Input: $S_k^{(r)} \in \mathcal{S}_+(n,r)$; $\delta > 0$, $\epsilon_1, \epsilon_3 > 0$, $\epsilon_2 \geq 0$, $c_A, \tau_\epsilon, \tau_\delta \in (0,1)$; maximum possible rank $p_{\max}$;
Output: $\tilde{S}_{k+1}^{(p)} \in \mathcal{S}_+(n,p)$; threshold parameters δ, ϵ_3;

Take thin singular value decomposition for $S_k^{(r)}$, i.e., $S_k^{(r)} = U\operatorname{Diag}(\sigma_1,\ldots,\sigma_r)U^T$, where $U \in St(r,n)$ and $\sigma_1 \geq \cdots \geq \sigma_r \geq 0$;
if $\|\operatorname{grad} f_F(S_k^{(r)}) - \operatorname{grad} f_r(S_k^{(r)})\| > \max\{\epsilon_1\|\operatorname{grad} f_r(S_k^{(r)})\|, \epsilon_2\}$ and $r < p_{\max}$ **then**
  $\tilde{r} \leftarrow r$; $\eta^* \leftarrow -\operatorname{grad} f_r(S_k^{(r)})$; choose $\epsilon_4 < \epsilon_1$;
  while $\|-\operatorname{grad} f_F(S_k^{(r)}) - \eta^*\| > \epsilon_4\|\eta^*\|$ and $\tilde{r} < p_{max}$ **do**
    $\tilde{r} \leftarrow \tilde{r} + 1$; choose $\eta^* \in \arg\min_{\eta\in \mathrm{T}_{\mathcal{D}_{\leq\tilde{r}}}} \|-\operatorname{grad} f_F(S_k^{(r)}) - \eta\|$;
  end while
  Select $\tilde{S}_{k+1}^{(\tilde{r})} \in \mathcal{D}_{\leq\tilde{r}}$ such that $f(S_k^{(r)}) - f(\tilde{S}_{k+1}^{(\tilde{r})}) \geq c_A(f(S_k^{(r)}) - f(\tilde{R}_{S_k^{(r)}}(t^*\eta^*)))$, where t^* is the rank-related Armijo step size returned by Algorithm 4;
  $p \leftarrow \operatorname{rank}(\tilde{S}_{k+1}^{(\tilde{r})})$;
else
  $\epsilon_3 \leftarrow \tau_\epsilon\epsilon_3$ and $\delta \leftarrow \tau_\delta\delta$;
end if

7.6 Riemannian Rank-Adaptive Role Extraction Algorithm

Algorithm 6 summarizes our Riemannian rank-adaptive role extraction approach. For our algorithm, we suggest starting from an initial rank larger than the expected number of roles since much of the information for the similarity measure is contained in the neighborhood patterns of length 1.

The main part of Algorithm 6 is determining when to increase or decrease the rank of the similarity matrix such that we have enough information about the network to extract the role structure. In practice, the gap in the singular values of the similarity matrix is not that large (with

Algorithm 6 Riemannian Rank-Adaptive Role Extraction Method (RRAREM)

Input: $X_1^{(r)} \in \mathbb{R}^{n\times r}$ a representation of initial point $\pi(X_1^{(r)})$ for F_r; stopping criterion threshold ϵ and ϵ_3; rank reducing threshold δ; a Riemannian optimization method; $\tau_\delta, \tau_\epsilon \in (0,1)$;

Output: Y^*; role partition c^*; number of roles q^*;

Set initial number of roles $q_1 \leftarrow r$ and $\hat{X}_1^{(r)} \leftarrow X_1^{(r)}$;
for $k = 1,\ 2, \ldots,$ **do**
  Apply a Riemannian optimization method to maximize F_r over $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$ with initial point $\pi(\hat{X}_k^{(r)})$ until i-th iterate $X_i^{(r)}$ satisfies $g(\operatorname{grad} F_r, \operatorname{grad} F_r) < \epsilon_3^2$ ($rank\_dec\_flag \leftarrow 0$) or the requirement of reducing rank with threshold δ ($rank\_dec\_flag \leftarrow 1$);
  if $r > 1$ **then**
    Apply Algorithm 2 on $X_k^{(r)}$ using r and q_k as bounds for the list of assumed number of roles to determine the updated role partition c_{k+1} and number of roles q_{k+1};
  else
    Set $rank\_dec\_flag \leftarrow 0$;
  end if
  if $g(\operatorname{grad} F_r, \operatorname{grad} F_r) < \epsilon^2$ and $r \geq q_{k+1}$ **then**
    Find a minimizer $Y^* = Y_i$ over $\mathbb{R}_*^{n\times r}/\mathcal{O}_r$;
  else
    if $r \leq q_{k+1}$ and $rank\_dec\_flag = 0$ **then**
      Apply Algorithm 5 to increase rank and obtain an output factors $(\hat{U}, \hat{D})$ on the embedded submanifold representation of $\mathcal{S}_+(n, \tilde{r})$;
      $r \leftarrow \tilde{r}$; Set $X_{k+1}^{(r)} = \hat{U}\operatorname{diag}(\sqrt{\hat{D}})$;
    else
      Apply Algorithm 3 with threshold δ to reduce the rank and obtain an output $\hat{Y} \in \mathbb{R}^{n\times\tilde{r}}$;
      $r \leftarrow \tilde{r}$; set $X_{k+1}^{(r)} \leftarrow \hat{Y}$; $\epsilon_3 \leftarrow \tau_r\epsilon_3$;
      if $\delta < \epsilon_\delta$ **then**
        $\delta \leftarrow \delta_0$;
      else
        $\delta \leftarrow \tau_\delta\delta$;
      end if
    end if
  end if
end for

an exception for the ideal case where the nodes are structurally equivalent), hence, it is difficult to determine when to increase or decrease the rank using generic methods, such as Zhou et al.'s

rank-adaptive method [Zho15, ZHG+16]. Therefore, we use heuristics based on our knowledge of the role extraction problem, the current rank r and number of roles q_{k+1} to determine if we should increase or decrease the rank.

Since we have seen that we are more likely to extract the role structure when the rank is greater than the number of roles, if we find a low-rank solution before convergence where $r \leq q_k$, then we apply our rank increasing strategy (Algorithm 5). However, recall that in Algorithm 5 the rank only increases if the angle and difference between the full gradient and local gradient is larger than some parameter. If this is not the case, then we reduce the convergence parameter ϵ_3 and the rank reducing parameter δ by some factor and go to the role extraction step. This case happens in situations where we have reached a low-rank solution where the rank is equal to the number of roles. Therefore, the heuristic prevents us from accidentally reducing the rank to a solution where we will be unable to extract the role structure from and the rigorous gradient check prevents us from accidentally increasing the rank when we have reached a low-rank solution of the similarity matrix.

If the rank is greater than the number of roles, we apply our rank reduction strategy (Algorithm 3). This allows us to reduce the rank of the solution and narrow down the assumed number of roles we have to test for k-means clustering, since the role extraction steps is typically the most time consuming step in the algorithm due to the fact we have to test and evaluate several different partitions of the network. The silhouette statistic helps us determine the optimal number of roles.

The main problem with Algorithm 6 is that the rank increasing strategy requires the full gradient, which is $O(n^2 r_0 + n^2 r)$ in complexity, where r_0 is the rank of the initial condition. Future work is required to improve the efficiency of line 13 in Algorithm 5 by avoiding computing the full gradient.

7.7 Evidence of Robustness of the One-Phase Approach Compared to the Two-Phase Approach

One of the main issues with our two-phase approach is that it depends on the rank chosen by the user. If the rank is assumed to be less than the number of roles, then the two-phase approach will not extract the role partition since we only test role assignments from 2 to $r+1$. If r is chosen to be greater than the true rank of the similarity measure (which is equal to the number of roles in

the noiseless graph case), then the matrix X^TX in the Riemannian objects defined in Section 4.1.2 may be singular; hence affecting the numeric of our approach when computing the inverse of the matrix. Our two-phase approach allows for the rank to decrease if the matrix X^TX is close to singular, but it does not allow for the rank to increase. Therefore, we need to be able to adjust the rank of the similarity matrix if we overestimate, or underestimate, the rank with respect to the number of roles.

In Table 7.1, we generate unweighted, random Erdós-Rényi graphs that have the same 3 role structures given in Figure 6.1, i.e., block cycle role structure (role graph (a)) and almost isomorphic role structure (role graph (b)), where there are 100 nodes in each role. In the table, we show the average total role extraction time ($RMP\ time$) and NMI values for 20 random realizations for p_{in} and p_{out} probability pairs $(1, 0)$, $(0.9, 0.1)$, and $(0.7, 0.3)$. We increase the initial rank r_0 (or fixed rank for the two-phase algorithm) from 1 to 5. For RRAREM, we set the maximum rank for the algorithm to 5. In Table 7.1, if the algorithm fails to extract a role structure, we denote the number of roles ($\#ofRoles$) by $\sim$. r_f denotes the final rank of the algorithm.

We consider RNewton for both the two-phase approach and the Riemannian update in RRAREM. The algorithm parameters for the two-phase approach are the same as in Section 6.1. For our one-phase approach, the initial rank reducing parameter δ in the rank-update step is 0.5 and we reduce the parameter by a factor of $\tau_\delta = 0.1$ to account for potential noise within the graphs. We reset parameter δ to 0.5 when it is less than 10^{-8}. The rank-increasing parameters ϵ_1, ϵ_2, and ϵ_4, are $\sqrt{5}$, 10^{-2}, and $0.5\epsilon_1$, respectively. Note that we set the angle threshold parameter ϵ_1 to be larger than in [Zho15, ZHG+16], where it was $\sqrt{3}$. This choice is due to the fact that the solutions our algorithms find are not the optimal low-rank solution; they are projected onto a rank that is good enough to extract the role structure. So, we want to be able to increase the rank if we have reduced the rank too far, but the solution we want may never give us a small angle between the full and local gradient in Algorithm 5.

Lastly, the initial stopping criterion value ϵ_3 is 1 and we reduce it by a factor of 0.1 until the norm of the gradient is less than 10^{-6}. The maximum number of outer iterations for RRAREM is 100 and the maximum number of iterations for the local Riemannian optimization problem (line 3 in Algorithm 6) is 2000.

Table 7.1: Results for the role graphs in Figure 6.1. The subscript ν indicates a scale of 10^{ν}.

Graph	(p_{in}, p_{out})	Algorithm	r_0	1			2			3			4			5		
				β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
(a)	$(1,0)$	Two-Phase RNewton	*RMP time*	3.37_{-2}	3.36_{-2}	3.57_{-2}	6.33_{-2}	6.71_{-2}	6.67_{-2}	8.12_{-2}	8.19_{-2}	8.16_{-2}	1.26_{-1}	1.09_{-1}	1.10_{-1}	1.37_{-1}	1.38_{-1}	1.42_{-1}
			NMI	5.97_{-1}	6.20_{-1}	5.89_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	2	2	2	3	3	3	3	3	3	3	3	3
			#ofRoles	2	2	$2, \sim$	3	3	3	3	3	3	3	3	3	3	3	3
		RRAREM RNewton	*RMP time*	4.66_{-1}	5.86_{-1}	4.34_{-1}	3.90_{-1}	4.05_{-1}	4.27_{-1}	2.20_{-1}	1.49_{-1}	2.12_{-1}	1.05	9.79_{-1}	1.05	1.29	1.22	1.32
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
			#ofRoles	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
	$(0.9, 0.1)$	Two-Phase RNewton	*RMP time*	3.56_{-2}	4.03_{-2}	3.94_{-2}	6.14_{-2}	7.08_{-2}	6.92_{-2}	9.21_{-2}	1.08_{-1}	9.91_{-2}	1.35_{-1}	1.44_{-1}	1.69_{-1}	1.83_{-1}	1.94_{-1}	2.65_{-1}
			NMI	4.97_{-3}	4.58_{-3}	5.41_{-3}	8.80_{-1}	8.80_{-1}	8.80_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
			#ofRoles	2	2	2	2, 3	2, 3	2, 3	3	3	3	3	3	3	3	3	3
		RRAREM RNewton	*RMP time*	4.43	4.46	8.33	9.91_{-1}	9.77_{-1}	8.80_{-1}	1.04	9.09_{-1}	1.02	6.65_{-1}	6.35_{-1}	6.73_{-1}	6.42_{-1}	6.84_{-1}	8.09_{-1}
			NMI	1.04_{-2}	8.43_{-3}	8.16_{-3}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	3, 4, 5	4, 5	4, 5	5	4, 5	4, 5	5	4, 5	4, 5	4, 5	4, 5	4, 5	4, 5	5	5
			#ofRoles	2, 3, 5, 6, 7	2, 3, 4, 5, 7	2, 3, 5, 6, 7, 12	3	3	3	3	3	3	3	3	3	3	3	3
	$(0.7, 0.3)$	Two-Phase RNewton	*RMP time*	3.42_{-2}	3.81_{-2}	3.78_{-2}	6.29_{-2}	6.91_{-2}	7.55_{-2}	8.97_{-2}	1.01_{-1}	1.01_{-1}	1.17_{-1}	1.31_{-1}	1.49_{-1}	1.56_{-1}	1.69_{-1}	2.25_{-1}
			NMI	8.16_{-3}	4.16_{-3}	4.30_{-3}	7.99_{-1}	7.99_{-1}	7.99_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
			#ofRoles	2	2	2	2, 3	2, 3	2, 3	3	3	3	3	3	3	3	3	3
		RRAREM RNewton	*RMP time*	4.43	4.16	3.75	7.31_{-1}	9.59_{-1}	8.87_{-1}	1.21	7.20_{-1}	1.05	5.58_{-1}	6.09_{-1}	6.30_{-1}	5.93_{-1}	6.09_{-1}	7.06_{-1}
			NMI	4.87_{-3}	5.19_{-3}	5.12_{-3}	8.00_{-1}	8.00_{-1}	7.60_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	3, 4, 5	3, 4, 5	3, 4, 5	3, 4, 5	3, 4, 5	3, 4, 5	3, 5	4, 5	4	4, 5	4, 5	5	5	5	4, 5
			#ofRoles	2, 3, 7, 9	2, 3, 8	2, 3, 5, 6, 7	2, 3	2, 3	2, 3	3	3	3	3	3	3	3	3	3
(b)	$(1,0)$	Two-Phase RNewton	*RMP time*	2.97_{-2}	2.86_{-2}	2.79_{-2}	5.30_{-2}	5.16_{-2}	5.25_{-2}	7.95_{-2}	7.68_{-2}	7.62_{-2}	1.07_{-1}	1.04_{-1}	1.03_{-1}	1.35_{-1}	1.32_{-1}	1.31_{-1}
			NMI	3.61_{-1}	3.63_{-1}	3.61_{-1}	7.34_{-1}	7.34_{-1}	7.34_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	2	2	2	3	3	3	3	3	3	3	3	3
			#ofRoles	2	2	2	2	2	2	3	3	3	3	3	3	3	3	3
		RRAREM RNewton	*RMP time*	1.31_{-2}	1.11_{-2}	1.28_{-2}	6.93_{-1}	4.43_{-1}	4.24_{-1}	2.19_{-1}	1.46_{-1}	1.44_{-1}	1.05	9.83_{-1}	9.72_{-1}	1.28	1.21	1.20
			NMI	1.38_{-15}	1.38_{-15}	1.38_{-15}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	3	3	3	3	3	3	3	3	3	3	3	3
			#ofRoles	1	1	1	3	3	3	3	3	3	3	3	3	3	3	3
	$(0.9, 0.1)$	Two-Phase RNewton	*RMP time*	3.35_{-2}	3.75_{-2}	3.93_{-2}	6.31_{-2}	6.99_{-2}	7.07_{-2}	9.65_{-2}	1.00_{-1}	1.08_{-1}	1.46_{-1}	1.46_{-1}	1.59_{-1}	1.96_{-1}	1.96_{-1}	2.12_{-1}
			NMI	7.68_{-3}	1.06_{-2}	8.09_{-3}	7.34_{-1}	7.34_{-1}	7.34_{-1}	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
			#ofRoles	2	2	2	2	2	2	3	3	3	3	3	3	3	3	3
		RRAREM RNewton	*RMP time*	9.27_{-1}	9.36_{-1}	9.63_{-1}	1.03	1.03	1.01	9.14_{-1}	1.04	9.42_{-1}	8.14_{-1}	8.29_{-1}	8.72_{-1}	7.54_{-1}	7.96_{-1}	8.58_{-1}
			NMI	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
			r_f	4, 5	4, 5	4, 5	4, 5	4, 5	4, 5	5	4, 5	5	4, 5	4, 5	4, 5	4, 5	5	4, 5
			#ofRoles	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
	$(0.7, 0.3)$	Two-Phase RNewton	*RMP time*	3.63_{-2}	4.31_{-2}	3.38_{-2}	6.19_{-2}	8.05_{-2}	7.34_{-2}	9.97_{-2}	1.09_{-1}	1.02_{-1}	1.36_{-1}	1.48_{-1}	1.38_{-1}	1.84_{-1}	1.90_{-1}	1.82_{-1}
			NMI	9.48_{-3}	8.96_{-3}	1.07_{-2}	7.34_{-1}	7.34_{-1}	7.34_{-1}	9.98_{-1}	9.97_{-1}	9.89_{-1}	9.98_{-1}	9.97_{-1}	9.89_{-1}	7.47_{-1}	9.83_{-1}	9.89_{-1}
			r_f	1	1	1	2	2	2	3	3	3	4	4	4	5	5	5
			#ofRoles	2	2	2	2	2	2	3	3	3	3	3	3	2, 3	2, 3	3
		RRAREM RNewton	*RMP time*	9.47_{-1}	9.85_{-1}	1.03	6.66_{-1}	5.51_{-1}	9.78_{-1}	8.34_{-1}	7.57_{-1}	1.03	6.41_{-1}	6.46_{-1}	6.30_{-1}	7.03_{-1}	6.70_{-1}	6.78_{-1}
			NMI	7.08_{-1}	7.42_{-1}	7.65_{-1}	7.34_{-1}	7.47_{-1}	9.70_{-1}	9.98_{-1}	9.97_{-1}	9.89_{-1}	9.98_{-1}	9.97_{-1}	9.89_{-1}	7.47_{-1}	9.83_{-1}	9.89_{-1}
			r_f	3, 4, 5	4	4, 5	3	3, 5	4, 5	4, 5	4, 5	4, 5	4, 5	4	4, 5	5	4, 5	4, 5
			#ofRoles	2, 3	2, 3	2, 3	2	2, 3	3	3	3	3	3	3	3	2, 3	2, 3	3

For role graph (a) when $(p_{in}, p_{out}) = (1, 0)$, both the two-phase and one-phase approaches are able to extract the role partition when r is greater than or equal to the number of roles, i.e., $r \geq 3$. However, RRAREM is also able to extract the 3 role block cycle partition for all three β's when $r_0 = 2$. When $r_0 = 1$, RRAREM extracts the role partition for every graph for β_1 and β_2, but fails to extract the three role partition for one graph when $\beta = \beta_3$. The inconsistency of extracting the correct role partition is probably due to the fact that a rank-1 representation of the neighborhood patterns of length 1 is missing information about the similarity matrix, hence making it difficult to the to extract the exact role partition. Note that for $(p_{in}, p_{out}) = (0.9, 0.1)$ and $(0.7, 0.3)$ that RRAREM still fails to extract the role partition for $r_0 = 1$ and all three β's. When $(p_{in}, p_{out}) = (0.9, 0.1)$ and $(0.7, 0.3)$ for all three β values, the two-phase approach is able to extract the role partition when $r \geq 3$ since the rank of the similarity matrix is not 3, but there still exists a large enough gap in the singular values such that k-means clustering with the silhouette statistic is able to extract the 3 role partition in the second phase of the approach. When $(p_{in}, p_{out}) = (0.9, 0.1)$, RRAREM extracts the exact role partition for $r_0 \geq 2$. However, for $(p_{in}, p_{out}) = (0.7, 0.3)$, RRAREM is only able to extract the role partition for $r_0 \geq 3$. RRAREM fails to extract the 3 role partition for most of the graphs when $r_0 = 2$ because there exists too much noise within the graphs that the algorithm is unable to recover enough information within the rank adjustment step to distinguish the 3 roles. However, when r_0 is assumed to be greater than or equal to the number of roles, there exists enough information from the initial condition to adjust the ranks of the longer neighborhood patterns and extract the exact role partition.

Similar to the previous role graph, the two-phase approach is able to extract the role partition for role graph (b) when $(p_{in}, p_{out}) = (1, 0)$ and $(0.9, 0.1)$ when $r \geq 3$. For $(p_{in}, p_{out}) = (0.7, 0.3)$, the two-phase approach struggles to extract the role partition for every graph for $r \geq 3$, but it finds an alternative 3 role partition that has the same role structure as role graph (b). For this example, due to the noise within the graphs, what was previously assumed to be the true role partition may not be the role partition that best represents the network. For $r = 5$ and $\beta = \beta_1$, the two-phase approach determines a 2 role partition. However, recall for noisy graphs, longer neighborhood patterns are necessary to distinguish between the roles in the role structure since two of the roles are almost isomorphic to each other.

For role graph (b), RRAREM extracts the 3 role partition for $r_0 \geq 2$ when $(p_{in}, p_{out}) = (1, 0)$ and for $r_0 \geq 1$ when $(p_{in}, p_{out}) = (0.9, 0.1)$. Unlike the previous role graph, RRAREM is unable to find the 3 role partition for role graph (b) when $r_0 = 1$ and $(p_{in}, p_{out}) = (1, 0)$. It fails for this case because the initial iterate satisfied the stopping criterion. That is, the norm of the gradient is less than the final tolerance and the number of roles is equal to the rank. Thus, RRAREM returns a role partition with one role. When some noise is added to the network, RRAREM may extract the role partition; however, the unpredictability of the algorithm does not make $r_0 = 1$ an ideal initial rank choice for RRAREM. When $(p_{in}, p_{out}) = (0.7, 0.3)$, the behavior of RRAREM is similar to the two-phase approach for $r_0 \geq 2$.

While RRAREM is able to compete with the two-phase approach in quality for role graphs (a) and (b), it cannot compete in time, except for instances when the two-phase approach fails to extract the role partition. A way to improve the computational time of RRAREM is to avoid the computation of the full gradient in the rank increasing algorithm (Algorithm 5). Computing the full gradient is expensive, and an alternative approach that avoids this computation by either approximating the full gradient or exploiting properties of symmetric positive semidefinite matrices and the Euclidean inner product needs to be explored.

Note that there is not a significance difference in time between the three β parameters because both approaches are dominated in computational time by the role extraction step, which is not dependent upon β. Therefore, overall improvement in computation time to both approaches would be to improve the efficiency the role extraction step.

Overall, RRAREM is able to compete with our two-phase approach in quality when the initial rank is assumed to be greater than or equal to the number of roles, and superior if the initial rank is assumed to be less than the number of roles because it allows for recoverability. In practice, the true rank of the similarity matrix with respect to the number of roles in the network is unknown. Therefore, one should implement the two-phase approach for several ranks to determine the role partition of the graph. This systematic search may be expensive in time, especially when n or r are large. Therefore, the rank-adaptive techniques in RRAREM provides a more sophisticated approach to approximating the rank and finding the number of roles.

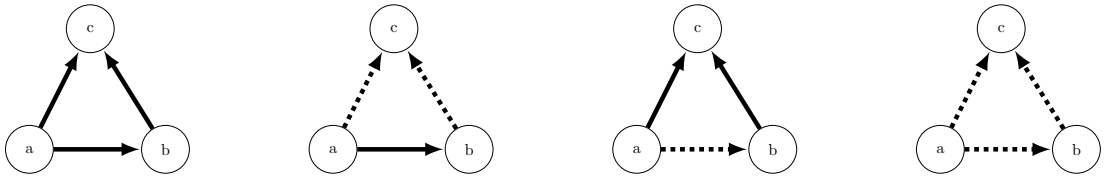
CHAPTER 8

SIGNED NETWORK EXPERIMENTS FOR TWO-PHASE INDIRECT APPROACH

In this chapter we investigate our two-phase indirect approach on the role graphs of signed graphs.

8.1 Introduction

Networks of interest often have both positive and negative weights, e.g., social analysis networks [WF94, DBF05, Tra14], where negative edges often denote a dislike towards a person, place or thing. Partitioning signed networks can be determined by social balance theory (also known as structural balance theory) [WF94, DBF05, Tra14]. First formulated by Heider in [Hei46, Hei58] and then generalized by Cartwright and Haray in [CH56], given a triad, i.e., a cycle of length 3, social balance theory predicts the third edge of the triangle such that it is balanced, i.e., it does not cause the system to change. For example, given three nodes a, b, and c, then if c is a friend of b who is a friend of a, then c is a friend of a (Figure 8.1a). Similarly, in Figure 8.1b, if c is an enemy of b who is a friend of a, then c is an enemy of a [DBF05]. However, an imbalanced system would be if both a and b are friends with c, but b is an enemy of a (Figure 8.1c), or an enemy (c) of my enemy (b) is my enemy (a) (Figure 8.1d). These two examples are imbalanced because they cause strain on the system. Several more examples of balanced and imbalanced triads are in [WF94, DBF05].



(a) Balanced (b) Balanced (c) Imbalanced (d) Imbalanced

Figure 8.1: Balanced (left) and imbalanced (right) signed triads, where dashed lines indicate negative edges and solid lines indicate positive edges.

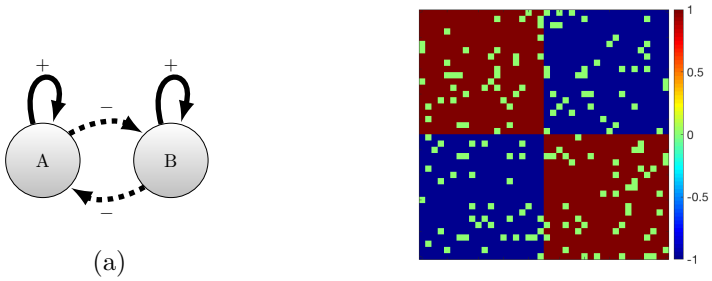
Recall that a signed graph is denoted $G(V, E^-, E^+)$, where $E^- \subseteq V \times V$ are the negative edges, $E^+ \subseteq V \times V$ are the positive edges, and no edge can be both positive and negative (i.e., $E^- \cap E^+ = \emptyset$). The definitions of walks, paths, cycles, and semiwalks are the same for signed graphs as they are for unsigned graphs (see Section 1.2), except that the edges now include signs. The sign of a (semi)walk is the product of signs of the lines in the sequence. Thus, the (semi)walk on signed networks is positive if and only if it has an even number of negative edges; otherwise it is negative.

In Figure 8.1, the balanced triads are positive, while the imbalanced triads are negative. Therefore, a signed network is balanced if and only if the sets of nodes can be partitioned into two clusters such that every positive edge joins nodes of the same cluster and every negative edge joins nodes from difference clusters [DBF05, Chapter 10]. For example, Figure 8.2 is a balanced signed network partitioned such that the positive blocks have only positive or zero entries and are on the block diagonal while the negative blocks have negative or zero entries and are off-diagonal blocks. That is, balanced signed networks can be determined by the following theorem.

Theorem 8.1.1 *[DBF05, Theorem 10.1] A signed network $G(V, E^-, E^+)$ is balanced if and only if every closed semiwalk is positive.*

In the next section, we investigate if our indirect role extraction methods can find role structures in signed networks that conform to balance theory. That is, we want to find well-defined role structures where every positive edge joins nodes in the same role and every negative edge joins nodes from different roles.

In addition to partitioning the network appropriately for positive and negative edges, we investigate if our indirect methods can extract a positive subgraph within the larger signed network (Figure 8.3). Being able to extract smaller subgraphs within larger signed graphs is of interest in examples where people are interested in isolating positive, or negative, characteristics of the network. For example, Ayroles et al. in [ACS+09] computed the correlation of $10,096$ genetic transcripts of 40 inbred lines of *Drosophila melanogaster* and used community detection methods to cluster the correlation of the transcripts into 241 communities. They isolated positive correlation subsets of data related to a specific traits and observed any biological significance and overlap between genes within those traits. While their analysis focused on community structures and did not consider role



(a)

(b)

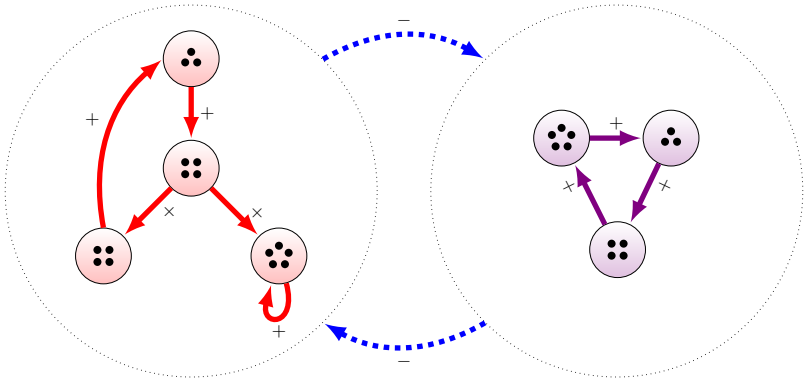
Figure 8.2: Example of block structure of signed networks.

structures, it inspired the idea of there possibly existing role subgraphs within a larger role graph of a signed network.

8.2 Experiments

To construct the signed graphs, we assume there exist two separate role subgraphs such that each role subgraph has positive edges within the subgraph and negative edges between the subgraph (see Figure 8.3a). The role signed role graph is denoted by $G_B(V_B, E_B^+, E_B^-)$, where $E_B^+ \subseteq V_B \times V_B$ is the set of positive edges in G_B and E_B^- is the set of negative edges in G_B. In Figure 8.3a, the red positive edges represent the 4 role structure in the network and the purple edges represent the block cycle role structure. The dashed blue edges are the negative edges between the two structures. The adjacency matrix of Figure 8.3a is shown in Figure 8.3c, where the size n is fixed at 1700 and the sizes of the roles in the first structure are 200, 300, 100, and 500, and the size of the roles in the second role structure are 100, 100, and 400. The edges of the adjacency matrix are unweighted.

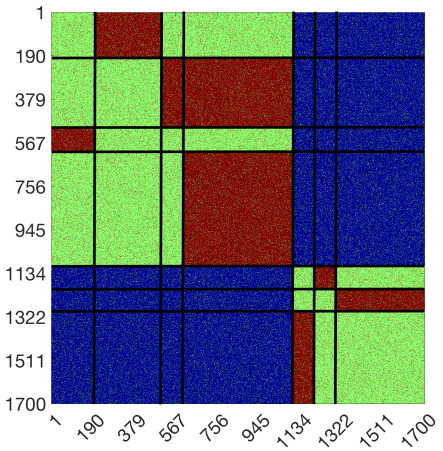
We build our signed graph $G_A(V_A, E_A^+, E_A^-)$ by adding edges to E_B^+ and E_B^- according to the probability parameters p_{in} and p_{out}, where the negative edges in the role graph are denoted by negative edges in the image matrix (Figure 8.3b). That is, for every pair of nodes $i, j \in V_A$, a positive edge $(i, j) \in E_A^+$ is added with probability p_{in} if a positive edge between the corresponding roles exists in G_B. Similarly, a negative edge $(i, j) \in E_A^-$ is added with probability p_{in} if a negative edge between the corresponding roles exists in G_B. If a positive (or negative) edges does not exist



(a)

$$B = \begin{bmatrix} 0 & 1 & 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 1 & 1 & -1 & -1 & -1 \\ 1 & 0 & 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & 0 & 1 & 0 \\ -1 & -1 & -1 & -1 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & 1 & 0 & 0 \end{bmatrix}$$

(b)



(c)

Figure 8.3: Signed Erdós-Rényi graph with two role subgraphs, where there exists negative edges between the subgraphs and positive edges within the subgraphs.

between the corresponding roles, then the positive (or negative) edge is added with probability p_{out}. We consider two p_{in} and p_{out} pairs: $(0.9, 0.1)$ and $(0.7, 0.3)$.

We extract the role structures using our two-phase approach, and compare the quality and time. We consider three β values given by (6.11). Since $G_A(V_A, E_A^+, E_A^-)$ has negative edges, then, in general, β_1 may not be less than β_2 and β_3 since the left-hand side of the inequality (5.18) in Theorem 5.2.1 may not be true. However, all three β values are less than the upper bound in (5.18). That is, the similarity metric is well-defined for all three β values.

As in Section 6.1, we compare our structures when starting from initial ranks 1 to 15. Table 8.1 summarizes the average time it took to complete the role extraction process once and the average

NMI results out of 5 runs.

We have omitted Browet and Van Dooren's approach from this experiment because the community detection cost functions within their algorithm are not designed to handle negative edge weights. Modifications for negative edge weights could be made to the code using ideas in [Tra14].

From Table 8.1, observe for $r = 7$, our two-phase role extraction approach is able to extract the role structure for the signed network for all three β values when $(p_{in}, p_{out}) = (0.9, 0.1)$. When more null edges are added to the network, the approach only extracted 5 roles for β_1, and 2 roles for β_2 and β_3. However, the silhouette coefficient indicates that all of the partitions are a good representations of the dataset. The high silhouette value indicates that we were able to extract roles partitions for the graph that were balanced. That is, we have groups of nodes in a role that are joined by a positive edge and every negative edges joins nodes from different roles.

In Figure 8.4, we observe the three different role structures extracted by our two-phase process when $r = 7$ for β_1 for both probability pairs and for β_2 when $(p_{in}, p_{out}) = (0.7, 0.3)$. Notice in Figure 8.4a that we extract the role structures exactly, as indicated by the NMI value of 1.00 in Table 8.1. However, in Figure 8.4c, we obtain a 5 role structure where 4 of the blocks follow our assumed 4 role subgraph and the fifth block of the permuted adjacency matrix is nonnegative and there is no clear grouping of positive edges.

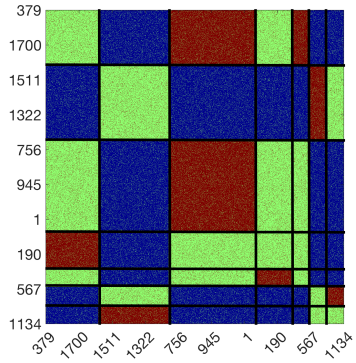
Since we have almost constructed the full role graph of the network, we isolated this nonnegative block from the adjacency matrix and applied our role extraction approach again on the smaller sub-block to see if there is a hidden role structure within the sub-block. This allowed us to extract the small block cycle role structure (see Figure 8.4e) and the NMI value of the new, role structure is 1.00. Similarly, for the 2 role graph in Figures 8.4f and 8.4g, we were able to obtain our two separate role structures (Figure 8.4h) after implementing our role extraction algorithm on each block.

8.3 Summary

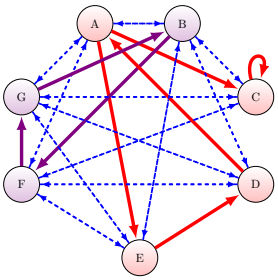
Therefore, for signed networks, our two-phase role extraction algorithm is able to extract role structures that satisfy the ideas of balanced signed network. This is probably because the neighborhood patterns of the similarity matrix are semiwalks with even lengths and the similarity measure

Table 8.1: Time and NMI comparison of signed role graph 8.3. The subscript ν indicates a scale of 10^{ν}.

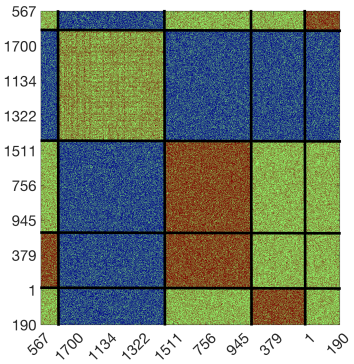
(p_{in}, p_{out})	Algorithm	r	3			6			7			14			15		
			β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3	β_1	β_2	β_3
$(0.9, 0.1)$	Two-Phase LRBFGS IC1	gf	3.86_{-7}	3.95_{-7}	5.51_{-7}	5.28_{-7}	2.49_{-7}	6.51_{-7}	7.89_{-7}	5.18_{-7}	7.65_{-7}	2.16_{-8}	5.15_{-7}	5.37_{-7}	3.03_{-8}	5.39_{-7}	7.18_{-7}
		$RMP\ time$	1.43	2.02	2.63	2.23	2.81	3.22	2.52	3.22	3.60	5.76	6.51	7.42	6.40	7.55	7.74
		$Silh.\ NMI$	5.39_{-1}	5.39_{-1}	8.33_{-1}	1.00	1.00	9.05_{-1}	1.00	1.00	1.00	1.00	9.05_{-1}	9.05_{-1}	1.00	9.05_{-1}	9.05_{-1}
		SC	9.77_{-1}	9.86_{-1}	9.93_{-1}	9.98_{-1}	9.97_{-1}	9.97_{-1}	9.98_{-1}	9.98_{-1}	9.97_{-1}	9.88_{-1}	9.90_{-1}	9.92_{-1}	9.85_{-1}	9.88_{-1}	9.91_{-1}
		$\#ofRoles$	2	2	4	7	7	5	7	7	7	7	5	5	7	5	5
	Two-Phase LRBFGS IC2	gf	3.87_{-7}	3.78_{-7}	9.30_{-7}	5.30_{-7}	2.59_{-7}	6.00_{-7}	7.89_{-7}	5.25_{-7}	8.02_{-7}	2.48_{-8}	5.91_{-7}	4.93_{-7}	8.15_{-7}	5.97_{-7}	6.56_{-7}
		$RMP\ time$	1.43	2.01	2.54	2.23	2.75	3.19	2.51	3.18	3.57	5.87	6.52	7.25	6.36	7.09	8.09
		$Silh.\ NMI$	5.39_{-1}	5.39_{-1}	5.39_{-1}	1.00	1.00	9.05_{-1}	1.00	1.00	1.00	1.00	9.05_{-1}	9.05_{-1}	1.00	9.05_{-1}	9.05_{-1}
		SC	9.77_{-1}	9.86_{-1}	9.91_{-1}	9.98_{-1}	9.97_{-1}	9.97_{-1}	9.98_{-1}	9.98_{-1}	9.97_{-1}	9.88_{-1}	9.90_{-1}	9.93_{-1}	9.87_{-1}	9.90_{-1}	9.92_{-1}
		$\#ofRoles$	2	2	2	7	7	5	7	7	7	7	5	5	7	5	5
$(0.7, 0.3)$	Two-Phase LRBFGS IC1	gf	3.72_{-7}	9.34_{-7}	7.33_{-7}	3.41_{-7}	6.62_{-7}	7.55_{-7}	3.31_{-7}	8.76_{-7}	5.64_{-7}	3.59_{-7}	8.50_{-7}	3.68_{-7}	3.99_{-7}	8.56_{-7}	2.40_{-7}
		$RMP\ time$	1.21	1.62	2.31	2.13	2.89	3.81	2.46	3.05	4.34	5.67	6.47	7.99	6.19	7.01	8.32
		$Silh.\ NMI$	5.39_{-1}	5.39_{-1}	5.39_{-1}	9.05_{-1}	5.39_{-1}	5.39_{-1}	9.05_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}
		SC	9.93_{-1}	9.96_{-1}	9.98_{-1}	9.89_{-1}	9.92_{-1}	9.96_{-1}	9.87_{-1}	9.91_{-1}	9.96_{-1}	9.83_{-1}	9.90_{-1}	9.95_{-1}	9.82_{-1}	9.90_{-1}	9.95_{-1}
		$\#ofRoles$	2	2	2	5	2	2	5	2	2	2	2	2	2	2	2
	Two-Phase LRBFGS IC2	gf	3.50_{-7}	9.59_{-7}	8.03_{-7}	3.58_{-7}	4.43_{-7}	6.35_{-7}	3.30_{-7}	4.18_{-7}	9.90_{-7}	8.77_{-7}	8.11_{-7}	4.02_{-7}	7.87_{-7}	7.02_{-7}	4.06
		$RMP\ time$	1.22	1.60	2.30	2.13	2.59	3.67	2.46	2.96	4.14	5.63	6.27	8.72	6.56	7.13	7.71
		$Silh.\ NMI$	5.39_{-1}	5.39_{-1}	5.39_{-1}	9.05_{-1}	5.39_{-1}	5.39_{-1}	9.05_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}	5.39_{-1}
		SC	9.93_{-1}	9.96_{-1}	9.98_{-1}	9.89_{-1}	9.92_{-1}	9.96_{-1}	9.85_{-1}	9.91_{-1}	9.96_{-1}	9.83_{-1}	9.91_{-1}	9.95_{-1}	9.83_{-1}	9.91_{-1}	9.95_{-1}
		$\#ofRoles$	2	2	2	5	2	2	5	2	2	2	2	2	2	2	2



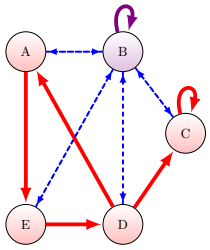
(a) $(0.9, 0.1)$ and β_1: Permuted A



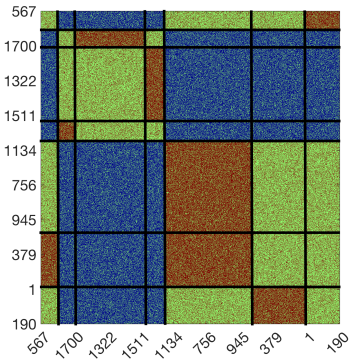
(b) Role graph



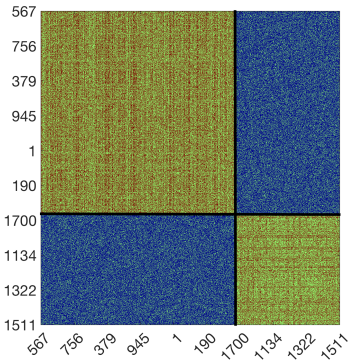
(c) $(0.7, 0.3)$ and β_1: Permuted A



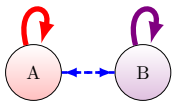
(d) Role graph



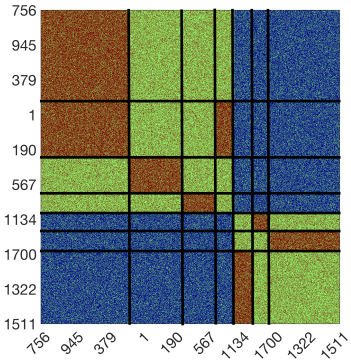
(e) A after implementing algorithm on block 2.



(f) $(0.7, 0.3)$ and β_2: Permuted A



(g) Role graph



(h) A after implementing algorithm on both blocks individually.

Figure 8.4: Results from our two-phase role extraction approach. Left: Permuted adjacency matrix A. Middle: Corresponding role graph. Solid lines indicate positive edges and dashed lines indicate negative edges. Right: Final solution.

may be creating indirectly balanced triads, but a more rigorous analysis of the similarity measure on signed networks is needed.

When trying to find a positive role subgraph within the larger role graph, our approach indicates the need to run an indirect hierarchical role detection algorithm. Our simulated experiments show that we must run the indirect approach twice to find the role subgraphs, where the first implementation of the algorithms partitions the adjacency matrix into positive and negative blocks. If the positive blocks are not dense or do not have a dense sub-block, then the positive block can be isolated from the adjacency matrix and the role extraction methods can be implemented again on the smaller block to see if there exists a role subgraph. Therefore, an indirect hierarchical role detection algorithm may need to be considered for signed networks in case there exists role subgraphs within the larger graph, and we would need to run the hierarchical algorithm at least twice to find the complete role structure. However, in practice, the required number of times the indirect approach should be run is unknown and must be investigated.

CHAPTER 9

OVERLAPPING COMMUNITIES VERSUS ROLES

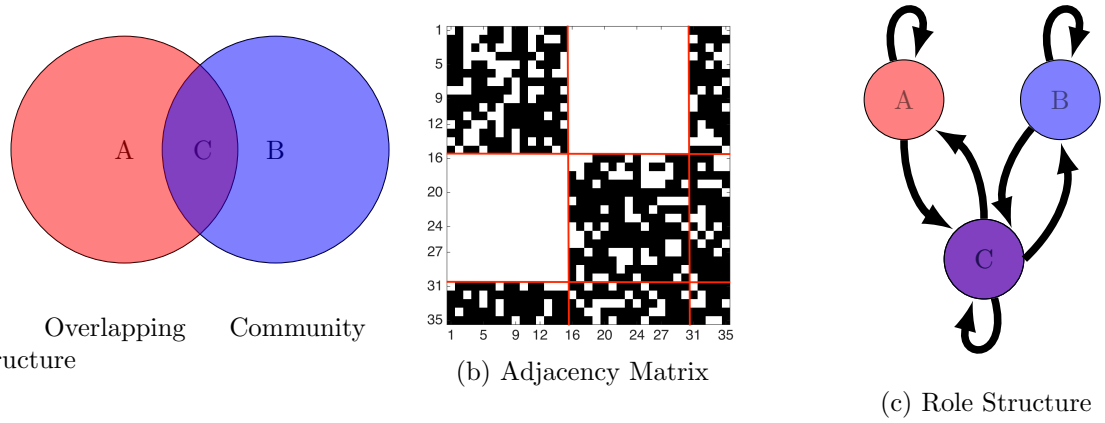
9.1 Introduction

Community detection emphasizes the presence of dependencies inside a group and the absence of dependencies between groups (see Section (2.2)), while role extraction focuses on interdependencies between groups (see Section 1.1). Therefore, community structures are a special case of role structures. However, there may exist nodes that can be placed in multiple communities without significantly altering the value of the quality function optimized. That is, given two separate communities **A** and **B**, a third group of nodes **C** may be included in either **A** or **B** if the quality function fails to determine a significance of one community over the other [PDFV05, Rei09]. Therefore, it can be concluded that **A** and **B** are overlapping communities and **C** is the overlap (see Figure 9.1a).

Palla et al. developed an algorithm to detect overlapping community structures by the clique percolation method [DPV05, PDFV05, APF+06, FPV07, PDV07, PFP+07]. The clique percolation method builds communities from *k-cliques*, i.e., complete subgraphs of k nodes. A *k-clique-community* is the union of all k-cliques that can be reached through a series of adjacent k-cliques, where two k-cliques are adjacent if they share $k-1$ nodes [PDFV05]. The basic definition for k-cliques is only for unweighted and undirected graphs; however, the authors extended the definition of k-cliques to handle directed and weighted graphs [FPV07, PFP+07]. The software, called *CFinder*[1] implements the clique percolation method by Palla et al. for overlapping community structures [APF+06].

A problem with the clique percolation method is that it is dependent upon the existence of triangles in the network. Nodes that are not connected by triangles to communities cannot be a part of the communities. Also, only nodes with at least $k-1$ cliques can be a part of a k-clique. So, single edges may be added or removed from the network [Rei09]. This is a concern for noisy networks since two communities may be connected by a single edge. Lastly, searching for k-cliques

[1] http://www.cfinder.org/



(a) Overlapping Community Structure

(b) Adjacency Matrix

(c) Role Structure

Figure 9.1: Example of two overlapping communities.

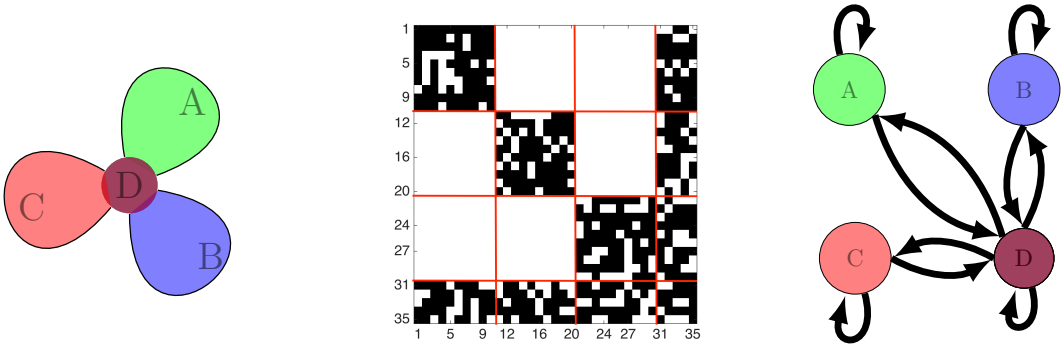
in a network may be slow and costly, especially for dense graphs [DPV05, Rei09]. Hence, the clique percolation method mainly works well for sparse and small graphs.

Observe for the adjacency matrix (Figure 9.1b) that the overlapping community structure in Figure 9.1a can be represented by the 3 role structure in Figure 9.1c. That is, the overlap **C** can be represented by its own role where its role has connections to other nodes within the same role and to nodes in roles **A** and **B**. Also, the nodes in roles **A** and **B** do not interact with each other and only interact with nodes in the same role or with nodes in role **C**. Lastly, the above role structure is a valid role structure since it satisfies our role constraint that no two roles can be structurally equivalent (see Section 1.1). Thus, role extraction algorithms can be used to extract role partitions that are representative of overlapping community structures.

In the next section, we show empirically that the overlapping community structures can be viewed as role structures where the overlap is a role.

9.2 Experiments

In this section, we observe a relationship between role structures and overlapping community structures for synthetic networks. For our first set of experiments, we compare the overlapping community structure found by the clique percolation method (*CFinder*) [APF+06] and the role structure extracted by Browet and Van Dooren's full-rank iterative algorithm for the neighborhood pattern similarity measure and k-means clustering with the silhouette statistic. For the clique



(a) 3 Overlapping Communities where **D** is the overlap

(b) Adjacency Matrix

(c) 4 Roles

Figure 9.2: Example of three overlapping communities.

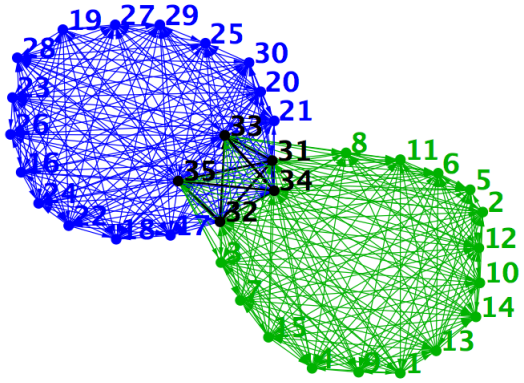
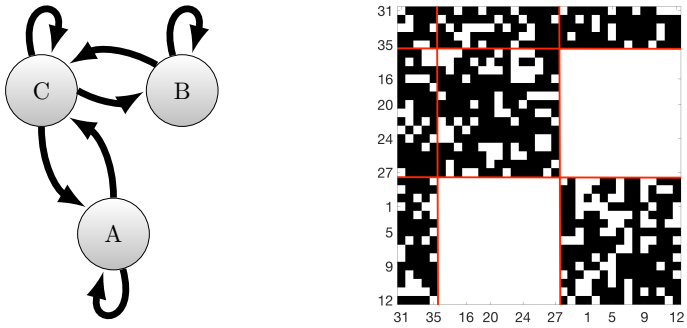


Figure 9.3: Two overlapping communities found by *CFinder* where the black nodes are the overlap.

percolation method, we look at 6-clique communities, and for the silhouette statistic, we run k-means clustering for 2 to 6 clusters and choose the one with the largest silhouette value. We use the unweighted Erdós-Rényi graphs with probability parameters $(p_{in}, p_{out}) = (0.7, 0)$ to generate our synthetic overlapping community examples. We assume two different overlapping community structures: a two overlapping community structure (Figure 9.1) and a three overlapping community structure (Figure 9.2). The size of the adjacency matrix is $n = 35$ for both examples. For the two overlapping community structure, there are 20 nodes in each community, where 5 of the nodes are in the overlap, and for the three overlapping community structure, there are 15 nodes in each community, where all three communities share 5 nodes.

Figure 9.3 is the two overlapping communities determined by *CFinder*. In the figure, nodes 31,



(a) 3 Role Structure

(b) Permuted Adjacency Matrix

Figure 9.4: Three role structure found by role extraction method where role **C** is the overlap.

32, 33, 34, and 35 are the overlapping nodes. This overlapping community structure corresponds with the three role structure found by the role extraction method (Figure 9.6) where role **C** (first block in the adjacency matrix) contains nodes 31, 32, 33, 34, and 35, and these nodes interact with themselves, with nodes in roles **B** (second block in the adjacency matrix), and with nodes in role **A** (third block of the adjacency matrix). Roles **B** and **A** only interact with other nodes in the same role, similar to how the nodes in the blue and green communities in Figure 9.3 are only connected to other nodes in the same community. Therefore, the role extraction method can extract a three role partition that is similar to a two overlapping community structure, where the third role that contains edges to and from the other roles and a self-loop can be viewed as the overlap of the two community structure.

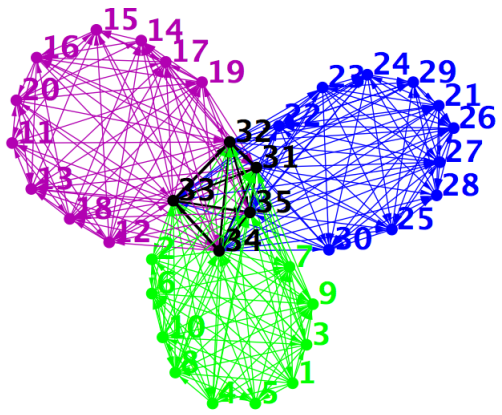
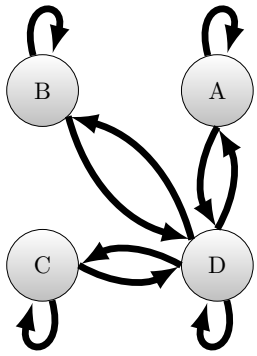
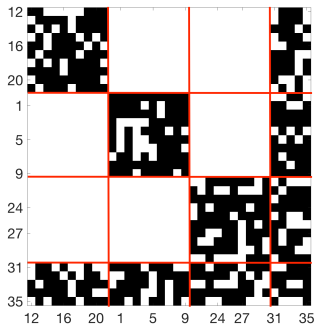


Figure 9.5: Three overlapping communities found by *CFinder* where the black nodes are the overlap.



(a) 4 Role Structure



(b) Permuted Adjacency Matrix

Figure 9.6: Four role structure found by role extraction method where role **C** is the overlap.

Next we consider our three overlapping community structure in Figure 9.2. For this structure, we have three communities **A**, **B**, and **C** that all share 5 nodes that are located in overlap **D**. Figure 9.5 are the results from *CFinder* on this simulated data. Observe that the overlapping community detection algorithm found 3 communities that had 5 overlapping nodes. The role extraction method finds four roles where each role interacts with nodes in the same roles and roles **A**, **B**, and **C** also interact with the 5 nodes in role **D** (Figure 9.6). This structure is equivalent to the overlapping community structure, where role **D** in Figure 9.6 is the overlap in Figure 9.5.

As we increase the number of nodes in the graphs, we can still extract a role structure that is equivalent to an overlapping community structure. For example, consider the three overlapping community structure in Figure 9.2 generated with probability parameter pair $(0.7, 0)$ and 350 nodes, where there are 150 nodes in each community and 50 of those nodes makes up the overlap. The low-rank role extraction methods can extract the 4 role structure in Figure 9.6a.

In Table 9.1, we compare the role partition extracted by the low-rank role extraction algorithms with exact 4 role structure that is equivalent to the overlapping community structure. The table shows that the low-rank algorithms are able to extract the 4 role structure in Figure 9.6a depending upon the assumed rank r and the neighborhood pattern similarity parameter β.

Therefore, overlapping community structures can be viewed as role structures where the overlap of the communities is classified as its own role and we can use role extraction algorithms to find this role structure. For large networks with dense roles, we can use a low-rank role extraction algorithm to find overlapping community structures.

Our simulations on synthetic data support this observation, but a more thorough investigation on empirical datasets must be done. In particular, a survey on the parameters defined for role extraction algorithms applied to overlapping community structures needs to be performed.

Table 9.1: Comparison of NMI values of the low-rank role extraction algorithms for overlapping community role structure and three β values given by (6.11). The subscript ν indicates a scale of 10^{ν}.

	Algorithm	Browet LR CNM		Two-Phase LRBFGS IC1		Two-Phase LRBFGS IC2	
r		$\#\ of\ Roles$	NMI	$\#\ of\ Roles$	NMI	$\#\ of\ Roles$	NMI
2	β_1	2	6.10_{-1}	3	8.88_{-1}	3	8.88_{-1}
	β_2	2	5.85_{-1}	3	8.88_{-1}	3	8.88_{-1}
	β_3	2	5.68_{-1}	3	8.88_{-1}	3	8.88_{-1}
3	β_1	3	7.70_{-1}	4	1.00	4	1.00
	β_2	3	7.70_{-1}	4	1.00	4	1.00
	β_3	3	7.70_{-1}	4	1.00	4	1.00
4	β_1	4	1.00	4	1.00	4	1.00
	β_2	4	1.00	4	1.00	4	1.00
	β_3	4	1.00	4	1.00	4	1.00
10	β_1	4	1.00	4	1.00	4	1.00
	β_2	4	1.00	4	1.00	4	1.00
	β_3	4	1.00	4	1.00	4	1.00
15	β_1	4	1.00	4	1.00	4	1.00
	β_2	4	1.00	4	1.00	4	1.00
	β_3	4	1.00	4	1.00	4	1.00

CHAPTER 10

CONCLUSIONS AND FUTURE RESEARCH

10.1 Conclusion

In this dissertation, we propose a two-phase indirect approach to solving the role extraction problem. The first phase consists of using Riemannian optimization to approximate the low-rank neighborhood pattern similarity measure and the second phased uses k-means clustering with the silhouette statistic on the low-rank factor to extract the role partition of the graph. We also propose a one-phase indirect approach that combines the two phases of our indirect approach that iteratively approximates the best low-rank solution of the similarity measure and determine the role structure of the network.

The major contributions of this dissertation are:

1. **We analyze the rank of the neighborhood pattern similarity measure with respect to the rank of the adjacency matrix of a graph and the number of roles**.

   The idea of using a rank projection of the similarity matrix and what that means for the role extraction problem is a very recent development. Browet and Van Dooren empirically showed that the role graph could be recovered from a low-rank projection of the neighborhood pattern similarity measure. We extend this by theoretically analyzing when the number of roles is equal to the rank. In addition, we analyze, for certain role graph structures, the relationship between the rank of the adjacency matrix, the rank of the similarity matrix and the number of roles. Our work is the first theoretical analysis of this relationship.

2. **We develop a cost function and use Riemannian optimization to approximate the neighborhood pattern similarity measure on the symmetric positive semidefinite fixed-rank manifold for the first phase of the role extraction problem**.

   For the first phase of the indirect approach, we use a similarity measure to measure the node-to-node similarity. Browet and Van Dooren propose in [Bro14, BD14] to use a low-rank projection of the neighborhood pattern similarity measure to measure the similarity between the nodes in a fast and efficient way. We derive a cost function (4.1) from their iterative low-rank similarity algorithm and use Riemannian optimization on the symmetric positive semidefinite fixed-rank manifold to approximate the similarity measure in a robust and efficient manner.

3. **We define new Riemannian objects for the symmetric positive semidefinite fixed-rank manifold to improve the performance of state-of-the-art Riemannian algorithms**.

   Since we assume the networks are large, we use recent developments in Riemannian optimization such as the intrinsic representation of tangent vectors and vector transport by parallelization to improve the efficiency and minimize the storage for the Riemannian approach. However, the Euclidean metric for the quotient manifold representation of the symmetric positive semidefinite fixed-rank manifold cannot be used because the natural basis of the horizontal space with respect to the metric is not a orthonormal basis. This is important to simplify the representation and computation of the intrinsic approach. Thus, we define a new metric for the manifold where the natural basis of the new horizontal space is orthonormal with respect to the metric.

4. **We use of k-means clustering and the silhouette statistic on the low-rank factor to extract the role partition for the second phase of the role extraction problem**.

   Browet and Van Dooren use a fast community detection to extract the role partition from the similarity measure. However, the community detection algorithm requires the full approximate similarity measure, which may be costly for large n and r. Therefore, to avoid computing the full similarity measure, we use k-means clustering with the silhouette statistic on the low-rank factor to extract the role partition. The silhouette statistic allows us to efficiently determine the optimal number of roles for k-means clustering from an index set of assumed roles. Furthermore, we show empirically that this approach for the second phase of the algorithm is efficient in time and robustness for simulated graphs. Also, for the metal world trade data, this approach gives us role partitions that make sense from the context of the problem.

5. **We prove that Browet and Van Dooren's full-rank iterative algorithm is a Euclidean gradient projection method along a projection arc with a fixed stepsize**.

   With respect to the new cost function (4.5), we prove that Browet and Van Dooren's full-rank iterative algorithm is a Euclidean gradient projection method along a projection arc with a fixed stepsize. From a proposition in [Ber99], we prove that their full-rank iterative algorithm converges to a stationary point if the similarity parameter β satisfies bound (5.17).

   In addition, for nonnegative matrices, we prove that the new bound for β is less than necessary and sufficient condition (2.25) that ensures the convergence of Browet and Van Dooren's iterative algorithm, and is greater than Browet and Van Dooren's sufficient bound for β (2.26).

6. **We include a stepsize and Armijo line-search method to the Euclidean gradient projection method along a projection arc, and prove that the choice of the Armijo stepsize is dependent upon the similarity measures parameter β.**

Since Browet and Van Dooren's full-rank iterative algorithm is a Euclidean gradient projection method along a projection arc with a fixed stepsize, we include a stepsize and Armijo line-search method to the optimization method. Then, we prove that the Armijo stepsize is dependent upon the similarity parameter β when β satisfies (5.17).

7. **We develop empirical evidence that our two-phase role extraction approach is more efficient in time and more robust compared to Browet and Van Dooren's role extraction approach**.

   We provide empirical evidence that our proposed two-phase role extraction approach is efficient in time and robust compared to Browet and Van Dooren's low-rank role extraction approach. We show that as we consider longer neighborhood paths for the neighborhood pattern similarity measure, our Riemannian approach is able to approximate a low-rank solution faster than Browet and Van Dooren's low-rank algorithm and the clustering algorithm is able to extract the role partition from the similarity approximation.

   We compare the time and robustness of the silhouette and gap statistics with k-means clustering. In the literature, the gap statistic has been shown to be more robust than the silhouette statistic in clustering; however, the gap statistic is significantly slower than the silhouette statistic. We show empirically that the silhouette statistic is fairly robust for certain types of role graphs and that the ability of the silhouette statistic to extract the role partition depends on characteristics of the set of clusters.

8. **We develop and analyze techniques to form a one-phase indirect approach to the role extraction problem and provide empirical evidence of robustness of the approach than to our two-phase approach**.

   Our two-phase approach is dependent upon the assumed rank, which is a problem if the rank is chosen too large or small compared to the number of roles in the graph. So, we propose to combine our two-phase approach into a one-phase indirect approach and use Riemannian rank-adaptive techniques to adjust the rank of the similarity matrix as we attempt to extract the role partition of the graph. We provide empirical evidence of the robustness of our new approach compared to our two-phase approach.

9. **We provide empirical evidence that our two-phase indirect approach can partition signed networks such that the network is balanced**.

   In the literature, social balance theory is used to partition signed networks such that nodes can be grouped into two clusters such that every positive edges joins nodes in the same cluster and every negative edge joins nodes from different clusters. We generate a signed Erdós-Rényi graph where there exists two positive groups, where the nodes within the group are joined by positive edges and the nodes between the groups are joined by negative edges. Also,

we include a role subgraph structure within each positive block, and apply our two-phase approach to extract the overall role structure.

Most of the time, our two phase-approach extracted a large 2 role structure where the role subgraphs are within each role. We apply our role extraction approach again on both positive sets of data and show that we can extract the role subgraph. This provides empirical evidence that a hierarchical role extraction method is a promising approach to search for role subgraphs on signed networks.

10. **We provide empirical evidence that there exists a relationship between overlapping community structures and role structures**.

    We observe that overlapping communities in a graph can be represented by role structures. That is, the overlap of the separate communities is classified as its own role where the nodes in the role interact with other nodes in the same role and with nodes in each of the separate communities, and the separate communities are roles that only interact with nodes in the same community or with nodes in the overlap.

    We simulate small, random, unweighted, directed graphs that have an overlapping community structure and we compare the results of the clique percolation method to the two-phase indirect role extraction approach using the full-rank similarity matrix to empirically determine if it the overlapping community structure and the role structure are equivalent. Our results indicate a relationship between the two structures.

10.2 Future Research

There are several areas of future research for the role extraction problem that can be explored.

(1) **Improve the quality of our one-phase role extraction algorithm**;

We must investigate the rank-adaptive parameters within our one-phase role extraction algorithm to improve its robustness in quality and time. The parameter selection is currently based on heuristics derived from our analysis of the rank-role relationship in the neighborhood pattern similarity measure and empirical results derived from simulated experiments. A more rigorous analysis of these parameters is needed to improve the quality of the algorithm.

The rank increasing strategy uses the full gradient, which is computationally expensive. To improve the complexity of our approach, we must determine an alternative method for our rank increasing strategy that avoids computing the full gradient. That is, we either find a less expensive way to compute the search direction on the tangent cone without explicitly computing the full gradient, or we replace the rank increasing strategy by Zhou et al [Zho15, ZHG$^+$16] with another rank increasing strategy, such as the one in [HGZ16].

(2) **Extend our preliminary results on signed networks to determine if the neighborhood pattern similarity measure supports known results in social balance theory**;

Typically, mathematical extensions of social balance theory is used to partition signed networks. We showed empirically that we could partition a signed network into its ideal clustering using our two-phase indirect approach. A rigorous analytical explanation of the structure of the neighborhood pattern similarity measure with respect to signed networks must be investigated to determine if the measure supports theorems and conditions in social balance theory.

(3) **Extend our preliminary results on the relationship between overlapping community structures and role structures by applying our role extraction algorithm to real world applications with known overlapping community structures**;

We observed that an overlapping community structure can be represented by a role structure, and tested our hypothesis on two synthetic, unweighted and directed graphs by comparing the overlapping community detection results determined by the clique percolation method with the role extraction results determined by the two-phase indirect method. A more rigorous investigation on real world datasets with known overlapping community structure is needed.

(4) **Compare our role extractions algorithms to other competitive indirect and direct role extraction methods**

For this dissertation, we focused on indirect role extraction methods and compared our proposed two-phase method to the role extraction method proposed by Browet and Van Dooren. A complete survey and comparison of competitive indirect and direct role extraction methods is needed to determine the overall quality of our algorithms.

BIBLIOGRAPHY

[ABG07] P.-A. Absil, C.G. Baker, and K.A. Gallivan, *Trust-region methods on Riemannian manifolds*, Foundations of Computational Mathematics **7** (2007), no. 3, 303–330.

[ACS+09] Julien F Ayroles, Mary Anna Carbone, Eric A Stone, Katherine W Jordan, Richard F Lyman, Michael M Magwire, Stephanie M Rollmann, Laura H Duncan, Faye Lawrence, Robert R H Anholt, and Trudy F C Mackay, *Systems genetics of complex traits in Drosophila melanogaster*, Nat Genet **41** (2009), no. 3, 299 – 307.

[AILH09] Pierre-Antoine Absil, Mariya Ishteva, Lieven De Lathauwer, and Sabine Van Huffel, *A geometric Newton method for Oja's vector field*, Neural Computation **21** (2009), no. 5, 1415–1433.

[AM12] P.-A. Absil and Jéôme Malick, *Projection-like retractions on matrix manifolds*, SIAM Journal on Optimization **22** (2012), no. 1, 135–158.

[AMC10] C. Anteneodo, R. D. Malmgren, and D. R. Chialvo, *Poissonian bursts in e-mail correspondence*, The European Physical Journal B **75** (2010), no. 3, 389–394.

[AMS08] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*, Princeton University Press, Princeton, NJ, 2008.

[AO15] P.-A. Absil and I. V. Oseledets, *Low-rank retractions: a survey and new results*, Computational Optimization and Applications **62** (2015), no. 1, 5–29.

[APF+06] Balzs Adamcsek, Gergely Palla, Ills J. Farkas, Imre Dernyi, and Tams Vicsek, *Cfinder: locating cliques and overlapping modules in biological networks*, Bioinformatics **22** (2006), no. 8, 1021.

[AV07] David Arthur and Sergei Vassilvitskii, *K-means++: The advantages of careful seeding*, Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (Philadelphia, PA, USA), SODA '07, Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[BAD13] Arnaud Browet, P.-A. Absil, and Paul Van Dooren, *Fast community detection using local neighborhood search*, ArXiv e-prints (2013).

[Bak08] C. G. Baker, *Riemannian manifold trust-region methods with applications to eigenproblems*, Ph.D. thesis, Florida State University, Department of Computational Science, 2008.

[BCG+09] Duygu Balcan, Vittoria Colizza, Bruno Gonalves, Hao Hu, Jos J. Ramasco, and Alessandro Vespignani, *Multiscale mobility networks and the spatial spreading of infectious diseases*, Proceedings of the National Academy of Sciences **106** (2009), no. 51, 21484–21489.

[BD14] Arnaud Browet and Paul Van Dooren, *Low-rank similarity measure for role model extraction*, In Proceedings of the 21st International Symposium on Mathematical Theory of Networks and Systems, 2014, pp. 1412 – 1418.

[BDG+06] Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Grke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner, *On modularity – np-completeness and beyond*, 2006.

[BDG+08] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, *On modularity clustering*, IEEE Transactions on Knowledge and Data Engineering **20** (2008), no. 2, 172–188.

[BDVB13] M. Beguerisse-Daz, B. Vangelov, and M. Barahona, *Finding role communities in directed networks using role-based similarity, markov stability and the relaxed minimum spanning tree*, Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE, 2013, pp. 937–940.

[Ber99] D. P. Bertsekas, *Nonlinear programming*, 2nd ed., Athena Scientific, 1999.

[BGH+04] Vincent D. Blondel, Anahí Gajardo, Maureen Heymans, Pierre Senellart, and Paul Van Dooren, *A measure of similarity between graph vertices: Applications to synonym extraction and web searching*, SIAM Review **46** (2004), no. 4, 647–666.

[BGLL08] 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** (2008), no. 10, P10008.

[BM06] Vladimir Batagelj and Andrej Mrvar, *Pajek datasets*, 2006.

[BM08] J. A. Bondy and U. S. R. Murty, *Graph theory*, Springer, 2008.

[BMS10] S. Bonnabel, G. Meyer, and R. Sepulchre, *Adaptive filtering for estimation of a low-rank positive semidefinite matrix*, Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2010) (2010).

[BR05] James Baglama and Lothar Reichel, *Augmented implicitly restarted lanczos bidiagonalization methods*, SIAM Journal on Scientific Computing **27** (2005), no. 1, 19–42.

[Bro14] Arnaud Browet, *Algorithms for community and role detection in networks*, Ph.D. thesis, Université Catholique de Louvain, Department of Mathematical Engineering, 2014.

[BS10] S. Bonnabel and R. Sepulchre, *Rank-preserving geometric means of positive semi-definite matrices*, Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2010) (2010), 209–214.

[BV04] Stephen Boyd and Lieven Vandenberghe, *Convex optimization*, Cambridge University Press, New York, NY, 2004.

[CAD13] T.P. Cason, P.A. Absil, and P. Van Dooren, *Iterative methods for low rank approximation of graph similarity matrices*, Linear Algebra and its Applications **438** (2013), no. 4, 1863 – 1882, 16th ILAS Conference Proceedings, Pisa 2010.

[Cas12] Thomas P. Cason, *Role extraction in networks*, Ph.D. thesis, Université Catholique de Louvain, Department of Mathematical Engineering, 2012.

[CB11] K. Cooper and M. Barahona, *Role-similarity based comparison of directed networks*, ArXiv e-prints (2011).

[CE75] Jeff Cheeger and David G. Ebin, *Comparison theorems in Riemannian geometry*, North-Holland Publishing Co., Amsterdam, The Netherlands, 1975.

[CGT00] Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint, *Trust-region methods*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[CH56] D. Cartwright and F. Harary, *Structural balance: a generalization of heider's theory*, Psychological Review **63** (1956), no. 5, 277 – 293.

[CLVD16] Sibo Cheng, Adissa Laurent, and Paul Van Dooren, *Role model detection using low rank similarity matrix*, techreport, April 2016.

[DBF05] Patrick Doreian, Vladimir Batagelj, and Anuska Ferligoj, *Generalized blockmodeling*, Cambridge University Press, 2005.

[DDGDA05] Leon Danon, Albert Daz-Guilera, Jordi Duch, and Alex Arenas, *Comparing community structure identification*, Journal of Statistical Mechanics: Theory and Experiment **2005** (2005), no. 09, P09008.

[Den12] Damien Denayer, *Modé par rôles de grands graphes*, Master's thesis, Université Catholique de Louvain, Institute of Information and Communtication Technologies, Electronics and Applied Mathematics, 2012.

[DPM03] Jean-Pierre Dedieu, Pierre Priouret, and Gregorio Malajovich, *Newton's method on Riemannian manifolds: Covariant alpha-theory*, IMA J. NUMER. ANAL **23** (2003), 395–419.

[DPV05] Imre Derényi, Gergely Palla, and Tamás Vicsek, *Clique percolation in random networks*, Phys. Rev. Lett. **94** (2005), 160202.

[DYB10] J.-C. Delvenne, S. N. Yaliraki, and M. Barahona, *Stability of graph communities across time scales*, Proceedings of the National Academy of Sciences **107** (2010), no. 29, 12755–12760.

[EAS98] Alan Edelman, TomAs A. Arias, and Steven T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM Journal on Matrix Analysis and Applications **20** (1998), no. 2, 303–353.

[EB94] Martin G. Everett and Stephen P. Borgatti, *Regular equivalence: General theory*, The Journal of Mathematical Sociology **19** (1994), no. 1, 29–52.

[EB96] ———, *Exact colorations of graphs and digraphs*, Social Networks **18** (1996), no. 4, 319 – 331.

[FB07] Santo Fortunato and Marc Barthlemy, *Resolution limit in community detection*, Proceedings of the National Academy of Sciences **104** (2007), no. 1, 36–41.

[For10] Santo Fortunato, *Community detection in graphs*, Physics Reports **486** (2010), no. 3–5, 75 – 174.

[FPV07] Ills Farkas, Dniel bel, Gergely Palla, and Tams Vicsek, *Weighted network modules*, New Journal of Physics **9** (2007), no. 6, 180.

[Gab82] D. Gabay, *Minimizing a differentiable function over a differential manifold*, Journal of Optimization Theory and Applications **37** (1982), no. 2, 177–219.

[GMTA05] R. Guimerà, S. Mossa, A. Turtschi, and L. A. N. Amaral, *The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles*, Proceedings of the National Academy of Sciences **102** (2005), no. 22, 7794 – 7799.

[GN02] M. Girvan and M.E.J. Newman, *Community structure in social and biological networks*, Proceedings of the National Academy of Sciences, vol. 99, 2002, pp. 7821 – 7826.

[GNA05] Roger Guimerà and Luis A. Nunes Amaral, *Functional cartography of complex metabolic networks*, Nature **433** (2005), no. 7028, 895 – 900.

[GSPA04] Roger Guimerà, Marta Sales-Pardo, and Luís A. Nunes Amaral, *Modularity from fluctuations in random graphs and complex networks*, Phys. Rev. E **70** (2004), 025101.

[GSSP+10] R. Guimerà, D. B. Stouffer, M. Sales-Pardo, E. A. Leicht, M. E. J. Newman, and L. A. N. Amaral, *Origin of compartmentalization in food webs*, Ecology **91** (2010), no. 10, 2941–2951.

[GV13] G. H. Golub and C. F. VanLoan, *Matrix computations*, 4th ed., Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, 2013.

[GY05] Jonathan L. Gross and Jay Yellen, *Graph theory and its applications*, 2 ed., Discrete Mathematics and Its Applications, Chapman and Hall/CRC, 2005.

[HAG15] W. Huang, P.-A. Absil, and K. A. Gallivan, *A Riemannian symmetric rank-one trust-region method*, Mathematical Programming **150** (2015), 179–216.

[Hei46] Fritz Heider, *Attitudes and cognitive organization*, The Journal of Psychology **21** (1946), no. 1, 107–112.

[Hei58] ———, *The psychology of interpersonal relations*, New York: Wiley, 1958.

[HGA15] W. Huang, K. A. Gallivan, and P.-A. Absil, *A Broyden class of quasi-Newton methods for Riemannian optimization*, SIAM Journal on Optimization **25** (2015), no. 3, 1660–1685.

[HGZ16] Wen Huang, K. A. Gallivan, and Xiangxiong Zhang, *Solving phaselift by low rank Riemannian optimization methods*, Tech. Report UCL-INMA-2016.02, U.C.Louvain, January 2016, short version.

[HJ90] Roger A. Horn and Charles R. Johnson, *Matrix analysis*, Cambridge University Press, 1990.

[HKYH02] Petter Holme, Beom Jun Kim, Chang No Yoon, and Seung Kee Han, *Attack vulnerability of complex networks*, Phys. Rev. E **65** (2002), 056109.

[HM94] U. Helmke and J. B. Moore, *Optimization and dynamical systems*, Spfinger-Verlag, June 1994, doi:10.1109/JPROC.1996.503147.

[HMT11] JN. Halko, P. G. Martinsson, and J. A. Tropp, *Find structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Review **53** (20011), no. 2, 217 – 288.

[HS95] U. Helmke and M. A. Shayman, *Critical points of matrix least squares distance functions*, Linear Algebra and its Applications **19** (1995), no. 1995, 1–19.

[HT04] K. Huper and J. Trumpf, *Newton-like methods for numerical optimization on manifolds*, Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on, vol. 1, Nov 2004, pp. 136–139 Vol.1.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The elements of statistical learning: Data mining, inference, and prediction*, 2end ed., Springer, Princeton, NJ, 2009.

[Hua13] W. Huang, *Optimization algorithms on Riemannian manifolds with applications*, Ph.D. thesis, Florida State University, Department of Mathematics, 2013.

[Jay57] E. T. Jaynes, *Information theory and statistical mechanics*, Phys. Rev. **106** (1957), 620–630.

[JBAS10] M. Journée, F. Bach, P.-.A. Absil, and R. Sepulchre, *Low-rank optimization on the cone of positive semidefinite matrices*, SIAM Journal on Optimization **20** (2010), no. 5, 2327–2351.

[KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing*, Science **220** (1983), no. 4598, 671–680.

[KKKR02] J. Kim, P. L. Krapivsky, B. Kahng, and S. Redner, *Infinite-order percolation and giant fluctuations in a protein interaction network*, Phys. Rev. E **66** (2002), 055101.

[KL07] Othmar Koch and Christian Lubich, *Dynamical low-rank approximation*, SIAM Journal on Matrix Analysis and Applications **29** (2007), no. 2, 434–454, doi:10.1137/050639703.

[Kle10] D. J. Klein, *Centrality measure in graphs*, Journal of Mathematical Chemistry **47** (2010), no. 4, 1209–1223.

[KR05] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: An introduction to cluster analysis*, John Wiley & Sons, Inc., Hoboken, NJ, 2005.

[Lar98] Rasmus Larsen, *Lanczos bidiagonalization with partial reorthogonalization*, DAIMI Report Series **27** (1998), no. 537.

[LDB08] R. Lambiotte, J.C. Delvenne, and M. Barahona, *Laplacian dynamics and multiscale modular structure in networks*, ArXiv e-prints (2008).

[Lee97] John M. Lee, *Riemannian manifolds: An introduction to curvature*, vol. 176, Springer, New York, NY, 1997.

[LH05] Zonghua Liu and Bambi Hu, *Epidemic spreading in community networks*, EPL (Europhysics Letters) **72** (2005), no. 2, 315.

[LHN06] E. A. Leicht, Petter Holme, and M. E. J. Newman, *Vertex similarity in networks*, Phys. Rev. E **73** (2006), 026120.

[LKLS13] Joonseok Lee, Seungyeon Kim, Guy Lebanon, and Yoram Singer, *Matrix approximation under local low-rank assumption*, CoRR **abs/1301.3192** (2013).

[Llo82] S. Lloyd, *Least squares quantization in PCM*, IEEE Transactions on Information Theory **28** (1982), no. 2, 129–137.

[LN08] E. A. Leicht and M. E. J. Newman, *Community structure in directed networks*, Phys. Rev. Lett. **100** (2008), 118703.

[LT13] Yonatan Lupu and Vincent A. Traag, *Trading communities, the networked structure of international relations, and the Kantian peace*, Journal of Conflict Resolution **57** (2013), no. 6, 1011 – 1042.

[Lue72] D. G. Luenberger, *The gradient projection method along geodesics*, Management Science **18** (1972), no. 11, 620–631.

[Lue73] ———, *Introduction to Linear and Nonlinear Programming*, Addison-Wesley, Reading, MA, 1973.

[LW71] Franois Lorrain and Harrison C. White, *Structural equivalence of individuals in social networks*, The Journal of Mathematical Sociology **1** (1971), no. 1, 49–80.

[Man02] J. H. Manton, *Optimization algorithms exploiting unitary constraints*, IEEE Transactions on Signal Processing **50** (2002), no. 3, 635–650.

[MBS11] G. Meyer, S. Bonnabel, and R. Sepulchre, *Regression on fixed-rank positive semidefinite matrices : a Riemannian approach*, Journal of Machine Learning Research **12** (2011), 593–625.

[MJBS09] G. Meyer, M. Journée, S. Bonnabel, and R. Sepulchre, *From subspace learning to distance learning: a geometrical optimization approach*, Proceedings of the IEEE/SP 15th Workshop on Statistical Signal Processing (2009), 385–388.

[MM02] Robert Mahony and Jonathan H. Manton, *The geometry of the Newton method on non-compact lie groups*, J. of Global Optimization **23** (2002), no. 3-4, 309–327.

[MMBS13] B. Mishra, G. Meyer, F. Bach, and R. Sepulchre, *Low-rank optimization with trace norm penalty*, SIAM Journal on Optimization **23** (2013), no. 4, 2124–2149.

[MS13] B. Mishra and R. Sepulchre, *R3mc: A Riemannian three-factor algorithm for low-rank matrix completion*, ArXiv e-prints (2013), –1–1.

[Nat] United Nations, *Statistical yearbook of the United Nations*, Ed. 43, IVATION Datasystems Inc.

[Nat94] _____, *Statistical papers. commodity trade statistics*, 1994, Series D Vol. XLIV, No. 1 - 34.

[New04] M. E. J. Newman, *Analysis of weighted networks*, Phys. Rev. E **70** (2004), 056131.

[New10] M. E. J. Newman, *Networks: An introduction*, Oxford University Press, Inc., New York, NY, USA, 2010.

[NG04] M. E. J. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E **69** (2004), 026113.

[NMB11] Wouter De Nooy, Andrej Mrvar, and Vladimir Batagelj, *Exploratory social network analysis with pajek*, Cambridge University Press, New York, NY, USA, 2011.

[OHM06] R. Orsi, U. Helmke, and J. B. Moore, *A Newton-like method for solving rank constrained linear matrix inequalities*, Automatica **42** (2006), no. 11, 1875–1882, doi:10.1016/j.automatica.2006.05.026.

[O'N83] Barrett O'Neill, *Semi-Riemannian geometry*, vol. 176, Academic Press, New York, NY, 1983.

[OW00] B. Owren and B. Welfert, *The Newton iteration on lie groups*, BIT Numerical Mathematics **40** (2000), no. 1, 121–145.

[OW04] Donal B. O'Shea and Leslie C. Wilson, *Limits of tangent spaces to real surfaces*, American Journcal of Mathematics **126** (2004), no. 5, 951 – 980.

[pai14] paintmaps.com, *Color maps with statistical datas*, 2014.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd, *The pagerank citation ranking: Bringing order to the web.*, Technical Report 1999-66, Stanford InfoLab, November 1999, Previous number = SIDL-WP-1999-0120.

[PDFV05] Gergely Palla, Imre Derényi, Illes Farkas, and Tamás Vicsek, *Uncovering the overlapping community structure of complex networks in nature and society*, Nature **435** (2005), 814 – 818.

[PDV07] Gergely Palla, Imre Derényi, and Tamás Vicsek, *The critical point of k-clique percolation in the Erdős–Rényi graph*, Journal of Statistical Physics **128** (2007), no. 1, 219–227.

[PFP+07] Gergely Palla, Ills J Farkas, Pter Pollner, Imre Dernyi, and Tams Vicsek, *Directed network modules*, New Journal of Physics **9** (2007), no. 6, 186.

[POM09] M. A. Porter, J.-P. Onnela, and P. J. Mucha, *Communities in networks*, Notices of the AMS **56** (2009), no. 9, 1082 – 1097.

[PSR10] S. Pinkert, J. Schultz, and J. Reichardt, *Protein interaction networks – more than mere modules*, PLOS Computational Biology **6** (2010), no. 1, 1–13.

[Qi11] Chunhong Qi, *Numerical optimization methods on Riemannian manifolds*, Ph.D. thesis, Florida State University, 2011.

[RB06] Jörg Reichardt and Stefan Bornholdt, *Statistical mechanics of community detection*, Phys. Rev. E **74** (2006), 217 – 224.

[Rei09] Jörg Reichardt, *Structure in complex networks*, Springer, Lecture Notes in Physics, 2009.

[Rou87] Peter J. Rousseeuw, *Silhouettes: A graphical aid to the interpretation and validation of cluster analysis*, Journal of Computational and Applied Mathematics **20** (1987), 53 – 65.

[RW07] J. Reichardt and R. D. White, *Role models for complex networks*, The European Physical Journal B **60** (2007), no. 2, 217–224.

[RW12] W. Ring and B. Wirth, *Optimization methods on Riemannian manifolds and their application to shape space*, SIAM Journal on Optimization **22** (2012), no. 2, 596–627.

[SDYB12] Michael T. Schaub, Jean-Charles Delvenne, Sophia N. Yaliraki, and Mauricio Barahona, *Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit*, PLoS ONE **7** (2012), no. 2, 1–11.

[SLO99] Catherine A. Sugar, Leslie A. Lenert, and Richard A. Olshen, *An application of cluster analysis to health services research: empirically defined health states for depression from the sf-12*, Technical report, Stanford University, 1999.

[SM83] Gerard Salton and Michael J. McGill, *Introduction to modern information retrieval*, McGraw-Hill, New York u.a., 1983.

[Smi93] Steven Thomas Smith, *Geometric optimization methods for adaptive filtering*, Ph.D. thesis, Harvard University, The Division of Applied Sciences, Cambridge, MA, USA, 1993, UMI Order No. GAX93-31032.

[ST12] Leah B. Shaw and Ilker Tunc, *Epidemic spread in adaptive social networks with community structure*, pp. 519–520, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[Ste83] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM Journal on Numerical Analysis **20** (1983), no. 3, 626 – 637.

[Sug98] Catherine A. Sugar, *Techniques for clustering and classification with applications to medical problems*, Ph.D. thesis, Stanford University, Department of Mathematics, 1998.

[TBD+14] Michele Tizzoni, Paolo Bajardi, Adeline Decuyper, Guillaume Kon Kam King, Christian M. Schneider, Vincent Blondel, Zbigniew Smoreda, Marta C. Gonzlez, and Vittoria Colizza, *On the use of human mobility proxies for modeling epidemics*, PLoS Comput Biol **10** (2014), no. 7, 1 – 15.

[TKVD13] V. A. Traag, G. Krings, and P. Van Dooren, *Significant scales in community structure*, Scientific Reports **3** (2013), 2930.

[Tra14] Vincent Traag, *Algorithms and dynamical models for communities and reputation in social networks*, Springer, Heidelberg, 2014.

[TVDN11] V. A. Traag, P. Van Dooren, and Y. Nesterov, *Narrow scope for resolution-limit-free community detection*, Phys. Rev. E **84** (2011), 016114.

[TWH01] Robert Tibshirani, Guenther Walther, and Trevor Hastie, *Estimating the number of clusters in a data set via the gap statistic*, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **63** (2001), no. 2, 411–423.

[Van13] B. Vandereycken, *Low-rank matrix completion by Riemannian optimization*, SIAM Journal on Optimization **23** (2013), no. 2, 1214—1236.

[VAV09] B. Vandereycken, P.-A. Absil, and S. Vandewalle, *Embedded geometry of the set of symmetric positive semidefinite matrices of fixed rank*, Statistical Signal Processing, 2009. SSP '09. IEEE/SP 15th Workshop on, 2009, pp. 389–392.

[VAV12] ———, *A Riemannian geometry with complete geodesics for the set of positive semidefinite matrices of fixed rank*, IMA Journal of Numerical Analysis **33** (2012), no. 2, 481–514, doi:10.1093/imanum/drs006.

[VB96] L. Vandenberghe and S. P. Boyd, *Semidefinite programming*, SIAM Review **38** (1996), no. 1, 49 – 95.

[VV10] B. Vandereycken and Stefan Vandewalle, *A Riemannian optimization approach for computing low-rank solutions of Lyapunov equations*, SIAM Journal on Matrix Analysis and Applications **31** (2010), no. 5, 2553 – 2579.

[Wal74] Immanuel Wallerstein, *The modern world system: Capitalist agriculture and the origins of the european world-economy in the sixteenth century*, New York: Academic Press, 1974.

[WF94] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*, Cambridge University Press, 1994.

[WL08] Xiaoyan Wu and Zonghua Liu, *How community structure influences epidemic spread in social networks*, Physica A: Statistical Mechanics and its Applications **387** (2008), no. 2 - 3, 623 – 630.

[WR83] Douglas R. White and Karl P. Reitz, *Graph and semigroup homomorphisms on networks of relations*, Social Networks **5** (1983), no. 2, 193 – 234.

[Ye05] Jieping Ye, *Generalized low rank approximations of matrices*, Machine Learning **61** (2005), no. 1 – 3, 167 – 191.

[Zas82] Thomas Zaslavsky, *Signed graphs*, Discrete Applied Mathematics **4** (1982), no. 1, 47 – 74.

[ZHG+16] Guifang Zhou, Wen Huang, Kyle A. Gallivan, Paul Van Dooren, and Pierre-Antoine Absil, *A Riemannian rank-adaptive method for low-rank optimization*, Neurocomputing **192** (2016), 72 – 80.

[Zho15] G. Zhou, *Rank-constrained optimization: A Riemannian manifolds approach*, Ph.D. thesis, Florida State University, Department of Mathematics, 2015.

BIOGRAPHICAL SKETCH

Melissa Marchand, daughter of James and Christine Marchand, was born on June 12th, 1989 in Bakersfield, CA. In the spring of 2011, she graduated as the Outstanding Graduating Senior of the School of Natural Science, Mathematics, and Engineering and the Outstanding Graduating Senior of the Department of Mathematics from California State University of Bakersfield, with a bachelor's degree in Applied Mathematics and a minor in Computer Science. She enrolled in the doctoral program at Florida State University in 2011. Under the advisement of Professor Kyle A. Gallivan and Professor Paul Van Dooren, she obtained her Master's degree in the spring of 2015, and completed her Ph.D in the fall of 2017.

Melissa's research interests include convex and nonconvex optimization, graph partitioning and numerical linear algebra, and their applications to problems such as low-rank matrix approximation, the role extraction problem, and signal and image processing. Besides her dissertation research, she has collaborated with the Naval Surface Warfare Center Panama City Division on multi-input multi-output waveform optimization for synthetic aperture sonar and has contributed to the development of the software, called TreeScaper, for phylogenetic analysis.

After receiving her Ph.D., Melissa will continue to expand her knowledge in applied and computational mathematics.