UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Modeling and Algorithmic Advances for Random Dot Product Graphs

Bernardo Marenco

Doctorate program in Mathematics
Programa de Desarrollo de las Ciencias Básicas
Universidad de la República

Montevideo – Uruguay
October of 2025

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Modeling and Algorithmic Advances for Random Dot Product Graphs

Bernardo Marenco

Doctorate's Thesis submitted to the Doctorate Program in Mathematics, Programa de Desarrollo de las Ciencias Básicas of the Universidad de la República, as part of the necessary requirements for obtaining the title of Doctor in Mathematics.

Directors:
Ph.D. Prof. Paola Bermolen
Ph.D. Prof. Gonzalo Mateos

Academic director:
Ph.D. Prof. Paola Bermolen

Montevideo – Uruguay
October of 2025

Marenco, Bernardo

Modeling and Algorithmic Advances for Random Dot Product Graphs / Bernardo Marenco. - Montevideo: Programa de Desarrollo de las Ciencias Básicas, 2025.

xx, 142 p.: il.; 29, 7cm.

Directors:

Paola Bermolen

Gonzalo Mateos

Academic director:

Paola Bermolen

Doctorate's Thesis – Programa de Desarrollo de las Ciencias Básicas, Doctorate Program in Mathematics, 2025.

Bibliography: p. 136 – 142.

1. Modelos de redes, 2. Random Dot Product Graphs (RDPGs), 3. Aprendizaje de representaciones de grafos, 4. Grafos con pesos, 5. Optimización no convexa. I. Bermolen, Paola. Mateos, Gonzalo. II. Programa de Desarrollo de las Ciencias Básicas, Doctorate Program in Mathematics. III. Title: Modeling and Algorithmic Advances for Random Dot Product Graphs

MEMBERS OF THE THESIS DEFENSE COURT

Ph.D. Prof. Joaquín Fontbona

Ph.D. Prof. Florencia Leonardi

Ph.D Prof. Vince Lyzinski

Ph.D. Prof. Santiago Segarra (Revisor Externo)

Ph.D. Prof. Mauricio Velasco (Revisor Externo)

Montevideo – Uruguay
October of 2025

Agradecimientos

A Euge. A Quin y Guille. A Quique y Rosannita. A Santi.
A Paola y Gonchi. A Fede y Fiori.
A Tortoise, Los Olimareños, MF DOOM y Power Chocolatín Experimento.

Desde esta milonga se da las gracias
A toda persona que sea capaz
De estar dos minutos ensimismada
Tratando aplicada de mejorar

Fernando Cabrera, *Décimas porteñas*

RESUMEN

Las redes constituyen una herramienta natural para representar sistemas complejos, desde interacciones sociales hasta conexiones biológicas y tecnológicas. Un modelo estadístico fundamental para este tipo de datos es el Random Dot Product Graph (RDPG), en el cual cada nodo se asocia con un vector latente en un espacio de baja dimensión, y la probabilidad de que exista una arista depende del producto interno entre dichos vectores. Esta tesis estudia tanto los fundamentos teóricos como los aspectos algorítmicos de la inferencia bajo el modelo RDPG, con especial énfasis en avanzar el aprendizaje de representaciones en grafos (*graph representation learning*).

Comenzamos revisando el problema clásico de *embedding* y mostramos que puede reformularse como una tarea de factorización matricial con restricciones. Basándonos en avances recientes en optimización no convexa, llevamos a cabo un nuevo análisis del paisaje de optimización asociado al objetivo factorizado del RDPG. Demostramos que, bajo la geometría Riemanniana cociente inducida por la invarianza ortogonal del modelo, el paisaje es benigno: todo punto estacionario es un mínimo global o un punto silla estricto. Esto constituye la primera garantía rigurosa de que los algoritmos basados en descenso por gradiente pueden recuperar de manera confiable embeddings del RDPG. Extendemos además el análisis a un objetivo enmascarado que descarta la diagonal y acomoda datos faltantes, estableciendo propiedades de convexidad fuerte restringida y suavidad bajo el modelo RDPG.

Posteriormente, generalizamos el modelo para redes con pesos introduciendo el Weighted RDPG (WRDPG). El WRDPG ofrece un marco no paramétrico que captura momentos de orden superior de las distribuciones de pesos de las aristas más allá de la media, al mismo tiempo que admite un estimador consistente y asintóticamente normal. Asimismo, proponemos un mecanismo generativo que reproduce tanto la estructura como la distribución de pesos observada en redes reales.

Finalmente, exploramos aplicaciones en la detección en tiempo real de puntos de cambio en redes dinámicas. Al combinar embeddings RDPG con métodos estadísticos secuenciales, desarrollamos un marco liviano e interpretable capaz de detectar, en tiempo real, cambios estructurales en grafos en evolución.

En conjunto, esta tesis conecta métodos espectrales, teoría de la optimización e inferencia estadística para avanzar en el uso de embeddings basados en RDPG como base para el aprendizaje de representaciones en grafos.

Palabras claves:
Modelos de redes, Random Dot Product Graphs (RDPGs), Aprendizaje de representaciones de grafos, Grafos con pesos, Optimización no convexa.

ABSTRACT

The Random Dot Product Graph (RDPG) model has emerged as a fundamental tool for representing network data through latent position embeddings. In this framework, each node is associated with a low-dimensional vector, and edges are formed with probabilities given by the inner products of these latent positions. This thesis investigates both the theoretical foundations and algorithmic aspects of inference under the RDPG model, with a focus on advancing graph representation learning for statistical network analysis.

We begin by revisiting the classical embedding problem and showing that it can be reformulated as a constrained matrix factorization task. Leveraging recent advances in nonconvex optimization, we conduct a novel landscape analysis of the factored RDPG objective. We prove that, under the natural quotient Riemannian geometry induced by the model's orthogonal invariance, the optimization landscape is benign: every stationary point is either a global minimizer or a strict saddle. This provides the first rigorous guarantees that gradient-based algorithms can reliably recover RDPG embeddings. We further extend the analysis to a masked objective that naturally discards diagonals and accommodates missing data, establishing restricted strong convexity and smoothness properties under the RDPG model.

We then generalize the model to weighted networks by introducing the Weighted RDPG (WRDPG). The WRDPG offers a nonparametric framework that captures higher-order moments of edge-weight distributions beyond the mean, while admitting a consistent and asymptotically normal estimator. We also propose a generative mechanism that reproduces both the structure and weight distribution of real networks.

Finally, we explore applications in online change-point detection for dynamic networks. By coupling RDPG embeddings with sequential statistical methods, we develop a lightweight and interpretable framework capable of detecting structural changes in evolving graphs in real time.

Overall, this thesis bridges spectral methods, optimization theory, and statistical inference to advance the use of RDPG-based embeddings as a substrate for graph representation learning.

Keywords:
Network models, Random Dot Product Graphs (RDPGs), Graph Representation Learning, Weighted networks, Nonconvex optimization.

List of Figures

Notational conventions

Throughout this document we use the following notation. Real numbers are denoted by plain symbols (either upper or lowercase), such as $i, j, N \in \mathbb{R}$. Vectors in a Euclidean space $\mathbb{R}^d$ are denoted in bold, lowercase, e.g., $\mathbf{x} \in \mathbb{R}^d$, and are assumed to be column vectors. We reserve bold uppercase letters for matrices, such as $\mathbf{A} \in \mathbb{R}^{n \times m}$, with $\mathbf{I}_n$ denoting the $n \times n$ identity matrix, and $\mathbf{1}_{m \times n}, \mathbf{0}_{m \times n}$ the $m \times n$ all-ones and all-zeros matrices, respectively. We will drop the subscripts when the dimensions are clear from context. The symbol $\mathbf{A}^{(k)}$ stands for the k-fold Hadamard (i.e., entry-wise) product of matrix $\mathbf{A}$ with itself, that is:

$$\mathbf{A}^{(k)} := \underbrace{\mathbf{A} \circ \mathbf{A} \circ \cdots \circ \mathbf{A}}_{k \text{ times}}.$$

The group of orthogonal matrices of dimension d is denoted by $O(d)$, i.e., $\mathbf{Q} \in O(d)$ iff $\mathbf{Q} \in \mathbb{R}^{d \times d}$ and $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}$, where $(\cdot)^\top$ stands for transposition.

For any two indexes $i, j \in \mathbb{N}$, δ_{ij} will denote Kronecker's delta, i.e., $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise.

The dot product between $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ is denoted by $\mathbf{x}^\top \mathbf{y}$, whereas $\|\mathbf{x}\|_2$ and $\|\mathbf{x}\|_\infty$ denote the Euclidean (i.e., L_2) and maximum (i.e., L_∞) norm of vector $\mathbf{x}$. When dealing with matrices, $\|\mathbf{A}\|_{\mathrm{F}}$, $\|\mathbf{A}\|_\infty$, $\|\mathbf{A}\|_2$ and $\|\mathbf{A}\|_{2\to\infty}$ denote the Frobenius, infinity (i.e. maximum row sum or $\|\mathbf{A}\|_\infty := \max_i \sum_j |A_{ij}|$), spectral (i.e., largest singular value) and $2 \to \infty$ norm of matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, respectively. The latter is defined as

$$\|\mathbf{A}\|_{2\to\infty} := \sup_{\|\mathbf{x}\|_2 = 1} \|\mathbf{A}\mathbf{x}\|_\infty,$$

see (Cape et al., 2019). The $2 \to \infty$ norm is thus the operator norm of $\mathbf{A}$ acting as $\mathbf{A}(\cdot) : \mathbb{R}^m \mapsto \mathbb{R}^n$, when $\mathbb{R}^m$ is endowed with the Euclidean norm and $\mathbb{R}^n$ with the ∞ norm.

We repeatedly make use of the following relations between the $2 \to \infty$, spectral, and Frobenius norms of any matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$:

$$\begin{aligned} \|\mathbf{A}\|_{2\to\infty} &\le \|\mathbf{A}\|_2 \le \|\mathbf{A}\|_{\mathrm{F}}, \\ \|\mathbf{A}\|_2 &\le \min\left\{\sqrt{n}\|\mathbf{A}\|_{2\to\infty}, \sqrt{m}\left\|\mathbf{A}^\top\right\|_{2\to\infty}\right\}. \end{aligned}$$

The $2 \to \infty$ norm is not submultiplicative; however, the following relations hold for any matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ of suitable dimensions:

$$\begin{aligned}\|\mathbf{AB}\|_{2\to\infty} \leq & \|\mathbf{A}\|_{2\to\infty}\|\mathbf{B}\|_2 \leq \|\mathbf{A}\|_2\|\mathbf{B}\|_2, \\ \|\mathbf{CA}\|_{2\to\infty} \leq & \|\mathbf{C}\|_\infty\|\mathbf{A}\|_{2\to\infty} \leq \sqrt{n}\|\mathbf{C}\|_2\|\mathbf{A}\|_2.\end{aligned}$$

For a function $f : \mathbb{N} \to \mathbb{R}^+$, we say that a random variable $Y \in \mathbb{R}$ is $\mathcal{O}_{\mathbb{P}}(f(N))$ if, for every $\alpha > 0$, there exist constants $N_0 \in \mathbb{N}$ and $C > 0$ such that for all $N \geq N_0$, it holds that

$$\mathbb{P}\{|Y| < Cf(N)\} \geq 1 - N^{-\alpha}.$$

In this case, we write $Y = \mathcal{O}_{\mathbb{P}}(f(N))$. We will sometimes express this informally by saying that $|Y| < Cf(N)$ holds with high probability.

Furthermore, we say that Y is $\Theta_{\mathbb{P}}(f(N))$ if for each $\alpha > 0$ there exists $N_0 \in \mathbb{N}$ and $c, C > 0$ such that for all $N \geq N_0$ it holds that:

$$\mathbb{P}\{cf(N) < Y < Cf(N)\} \geq 1 - N^{-\alpha}.$$

In that case we write $Y = \Theta_{\mathbb{P}}(f(N))$; we will sometimes informally express this as Y being of order $f(N)$ with high probability.

Acronyms

The following acronyms are used throughout the manuscript:

ASE Adjacency Spectral Embedding 10

BCD block-coordinate descent 18, 32

CPD change-point detection 105

GD gradient descent 18

GRL graph representation learning 1, 18

IIR infinite impulse response 52

PSD positive semi-definite 7

RDPG Random Dot Product Graph 1, 4, 7

SBM Stochastic Block Model 2, 4, 5

SVD singular-value decomposition 13

mgf moment generating function 74

pdf probability density function 67

rvs random variables 73

Contents

Chapter 1

Introduction and thesis outline

The study of network-structured data has become central to modern data science, driven by the need to model, understand, and predict interactions in complex systems. From social networks and biological systems to technological and financial infrastructures, graphs offer a natural and expressive language for describing pairwise relationships among entities. Formally, a graph $G = (V, E)$ comprises a set of nodes V, representing entities, and a set of edges E, denoting interactions. However, as the complexity and scale of real-world networks grow, traditional graph-theoretic tools often fall short in providing scalable and flexible representations. This gap has led to the emergence of graph representation learning (GRL) as a powerful paradigm for encoding network data into low-dimensional vector spaces amenable to modern machine (Hamilton, 2020).

At its core, GRL seeks to learn embeddings–vector representations of nodes, edges, or entire graphs–that preserve meaningful structural and relational information. These embeddings serve as a foundation for downstream tasks such as clustering, classification, link prediction, and anomaly detection. Unlike hand-crafted network statistics or manually curated features, GRL offers a data-driven approach that adapts to the geometry and semantics of specific domains. Methods in GRL range from neural approaches such as graph neural networks (GNNs) and random-walk-based techniques like node2vec (Grover & Leskovec, 2016), to latent position models rooted in statistical theory. This thesis adopts the latter perspective, focusing on models that offer interpretability and principled inference guarantees.

Latent position models posit that the connectivity pattern in a graph is governed by unobserved node-level variables residing in a latent space. One particularly elegant and analytically tractable instance of this class is the Random Dot Product Graph (RDPG) model (Athreya et al., 2017; Scheinerman & Tucker, 2010). In an RDPG, each node is associated with a latent vector in a low-dimensional Euclidean space, and the probability of an edge between two nodes is given by the dot product of their latent vectors. This formulation provides a continuous relaxation of discrete

models like the Stochastic Block Model (SBM), naturally capturing phenomena such as transitivity, homophily, and community structure.

Despite its flexibility and mathematical appeal, the RDPG model exhibits a number of practical and theoretical limitations that restrict its utility in real-world applications:

- It traditionally assumes binary (unweighted) and undirected edges, which fails to account for the rich information conveyed by weighted or directed interactions.
- The standard inference procedure–Adjacency Spectral Embedding (ASE)–relies on a spectral decomposition that ignores modeling details such as missing data or known edge uncertainty.
- Its orthogonal invariance introduces identifiability issues in the estimated embeddings, complicating tasks like tracking nodes over time or comparing multiple networks.
- Its standard formulation does not naturally extend to dynamic or online settings, which are increasingly relevant in streaming data environments.

This thesis is motivated by the need to bridge the gap between the theoretical elegance of the RDPG model and the practical demands of modern GRL applications. We approach this goal by revisiting the *embedding problem*: given an observed adjacency matrix drawn from an RDPG, recover the latent positions of the nodes in a way that is both statistically consistent and computationally efficient. We demonstrate how rethinking this estimation problem–through the lens of optimization, generalization, and algorithm design–allows us to overcome several of the RDPG model's inherent shortcomings.

Our contributions are threefold:

1. By identifying structural mismatches between the model and traditional spectral methods, we propose a reformulated optimization problem that better reflects the model's latent geometry and allows for greater algorithmic flexibility and robustness. We develop gradient-based algorithms that efficiently solve the embedding problem of the RDPG, and present a novel landscape analysis which characterizes the behavior of our method under the appropriate quotient Riemannian geometry. We also introduce a Riemannian optimization framework for the embedding problem of the directed variant of the RDPG, where additional identifiability issues arise due to the asymmetric nature of the adjacency matrix. In this setting, the inherent ambiguities extend beyond global rotations, potentially degrading both the quality and interpretability

of the embeddings. To address this, we show that imposing some orthogonality constraints yields a novel GRL formulation for digraphs that preserves interpretability while enabling efficient optimization.

2. We generalize the RDPG model to handle weighted networks, where edge weights are positive real values rather than binary indicators. This leads to the Weighted RDPG (WRDPG), a nonparametric framework that captures higher-order moments of the edge-weight distribution beyond the mean. We develop a corresponding estimation procedure, providing statistical guarantees for the associated estimator, and establish theoretical consistency results. These guarantees are derived through a combination of random matrix concentration inequalities and perturbation bounds for eigendecompositions. In addition, we introduce a data-driven generative mechanism that can reproduce both the structural patterns and the empirical weight distribution observed in real networks, despite the WRDPG not being generative by construction.

3. We develop a lightweight and scalable framework for change-point detection in evolving networks that leverages the RDPG model without requiring repeated embeddings of successive observations. By sidestepping the need for full latent position estimation at each time step, our method remains computationally efficient and suitable for true online operation, while still providing interpretable and timely detection of structural changes in network dynamics.

Taken together, these advances extend the RDPG model's reach beyond its original scope, embedding it firmly within the broader paradigm of GRL. Rather than treating the RDPG as a fixed generative model, we promote a view in which it serves as a flexible modeling scaffold–one that can be adapted, optimized, and applied to a wide variety of settings where structure matters.

The remainder of the thesis is structured as follows. Chapter 2 introduces foundational concepts from statistical network analysis and latent position models, with an emphasis on the RDPG and its connections to other frameworks. Chapter 3 develops novel formulations and optimization strategies for the embedding problem, supported by a theoretical landscape analysis. Chapter 4 presents the WRDPG model and its inference guarantees. Chapter 5 illustrates how the RDPG framework can be leveraged for sequential change-point detection in temporal networks. Finally, Chapter 6 summarizes our findings and discusses open questions and avenues for future work.

Chapter 2

Preliminaries and related work

In this chapter, we present a class of probabilistic models for network data that rely on the notion of latent positions, with particular emphasis on the Stochastic Block Model (SBM) and the Random Dot Product Graph (RDPG) model. These models provide flexible frameworks for capturing the underlying generative mechanisms of observed graphs, and serve as foundations for a wide array of statistical inference procedures on network data.

As mentioned in Chapter 1, we model a network as a graph $G = (V, E)$, where V is the set of nodes (or vertices) and $E \subseteq V \times V$ is the set of edges. Throughout the chapter, we consider graphs with $N = |V|$ nodes, indexed as $V = \{1, 2, \ldots, N\}$. An edge $(i, j) \in E$ represents the presence of a relationship between nodes i and j. While we begin our exposition with undirected and unweighted graphs–meaning that $(i, j) \in E$ implies $(j, i) \in E$, and that edges do not carry weights–we later present an extension of the RDPG model to accommodate directed graphs. In Chapter 4, we also introduce an extension of the RDPG model that accommodates weighted edges. In what follows, we denote by $\mathbf{A} \in \mathbb{R}^{N \times N}$ the adjacency matrix of an unweighted graph; that is, $A_{ij} = 1$ if $(i, j) \in E$, and 0 otherwise.

Latent position models posit that each node $i \in V$ is associated with an unobserved vector $\mathbf{x}_i \in \mathbb{R}^d$, called its *latent position*, and that the probability of an edge between two nodes depends on their respective latent positions. Under the SBM, nodes are partitioned into discrete communities, and edge probabilities are determined solely by the community memberships. In contrast, the RDPG model assumes that edge probabilities arise from the inner products of continuous-valued latent vectors.

We will carefully introduce both models and explain how the SBM can be viewed as a special case of the RDPG, under suitable choices of latent positions. This unifying perspective reveals how both community structure and more general geometric relationships can be modeled within a single latent position framework.

The remainder of the chapter is structured as follows. We begin by formally

defining the SBM and reviewing its probabilistic structure. We then introduce the RDPG model and show how it can be viewed as a generalization of the SBM. Next, we explore the properties of the RDPG, discuss estimation techniques, and examine its applications in network inference tasks. We then present an extension of the RDPG model to accommodate directed graphs, and introduce a further generalization that captures heterophilous network structures.

2.1 Latent position network models

A powerful approach within the domain of statistical analysis of network data is the use of latent variable models (Hoff et al., 2002), where each vertex is endowed with an unobserved feature vector that encapsulates its propensity to establish relational ties. This general class of models offers a probabilistic lens through which the observed network structure can be understood in terms of latent geometric or probabilistic constructs (see e.g., Kolaczyk (2017, Ch. 2.2) and Izenman, 2023, Ch. 10.7).

A prominent example of latent variable models is the Stochastic Block Model (SBM), in which each vertex belongs to one of C latent communities (Holland et al., 1983). Formally, each vertex i is assigned a community label $z_i \in \{1, \ldots, C\}$, and the probability of an edge between vertices i and j depends solely on their community memberships. That is,

$$\mathbb{P}\left(A_{ij} = 1 \mid z_i = r,\, z_j = s\right) = b_{rs},$$

where $\mathbf{B} = (b_{rs}) \in [0,1]^{C \times C}$ is the so-called block probability matrix–that is, the matrix that encodes the connection probabilities between (and within) communities. This induces a block-constant expectation matrix $\mathbf{P} = \mathbb{E}\left(\mathbf{A}\right)$, where each block corresponds to a pair of communities and has constant entries given by the corresponding entry in the matrix $\mathbf{B}$. The SBM thus models networks exhibiting community structure–a common feature in social, biological, and technological networks–and offers a tractable framework for statistical inference.

While the community labels $z_1, \ldots, z_N$ can be treated as fixed, a common and often more realistic formulation treats them as random variables. Under this probabilistic version of the SBM, each z_i is drawn independently from a categorical distribution over C classes with mixing proportions $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_C)^\top$, i.e.,

$$\mathbb{P}\left(z_i = c\right) = \pi_c, \text{ for each } c = 1, \ldots, C, \text{ with } \sum_{c=1}^{C} \pi_c = 1.$$

This random labeling introduces an additional layer of variability into the model and allows for a Bayesian interpretation of the community structure. It also facilitates

statistical inference, as it models heterogeneity in community membership across different network realizations.

Although SBMs involve discrete latent variables, they are closely linked to continuous latent variable models. In latent variable models, each node i is associated with a continuous latent position $\mathbf{x}_i \in \mathbb{R}^d$, and the probability of an edge between nodes i and j is given by $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ for some function $\kappa : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$, which typically encapsulates some measure of proximity–often the Euclidean distance or the inner product between latent vectors. Notably, the SBM can be viewed as a degenerate latent space model where all vertices within the same block share identical latent positions. This equivalence reveals that latent space models generalize SBMs and offers a unified geometric perspective for representing network data.

This connection is rigorously formalized through the theory of graph limits and graphons (Borgs et al., 2008; Lovász & Szegedy, 2006). A graphon $W\colon [0,1]^2 \to [0,1]$ is a symmetric, measurable function which can be thought of as a continuous analogue of the adjacency matrix for graphs with infinitely many vertices. Given a graphon W, one generates a random graph by first sampling independent uniform variables $U_1, \ldots, U_N \sim \text{Uniform}[0,1]$, then connecting each pair (i,j) independently with probability $W(U_i, U_j)$.

The cut distance between two graphons W and W' is defined as

$$\delta_\square(W, W') = \inf_{\varphi} \sup_{S,T \subset [0,1]} \left| \int_{S \times T} \big(W(x,y) - W'(\varphi(x), \varphi(y))\big)\, dx\, dy \right|,$$

where the infimum is over all measure-preserving bijections φ of $[0,1]$. This metric captures how closely two graphons approximate each other under relabelings of the vertices. A key analytic result in this theory is the graphon analogue of Szemerédi's Regularity Lemma (Lovász & Szegedy, 2007), which states that any graphon can be approximated arbitrarily well in the cut norm by a step-function graphon–that is, a graphon corresponding to an SBM with finitely many blocks:

Theorem (Lovász and Szegedy (2007), Theorem 3.1). *For any graphon W and any $\varepsilon > 0$, there exists a step-function graphon W' such that*

$$\delta_\square(W, W') \leq \varepsilon.$$

This result implies that SBMs serve as universal approximators for the space of graphons under the cut distance, meaning any complex network structure can, in principle, be captured by an SBM with a sufficiently large number of blocks. However, the number of blocks required to achieve a given approximation accuracy can scale exponentially with the complexity of the underlying structure, making such approximations impractical in many settings. Despite this, SBMs remain useful

both as direct models for networks exhibiting community structure and as foundational tools for approximating more intricate latent variable models. For instance, a graphon arising from a latent space model with continuous latent positions can be approximated by a step-function graphon, effectively discretizing the latent space into regions corresponding to block memberships.

These insights have deep practical and theoretical ramifications. Highly complex latent structures inherent in continuous latent space frameworks can be closely approximated by SBMs, facilitating scalable inference and parameter estimation for large networks. Moreover, they unify disparate modeling approaches –ranging from community detection in SBMs to geometric embeddings in latent space models– under a common analytical framework rooted in graph limits. This connection underscores the SBM's dual role as both a practical modeling tool and a fundamental building block for approximating the limits of large network sequences. As we will shortly see, the SBM can be cast as a particular case of the central model addressed in this thesis–that is, the Random Dot Product Graph model.

2.2 The RDPG model

An especially compelling instantiation of the idea of latent position models is to define the edge probability between two nodes as a function of the inner product of their latent vectors. This is not merely an arbitrary choice: by Mercer's theorem, any positive semi-definite (PSD) kernel function can be approximated (in an L^2 sense) by inner products in a suitably chosen Hilbert space. Thus, modeling connectivity via inner products aligns naturally with the theory of kernel methods, ensuring both mathematical tractability and expressive power. It provides a principled way to capture a wide range of network structures while retaining computational efficiency. In the following, we delve into the Random Dot Product Graph (RDPG) model, which formalizes this notion and serves as a foundational building block for latent space modeling in network analysis. This model stands out for its interpretability and theoretical tractability, providing a bridge between latent geometry and observed connectivity patterns.

In the standard RDPG model, we consider a simple undirected and unweighted random graph $G = (V, E)$ with N vertices, where each vertex $i \in V$ is associated with a latent position $\mathbf{x}_i \in \mathbb{R}^d$. Typically, the dimensionality of the embedding space d is such that $d \ll N$. The RDPG model posits that for any pair of vertices i and j, the presence of an edge is determined by a Bernoulli random variable $A_{ij} \in \{0, 1\}$ with success probability given by the dot product of the latent positions, that is,

$$A_{ij} \sim \text{Bernoulli}(\mathbf{x}_i^\top \mathbf{x}_j), \quad \text{for } i > j,$$

where the A_{ij}'s are mutually independent. Since self loops are excluded, $A_{ii} \equiv 0$ for all $i \in V$. Here, $\mathbf{A} \in \{0,1\}^{N \times N}$ is the graph's symmetric adjacency matrix, and the inner product $\mathbf{x}_i^\top \mathbf{x}_j$ must lie in the interval $[0,1]$ to yield a valid probability. In other words, the RDPG model specifies

$$\mathbb{P}\left(\mathbf{A} \,\middle|\, \mathbf{X}\right) = \prod_{i<j} (\mathbf{x}_i^\top \mathbf{x}_j)^{A_{ij}} (1 - \mathbf{x}_i^\top \mathbf{x}_j)^{1-A_{ij}}, \tag{2.1}$$

where the matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times d}$ has all the nodes' latent position vectors as its rows. Note that under this model, the edge-formation probabilities are given by the off-diagonal entries $P_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$ of the PSD matrix $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$.

RDPGs can also incorporate random latent positions by defining a distribution F supported on a subset $\mathcal{X} \subset \mathbb{R}^d$, such that $\mathbf{x}^\top \mathbf{y} \in [0,1]$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. Under appropriate assumptions on the distribution F, it is possible to establish asymptotic results for estimators of the latent positions; see the ASE formulation discussed below.

Remark 2.2.1 (Identifiability of latent positions). The RDPG model is identifiable up to orthogonal transformations of $\mathbf{X}$. To see this, consider an orthogonal matrix $\mathbf{Q} \in O(d)$, and note that the matrix $\mathbf{Y} = \mathbf{X}\mathbf{Q}$ will produce the same probability matrix $\mathbf{P}$, since:

$$\mathbf{Y}\mathbf{Y}^\top = \mathbf{X}\mathbf{Q}(\mathbf{X}\mathbf{Q})^\top = \mathbf{X}\mathbf{Q}\mathbf{Q}^\top\mathbf{X} = \mathbf{X}\mathbf{X}^\top = \mathbf{P}.$$

Example 2.2.1 (Erdös-Rényi graphs and SBMs as RDPGs). The RDPG model is a tractable yet expressive family of random graphs that subsume Erdös-Rényi (ER) ensembles as particular cases. Indeed, if $\mathbf{x}_i = \sqrt{p}\ \forall\, i$, we obtain an ER graph with edge probability p.

SBMs with a PSD block probability matrix $\mathbf{B} \in \mathbb{R}^{C \times C}$ can also be naturally cast as instances of the RDPG model. To see this, let $\mathbf{C} \in \mathbb{R}^{N \times C}$ be the one-hot encoding matrix of community assignments, where the i-th row of $\mathbf{C}$ has a 1 in position c if node i belongs to community c, and 0 elsewhere. Note that under the SBM the edge-formation probabilities are given by the off-diagonal entries of matrix $\mathbf{P} = \mathbf{C}\mathbf{B}\mathbf{C}^\top$. Now, let $d = \operatorname{rank}(\mathbf{B})$, and consider the spectral decomposition $\mathbf{B} = \mathbf{U}_\mathbf{B}\mathbf{D}_\mathbf{B}\mathbf{U}_\mathbf{B}^\top$, where $\mathbf{U}_\mathbf{B} \in \mathbb{R}^{C \times d}$ and $\mathbf{D}_\mathbf{B} \in \mathbb{R}^{d \times d}$. Since $\mathbf{B}$ is PSD, we can define the matrix $\mathbf{Z} = \mathbf{U}_\mathbf{B}\mathbf{D}_\mathbf{B}^{1/2} \in \mathbb{R}^{C \times d}$, whose rows encode the latent positions corresponding to each community. Therefore $\mathbf{B} = \mathbf{Z}\mathbf{Z}^\top$ and:

$$\mathbf{P} = \mathbf{C}\mathbf{B}\mathbf{C}^\top = \mathbf{C}\mathbf{Z}\mathbf{Z}^\top\mathbf{C}^\top = \mathbf{X}\mathbf{X}^\top,$$

where we have defined the latent position matrix $\mathbf{X} := \mathbf{C}\mathbf{Z} \in \mathbb{R}^{N\times d}$. This shows that any SBM with a PSD block matrix $\mathbf{B}$ can be represented as an RDPG, with embedding dimension equal to $\text{rank}(\mathbf{B})$.

If the community assignments are random–i.e., each node belongs to community c independently with probability π_c–then the RDPG distribution F can be defined over the finite set $\mathcal{X} = \{\mathbf{z}_1, \ldots, \mathbf{z}_C\}$, where $\mathbf{z}_c$ is the c-th row of $\mathbf{Z}$, with F given by a categorical distribution on $\mathcal{X}$ with probabilities $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_C)$.

In light of the approximation result for SBMs discussed in Section 2.1, the RDPG possesses an expressiveness that has been a major driver behind the model's popularity. Moreover, the RDPG definition provides a natural geometric interpretation of connectivity: angles between latent position vectors indicate affinity between vertices, and their magnitudes indicate how well connected they are. For $d \leq 3$, visual inspection of the nodes' vector representations can reveal community structure. For higher dimensions or more complex scenarios, angle-based clustering of nodal embeddings can also be used (Lyzinski et al., 2017; Scheinerman & Tucker, 2010).

The associated RDPG inference problem of estimating latent positions from graph observations enjoys strong asymptotic properties, facilitating statistical inference tasks by bringing to bear tools of geometrical data analysis in latent space (Athreya et al., 2017). Let us describe the inference method in such a setup. Given an observed adjacency matrix $\mathbf{A}$ and a prescribed embedding dimension d–typically obtained using an elbow rule on $\mathbf{A}$'s eigenvalue scree plot, using e.g., (M. Zhu & Ghodsi, 2006)–, the goal is to estimate $\mathbf{X}$. If one were to seek a maximum likelihood estimator (MLE), then under the model in (2.1) this would involve solving the following optimization problem:

$$\hat{\mathbf{X}}_{\text{MLE}} = \underset{\mathbf{X}\in\mathbb{R}^{N\times d}}{\text{argmax}} \sum_{i<j} \left[A_{ij} \log(\mathbf{x}_i^\top \mathbf{x}_j) + (1 - A_{ij}) \log(1 - \mathbf{x}_i^\top \mathbf{x}_j)\right].$$

Although the RDPG model assumes that the inner products $\mathbf{x}_i^\top \mathbf{x}_j$ lie in the interval $[0, 1]$, in practice this constraint is typically relaxed when estimating the latent positions. This relaxation simplifies the optimization landscape and, as we will shortly see, makes spectral methods viable. Nonetheless, the MLE objective remains non-convex and computationally intensive, requiring a sum over all $\binom{N}{2}$ node pairs and involving a non-linear, non-concave likelihood function. Solving this directly has computational complexity on the order of N^2, which is prohibitive for large-scale graphs.

Furthermore, as pointed out by Xie and Xu (2023), the model (2.1) belongs to the curved exponential family, rather than to the more tractable canonical exponential family. As a consequence, standard results guaranteeing the existence and

uniqueness of the MLE do not readily apply. This distinction is significant, since, as highlighted by Bickel and Doksum (2015), the theoretical analysis of MLEs in curved exponential families presents greater challenges compared to their canonical counterparts.

In contrast, the Adjacency Spectral Embedding (ASE) estimator provides a scalable and effective alternative based on the spectral decomposition of the adjacency matrix. This estimator solves the following least-squares approximation (Scheinerman & Tucker, 2010) to obtain

$$\hat{\mathbf{X}} \in \underset{\mathbf{X} \in \mathbb{R}^{N \times d}}{\operatorname{argmin}} \left\| \mathbf{A} - \mathbf{X}\mathbf{X}^\top \right\|_{\mathrm{F}}^2. \tag{2.2}$$

In other words, $\hat{\mathbf{P}} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top$ is the best rank-d PSD approximation to the adjacency matrix $\mathbf{A}$, in the Frobenius-norm sense. The rationale behind the estimator (2.2) is that, since $\mathbb{E}(\mathbf{A}) = \mathbf{P} = \mathbf{X}\mathbf{X}^\top$, the approximation $\hat{\mathbf{P}}$ should be close to $\mathbf{P}$, provided that one can control the deviations of $\mathbf{A}$ from its mean under the RDPG model. This is the standard pathway used in the literature to establish statistical guarantees for the ASE estimator, as in (Athreya et al., 2016; Lyzinski et al., 2017; Sussman et al., 2014).

Note that, due to a classic result by Eckart and Young (1936), a solution to (2.2) is readily given by $\hat{\mathbf{X}} = \hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2}$, where $\mathbf{A} = \mathbf{U}_\mathbf{A}\mathbf{D}_\mathbf{A}\mathbf{U}_\mathbf{A}^\top$ is the eigendecomposition of $\mathbf{A}$, $\hat{\mathbf{D}} \in \mathbb{R}^{d \times d}$ is a diagonal matrix with the d largest-magnitude eigenvalues of $\mathbf{A}$, and $\hat{\mathbf{U}} \in \mathbb{R}^{N \times d}$ are the associated eigenvectors from $\mathbf{U}_\mathbf{A}$.[1] Also note that, due to the orthogonal ambiguity discussed in Remark 2.2.1, $\hat{\mathbf{Y}} := \hat{\mathbf{X}}\mathbf{Q}$ is also a solution to (2.2) for any $\mathbf{Q} \in O(d)$.

Asymptotic ($N \to \infty$) consistency and normality results for ASE are available; see e.g., (Athreya et al., 2017). These results formalize the intuition that, under the RDPG model, the adjacency spectral embedding $\hat{\mathbf{X}}$ approximates the true latent positions $\mathbf{X}$ increasingly well as the number of nodes grows. In particular, the consistency result provides uniform consistency of the adjacency spectral embedding in the maximum row-wise Euclidean norm. It implies that every estimated latent position converges almost surely to the corresponding true position (up to an orthogonal transformation), as the number of nodes grows.

Theorem (Athreya et al. (2017), Theorem 26). *Let $\hat{\mathbf{X}}$ be the ASE of $\mathbf{A}$, and let $\mathbf{X} \in \mathbb{R}^{N \times d}$ be the matrix of true latent positions in an RDPG. Provided d is*

[1] Under mild assumptions on the latent positions distribution F it is possible to show that the top d eigenvalues of $\mathbf{A}$ are nonnegative with probability tending to 1 as $N \to \infty$, so $\hat{\mathbf{X}}$ is well defined, see (Athreya et al., 2017) for details.

known, there exists a sequence of orthogonal matrices $\{\mathbf{Q}_N\}_{N\in\mathbb{N}}$ *such that:*

$$\max_{1\le i\le N}\left\|\mathbf{Q}_N\hat{\mathbf{X}}_i-\mathbf{X}_i\right\|_2 \underset{N}{\to} 0 \quad \textit{almost surely,}$$

where $\mathbf{X}_i$ *and* $\hat{\mathbf{X}}_i$ *are the* i*th rows of* $\mathbf{X}$ *and* $\hat{\mathbf{X}}$*, respectively.*

The asymptotic normality result further refines this by showing that, after suitable alignment and rescaling, the rows of $\hat{\mathbf{X}}$ are approximately multivariate normal centered at the true latent positions.

Theorem (Athreya et al. (2017), Theorem 27). *Let* $\mathbf{x}_1,\dots,\mathbf{x}_N$ *be i.i.d. samples from a distribution* F *on* $\mathcal{X}\subset\mathbb{R}^d$*, and let* $\hat{\mathbf{X}}$ *be the ASE of the corresponding RDPG. Fix a vertex* i *and assume the second moment matrix for* F *has full rank. Then there exists a sequence of orthogonal matrices* $\{\mathbf{Q}_N\}_{N\in\mathbb{N}}$ *such that for each fixed* i*,*

$$\sqrt{N}\left(\mathbf{Q}_N\hat{\mathbf{X}}_i-\mathbf{x}_i\right)\xrightarrow{\mathcal{L}}\mathcal{N}(0,\Sigma(\mathbf{x}_i)),$$

where $\Sigma(\mathbf{x}_i)$ *is an explicit covariance matrix depending on the latent position* $\mathbf{x}_i$ *and the distribution* F*.*

These results rely on a combination of tools from matrix perturbation theory (notably the Davis–Kahan $\sin\Theta$ theorem), concentration inequalities for random graphs (e.g., matrix Bernstein inequalities), and probabilistic limit theorems. As pointed before, the key insight is that the adjacency matrix $\mathbf{A}$ concentrates around its expectation $\mathbf{P}=\mathbf{X}\mathbf{X}^\top$, and so its leading eigenvectors and eigenvalues also concentrate, which allows for control over the quality of the low-rank approximation produced by the spectral decomposition. We will follow the same rationale in Chapter 4, where we prove analogous results for an extension of the RDPG model that accommodates weighted graphs.

Example 2.2.2 (RDPG inference on the karate club network). As a way to illustrate the inference procedure on a real graph, consider the well-known karate club network, also known as Zachary's karate club. In (Zachary, 1977), Wayne W. Zachary introduced a social network representing the interpersonal relationships among 34 members of a university karate club. The data was collected over a two-year period through direct observation and questionnaires, and the network is represented as an undirected graph where each node corresponds to a club member, and an edge indicates if the connected members have a reported social interaction outside the club.

During the observation period, a dispute between the club's administrator and its instructor escalated to the point of splitting the club into two factions. This division provides a natural ground truth for evaluating community detection algorithms. In

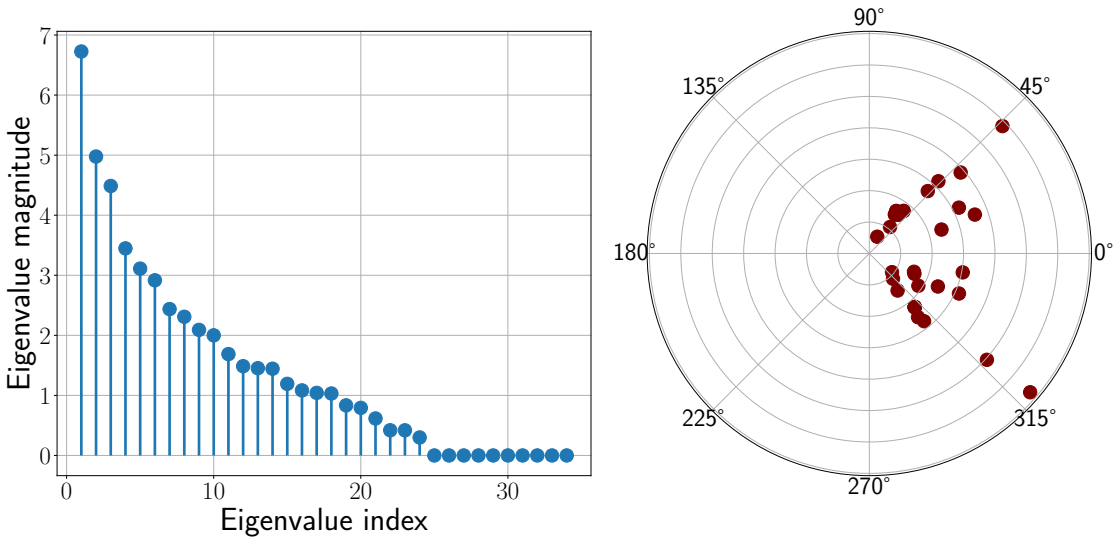

Figure 2.1: Scree plot for the adjacency matrix of Zachary's karate club graph (left) and polar plot of first two dimensions of nodes' embeddings obtained via ASE (right).

the context of the RDPG model, one can apply the ASE to the network's adjacency matrix to estimate latent positions for each node. In Figure 2.1, we display the scree plot of the adjacency matrix. Note the sharp elbow observed after the third eigenvalue, which motivates the choice of embedding dimension $d = 3$. This choice is consistent with the dimension selected using the elbow method proposed by M. Zhu and Ghodsi (2006), as implemented in Python's `Graspologic` library (Chung et al., 2019).

Figure 2.1 also shows the result of applying ASE, where we plot the first two dimensions of the embedding. We present the embeddings in polar coordinates, as angular alignment under the RDPG model is likely to reflect an underlying community structure. Notably, when projected onto two dimensions, the scatter plot reveals a clear separation between two groups, allowing for a visual identification of the emergent communities.

2.2.1 The RDPG model for directed graphs

As previously introduced, the RDPG model is only suitable for undirected graphs. Indeed, $\mathbf{X}\mathbf{X}^\top = \mathbf{P}$ is always symmetric. For directed graphs (*digraphs*), edges are defined as *ordered* pairs (i, j), with $i, j \in V$. Since edges (i, j) and (j, i) are different objects, so could be the probabilities P_{ij} and P_{ji}. By convention, we say (i, j) starts from i and points to j.

Digraphs then require an adaptation to the RDPG model, where each node $i \in V$ has an associated column vector $\mathbf{x}_i$–now in $\mathbb{R}^{2d}$ (Priebe et al., 2017). Let us denote by $\mathbf{x}_i^l$ and $\mathbf{x}_i^r$ the first and last d entries of $\mathbf{x}_i$, respectively. Likewise, let

$\mathbf{X}^l, \mathbf{X}^r \in \mathbb{R}^{N\times d}$ be the matrices stacking the transposed nodal vectors as their rows. In direct analogy to the undirected case, we define the directed RDPG (DRDPG) model as

$$\mathbb{P}\left(\mathbf{A} \,\middle|\, \mathbf{X}\right) = \prod_{i\neq j}[(\mathbf{x}_i^l)^\top \mathbf{x}_j^r]^{A_{ij}}[1-(\mathbf{x}_i^l)^\top \mathbf{x}_j^r]^{1-A_{ij}}$$

[cf. the product over all $i \neq j$ here versus $i < j$ in (2.1)], and the *asymmetric* matrix of connection probabilities now becomes

$$\mathbf{P} = \mathbf{X}^l(\mathbf{X}^r)^\top.$$

Intuitively, we say $\mathbf{x}_i^l$ models node i's outgoing connectivity and $\mathbf{x}_i^r$ its incoming one. The probability of existence of the edge (i, j) is given by $(\mathbf{x}_i^l)^\top \mathbf{x}_j^r$.

Let us now discuss how to estimate the matrices $\mathbf{X}^l$ and $\mathbf{X}^r$ from a graph observation. Since $\mathbf{P} = \mathbb{E}[\mathbf{A}]$ still holds, we seek a pair $\{\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r\}$ such that $\hat{\mathbf{X}}^l(\hat{\mathbf{X}}^r)^\top$ is the best rank-d approximant of $\mathbf{A}$. Letting $\mathbf{A} = \mathbf{U}_\mathbf{A}\mathbf{D}_\mathbf{A}\mathbf{V}_\mathbf{A}^\top$ be the singular-value decomposition (SVD) of $\mathbf{A}$, we set

$$\hat{\mathbf{X}}^l = \hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2} \text{ and } \hat{\mathbf{X}}^r = \hat{\mathbf{V}}\hat{\mathbf{D}}^{1/2}, \tag{2.3}$$

where, as before, $\hat{\mathbf{D}} \in \mathbb{R}^{d\times d}$ is a diagonal matrix with the d largest singular values of $\hat{\mathbf{A}}$ in its diagonal, and $\hat{\mathbf{U}}, \hat{\mathbf{V}} \in \mathbb{R}^{N\times d}$ are the matrices with the corresponding left and right singular vectors as columns, respectively. This procedure is also known as ASE.

While this extension of the model has been known for over a decade, to the best of our knowledge, the only available asymptotic result concerns the consistency of the ASE, as established in the work of Sussman et al. (2012). Although we do not explore this direction further in the present work, it is worth noting that a full characterization of the asymptotic distribution of the ASE in the directed setting remains an open problem in the field.

2.2.2 The Generalized RDPG model

While the RDPG model has proven useful for analyzing network data with latent geometric structure, its formulation inherently restricts the edge probability matrix to be PSD. This restriction limits the class of networks the model can represent–most notably, those exhibiting *homophily*, also known as *assortative mixing* in the social network literature. As described by Newman (2018), homophily refers to the tendency of vertices to form edges preferentially with others that are similar in some way–an idea often summarized by the adage "birds of a feather flock together". For example, in a social network of high school students, one often observes that

friendships form more frequently between students in the same grade level, club, or extracurricular group, illustrating a homophilic pattern.

In contrast, *heterophily* (or disassortative mixing) refers to the tendency of nodes to connect with others who are dissimilar–a pattern observed in, for example, bipartite networks like customer-product or author-paper graphs. Since the RDPG model cannot produce non-PSD edge probability matrices, it fails to capture such heterophilic structure.

To understand why the RDPG model inherently captures homophilic behavior, remember that under this model the probability of an edge between nodes i and j is given by the inner product $\mathbf{x}_i^\top \mathbf{x}_j$ between latent positions. This implies that high connection probabilities arise when $\mathbf{x}_i$ and $\mathbf{x}_j$ are close–that is, when the latent vectors are similar. Thus, nodes that are "similar" in the latent space (i.e., have nearby or aligned latent vectors) are more likely to be connected. This alignment captures the essence of homophily: the more alike two nodes are, the higher the chance they are linked. Since the model does not permit negative entries in $\mathbf{P}$, it cannot naturally express cases where dissimilarity increases the likelihood of connection–a hallmark of heterophily.

To overcome this limitation, Rubin-Delanchy et al. (2022) introduced the Generalized RDPG (GRDPG) model. This extension retains the idea of associating each node with a latent vector, but generalizes the dot product by introducing an indefinite inner product. In the GRDPG, the probability of an edge between nodes i and j is given by

$$\mathbb{P}(A_{ij} = 1) = \mathbf{x}_i^\top \mathbf{I}_{p,q} \mathbf{x}_j,$$

where $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$, and $\mathbf{I}_{p,q}$ is a signature matrix of size $d \times d$ that defines the inner product. This matrix is diagonal, with the first p diagonal entries equal to $+1$ and the remaining q entries equal to -1, so that $d = p+q$. This construction enables the GRDPG to model both assortative and disassortative connectivity patterns within the same framework.

The dimension d, as well as the values of p and q, can be estimated directly from data. A common approach is similar to the one used in the standard RDPG setting–that is, the elbow method proposed by M. Zhu and Ghodsi (2006). This method identifies the embedding dimension d by locating a sharp change (or "elbow") in the scree plot of the adjacency matrix's eigenvalues. Once d is determined, the number of positive and negative eigenvalues among the largest d is used to estimate p and q, respectively. This provides a practical, data-driven way to specify the signature matrix used in the GRDPG model.

Inference under the GRDPG model requires a slight modification of the ASE

of the observed adjacency matrix $\mathbf{A}$. As before, let $\hat{\mathbf{U}} \in \mathbb{R}^{N\times d}$ and $\hat{\mathbf{D}} \in \mathbb{R}^{N\times d}$ denote the eigenvectors and eigenvalues associated with the d largest (in magnitude) eigenvalues of $\mathbf{A}$. The estimated latent positions are given by $\hat{\mathbf{X}} = \hat{\mathbf{U}}|\hat{\mathbf{D}}|^{1/2}$, where the square root is applied elementwise to the absolute values of the eigenvalues. This embedding is well-defined even when some of the top d eigenvalues are negative, as is the case in the GRDPG. Under the standard RDPG model (and suitable assumptions on the distribution F of latent positions), these leading eigenvalues are all positive with high probability, so the ASE simplifies to the usual form $\hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2}$. The modification in the GRDPG setup allows the ASE to accommodate indefinite structure.

Example 2.2.3 (Back to the karate club). Revisiting the karate club network, recall that the scree plot in Figure 2.1 suggested an embedding dimension of $d = 3$. However, inspection of the top three eigenvalues in magnitude reveals that the third is negative. Following the discussion above, we therefore set $p = 2$ and $q = 1$, yielding a signature matrix $\mathbf{I}_{p,q}$ with diagonal entries $(1, 1, -1)^\top$. This implies that, when visualizing the embeddings, the third coordinate must be "reversed", i.e., its sign flipped when plotting. Figure 2.2 presents the resulting pairwise scatter plots of the embeddings, accounting for this reversal. As observed, even accounting for this reversal in the third dimension yields little improvement, with the second dimension remaining the most informative for distinguishing the two communities. The node colors in the figure correspond to the true communities provided with the dataset, not the outcome of any community detection procedure applied to the embeddings.

One notable drawback of the GRDPG model is that the identifiability of the latent positions is weaker than in the RDPG case. While latent positions in an RDPG are identifiable up to an orthogonal transformation (which preserves Euclidean distances), the GRDPG admits indefinite orthogonal transformations, which may distort pairwise distances. As a result, geometric relationships–such as closeness between nodes in the latent space–may not be preserved, complicating tasks like clustering or visualization that rely on distance-based structure.

Nonetheless, the GRDPG model retains favorable asymptotic properties similar to those of the RDPG. In (Rubin-Delanchy et al., 2022) a central limit theorem for the ASE under the GRDPG model is established. Specifically, they show that if $\mathbf{A} \sim \text{GRDPG}(F)$, where F is a distribution over $\mathbb{R}^d$ such that $\mathbf{x}^\top \mathbf{I}_{p,q} \mathbf{y} \in [0, 1]$ for all $\mathbf{x}, \mathbf{y}$ in its support, then the ASE $\hat{\mathbf{X}}$ satisfies a pointwise asymptotic normality result. That is, for each fixed node i, the embedding $\hat{\mathbf{x}}_i$ converges in distribution to a mixture of Gaussians centered around the true latent position, up to an indefinite orthogonal transformation. This result provides theoretical support for using

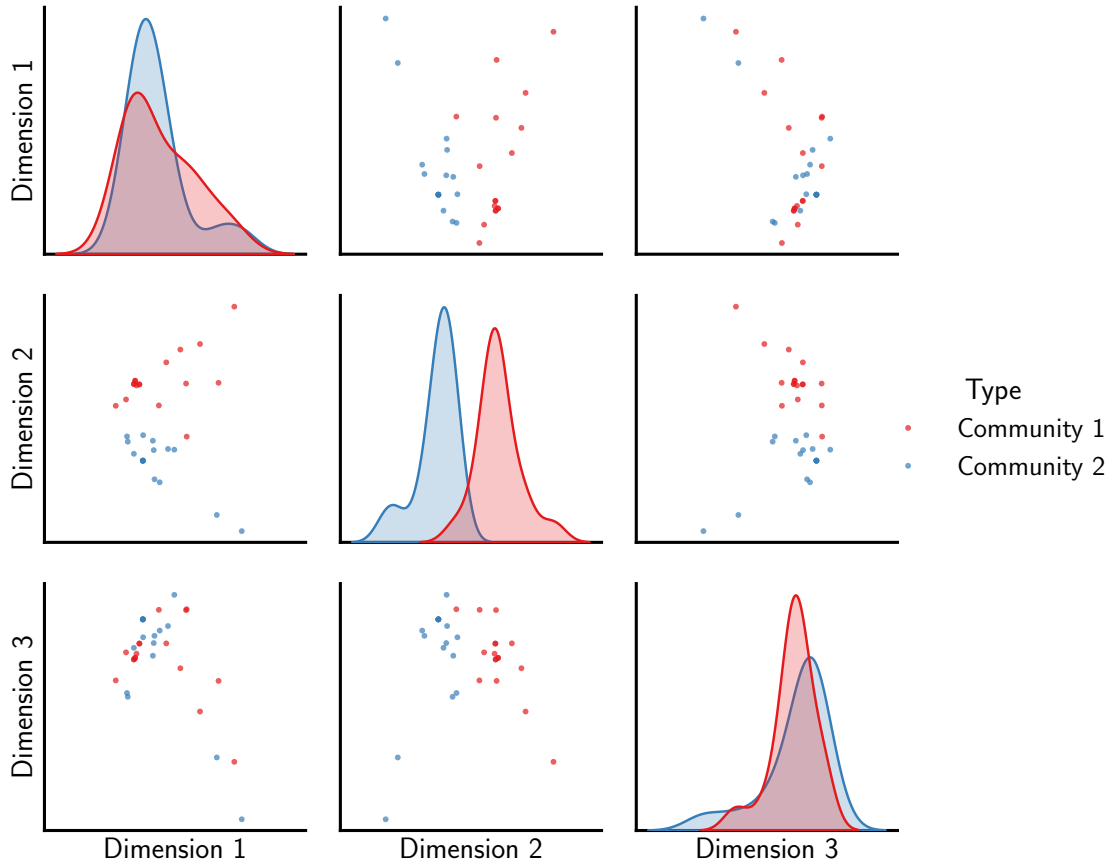

Figure 2.2: Pairwise scatter plots of the top-$d = 3$ generalized RDPG embeddings of the karate club network, with the third coordinate sign-reversed to account for the $(p, q) = (2, 1)$ signature. Node colors indicate the true communities from the original dataset. The second dimension remains the most informative for separating the two communities.

spectral methods in inference tasks, such as clustering or classification, even in the presence of heterophilic or mixed network structures.

By allowing for indefinite inner products, the GRDPG expands the scope of latent position models and enables spectral embedding methods to accommodate a wider variety of network topologies, including those with disassortative or mixed connectivity structures.

While in this thesis we restrict our attention to the standard RDPG model, it is possible to extend many of the results presented here to the broader context of the GRDPG model. In particular, the analysis of spectral embedding methods, consistency results, and inference procedures can be adapted to accommodate the indefinite inner product structure that defines the GRDPG. This typically involves modifying the theoretical analysis to account for the geometry induced by the signature matrix, and carefully handling the model's invariance under indefinite orthogonal transformations.

Chapter 3

Algorithmic advances for the inference problem in RDPGs

3.1 Challenges facing the ASE

Although the ASE is widely adopted and its statistical properties are well documented, it does present drawbacks which we seek to overcome.

Large data. The first challenge pertains to scalability. Computing the spectrum of a large adjacency matrix $\mathbf{A}$, even only the d dominant components, is computationally intensive and constitutes a bottleneck of state-of-the-art ASE implementations (Chung et al., 2019), especially when multiple graphs are to be embedded. Recent work explicitly comments on the difficulty of scaling spectral-based inference of RDPGs to large graph settings (Gallagher et al., 2021).

Missing data. A second drawback of ASE is its inability to properly account for missing data, meaning unobserved entries in $\mathbf{A}$. On a related note, the ASE neglects the all-zeros diagonal in the adjacency matrix. These limitations were recognized more than a decade ago by Scheinerman and Tucker (2010), yet to the best of our knowledge they have not been satisfactorily addressed in the RDPG literature. Indeed, repeated ASE computation to iteratively impute the unknown entries of $\mathbf{A}$ using the inner-product of the embeddings estimated in the previous step lacks convergence guarantees, and multiplies the ASE complexity by the number of iterations (Scheinerman & Tucker, 2010).

Streaming data. A third scenario that ASE cannot address satisfactorily arises with streaming data from a dynamic network; i.e., when we observe a sequence of graphs over time and would like to track the evolution of the corresponding embeddings, ideally without having to store past observations. Network dynamics may include changes in the edges between a fixed set of nodes (e.g., monitoring a wireless network), the addition of new information (e.g., a user that ranks an item

in a recommender system), or the deletion/addition of nodes (e.g., a new user in a social network). Especially for large graphs, re-computing the ASE from scratch each time step is computationally demanding. Given the rotational ambiguity inherent to RDPGs, independently obtaining the ASE after each modification to the graph will likely result in misaligned embeddings that can hinder the assessment of changes.

3.2 Contributions and chapter outline

We seek to address these limitations by (i) re-considering the underlying optimization problem of which ASE is a solution (Section 3.3); and (ii) developing iterative embedding algorithms for the refined formulations (Sections 3.4 and 3.5).

Unlike the traditional ASE approach, which relies on spectral decomposition of $\mathbf{A}$, we adopt a modern perspective inspired by recent advances in low-rank matrix factorization and propose solving the non-convex embedding problem directly via gradient descent (GD) (Luo & Garcia Trillos, 2022; Z. Zhu et al., 2021). Rather than treating the optimization landscape as an obstacle, we contribute a novel and tailored landscape analysis of the embedding objective, detailed in Section 3.4.2. This result is, to our knowledge, new in the context of the RDPG model. It rigorously establishes that the masked objective admits a benign optimization landscape when viewed through the natural quotient Riemannian geometry induced by the problem's orthogonal invariance. While this does not in itself constitute a full proof of global convergence of GD, it provides the key geometric guarantees that underlie such results and marks the first step toward global convergence guarantees in this setting.

Explicitly solving the optimization problem enables more precise and flexible graph representation learning (GRL); for instance, unobserved edges can be easily handled through the inclusion of a mask matrix. The iterative nature of GD also permits warm restarts for newly added or previously seen nodes, allowing for consistent embeddings of multiple–possibly streaming–graphs, with alignment between successive embeddings emerging naturally. Finally, by discarding the diagonal residuals of $\mathbf{A}$, we obtain improved nodal representations and exploit the resulting structure in Section 3.4.4 to derive efficient block-coordinate descent (BCD) updates tailored to the undirected RDPG setting.

Applying GD to embed digraph nodes requires special care. As we argue in Section 3.5.1, inherent ambiguities in the directed RDPG model extend beyond a global rotation, and they may compromise representation quality and the interpretability benefits discussed earlier. We show that an effective way of retaining these desirable features is to impose orthogonality constraints on the matrix factors in the decomposition of $\mathbf{A}$–a novel GRL formulation for digraphs. This constraint in turn

defines a smooth manifold, over which we optimize using a custom-made feasible method. We stress this is not the well-known Stiefel manifold, where matrices are constrained to be *orthonormal* (and not just orthogonal as here[1]). This is no minor point. Algorithm construction thus requires careful definition of the tangent space, the Riemannian gradient and the retraction (Absil et al., 2009; Boumal, 2023), all of which we derive in Section 3.5.2. Comprehensive synthetic and real-world (wireless network and United Nations voting) data experiments in Section 3.6 demonstrate the interpretability, robustness, and versatility of the novel GRL framework. In the interest of reproducible research, the code and datasets used to generate all figures in this chapter is publicly available at https://github.com/marfiori/efficient-ASE.

All in all, relative to prior art our RDPG embedding framework offers a *better* representation at a *competitive* computational cost, and it is applicable to *more general* settings. This chapter is based on joint work previously published in (Fiori et al., 2023).

3.3 Problem statement and related work

Recalling the definition of the RDPG model, the edge-wise formation probabilities are the entries $P_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$ of the rank-d, PSD matrix $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$. Since we do not allow for self-loops, the diagonal entries in the adjacency matrix $\mathbf{A}$ should be zero. Therefore, under the RDPG model have $\mathbb{E}\left[\mathbf{A} \,\middle|\, \mathbf{X}\right] = \mathbf{M} \circ \mathbf{P}$, where $\circ$ is the entry-wise or Hadamard product and $\mathbf{M} = \mathbf{1}_N\mathbf{1}_N^\top - \mathbf{I}_N$ is a mask matrix with ones everywhere except in the diagonal, where it is zero. Revisiting and adapting the legacy ASE, we propose to estimate the latent positions matrix by solving

$$\hat{\mathbf{X}} \in \underset{\mathbf{X}\in\mathbb{R}^{N\times d}}{\operatorname{argmin}} \left\|\mathbf{M} \circ (\mathbf{A} - \mathbf{X}\mathbf{X}^\top)\right\|_F^2. \tag{3.1}$$

Note that entry-wise multiplication with $\mathbf{M} = \mathbf{1}_N\mathbf{1}_N^\top - \mathbf{I}_N$ effectively discards the residuals corresponding to the diagonal entries of $\mathbf{A}$.

It is typical in the literature to ignore the mask $\mathbf{M}$, and even to impute non-zero values to the diagonal of $\mathbf{A}$ (Athreya et al., 2017; Marchette et al., 2011; Scheinerman & Tucker, 2010). This has the advantage of yielding a closed-form solution for $\hat{\mathbf{X}}$ –that is, the ASE. However, the formulation in (3.1) is the correct one under the RDPG model, although it does not admit a closed-form solution. We therefore develop efficient gradient-based iterative solvers for the embedding problem (3.1), which will also allow us to overcome the scalability limitations of the ASE outlined

[1]We will henceforth use the term *orthonormal matrix* to refer to any matrix $\mathbf{T} \in \mathbb{R}^{n\times m}$ such that $\mathbf{T}^\top\mathbf{T} = \mathbf{I}_m$ (i.e., the columns of $\mathbf{T}$ are orthonormal vectors). The term *orthogonal matrix* will be reserved for those matrices whose columns are mutually orthogonal, but not necessarily of unit norm.

in Section 3.1.

Incorporating the binary mask $\mathbf{M}$ has another advantage: if suitably redefined, it can be used for other purposes, such as modeling unknown edges when data are missing. For instance, in a recommender system we typically have the rating of each user over a limited number of items. This (dis)information can be captured in (3.1) by zeroing out the entries of $\mathbf{M}$ corresponding to the unknown edges.

Beyond scalability, the flexibility of our algorithmic framework and the ability to encode missing or uncertain connections via the mask $\mathbf{M}$ also offers a natural pathway to facilitate embedding graph sequences. In the applications we study in Section 3.6, such dynamic network data may be only partially observed, may be acquired in a streaming fashion, and can exhibit variation in both the number of nodes and edges over time.

Remark 3.3.1 (Recovering ASE). If the mask $\mathbf{M}$ is ignored (which is standard in the literature), then a solution to (3.1) is given by the standard ASE from Section 2.2.

3.3.1 Related work

The low-rank matrix factorization problem (3.1) has a long history, with applications to recommender systems–where the objective is to complete a matrix of user-item ratings which is assumed to have low rank (Koren et al., 2009)–; or, in sensor localization from pairwise distances–the so-called Euclidean distance matrix (Dokmanic et al., 2015)–, just to name a couple of examples. Solution methods typically rely on spectral decomposition of the full data matrix (as in ASE), or by considering a convex relaxation via nuclear-norm minimization (Davenport & Romberg, 2016). The latter is not best suited for our problem, where we are interested in the actual factors (not in $\mathbf{P}$), and their dimensionality could change with time due to e.g., node additions. Alternatively, over the last few years we have witnessed increased interest in non-convex optimization approaches for matrix factorization problems (Chi et al., 2019). Our work may be seen as an effort in this direction. In particular, we bring to bear recent advances in first-order methods for matrix factorization problems and demonstrate impact to GRL (specifically, RDPG inference). The formulation (3.17), introduced later to embed directed graphs, is novel to the best of our knowledge. To solve it we derive GD iterations over the manifold of orthogonal matrices, which is different from the Stiefel manifold and thus requires careful treatment given the unique geometric properties of our problem.

The scalability of ASE, or any other spectral embedding method for that matter, has long been considered an issue (Brand, 2006). This challenge is compounded

when multiple graphs are to be embedded, especially in *batch* settings where all graphs in the sequence are stored in memory (Gallagher et al., 2021). Existing approaches seeking aligned embeddings rely on the spectral decomposition of some matrix whose dimension(s) grow linearly with the number of graphs (Gallagher et al., 2021; Jones & Rubin-Delanchy, 2020; Levin et al., 2017). In addition to increasing the computation cost of ASE, these methods are not applicable in streaming scenarios, where a possibly infinite sequence of graphs $\{\mathbf{A}_t\}$ is observed and we want to recursively update the embeddings 'on-the-fly' as new graphs are acquired.

There is an extensive collection of numerical linear algebra approaches to recursively update an eigendecomposition or SVD when the (adjacency) matrix is perturbed; e.g., (Brand, 2006). However, these do not offer major computational savings except for specific types of changes (e.g., rank-1 updates), and they may be prone to error accumulation as t increases (Z. Zhang et al., 2018). Moreover, they can yield misaligned embeddings due to the rotational ambiguity of RDPGs. The sketching literature offers highly-scalable randomized algorithms (Halko et al., 2011). Other than to initialize our iterative methods we do not consider those here, because we are interested in exact solutions to (3.1).

In dynamic environments, not only does $\mathbf{A}_t$ change over time, but new nodes may be added to the graph and current ones removed. Embedding previously unseen nodes corresponds to an inductive learning setting, generally regarded as beyond the capabilities of shallow embeddings as the one we are discussing here (Hamilton (2020, Ch. 3.4), Chami et al. (2022)). Previous efforts in this direction (that avoid re-computing eigendecompositions from scratch) either assume that the connections between the existing nodes, or, their current embeddings, do not change (Kalantzis & Traganitis, 2023; Levin et al., 2018). In the latter case, a projection scheme onto the space of current embeddings produces an asymptotically ($N \to \infty$) consistent ASE for the new node (Levin et al., 2018). However, even if latent positions were time invariant, the estimation error of current nodes' embeddings propagates to the new estimates. We will use the projection-based estimate from Levin et al. (2018) as initialization for new nodes in our GD algorithms, demonstrating benefits in accuracy and stability especially as several nodes are added, while at the same time refining previous nodes' embeddings.

3.4 Embedding algorithms for undirected graphs

We start with the embedding problem for undirected graphs. Recognizing limitations of state-of-the-art ASE implementations, here we first review a GD algorithm with well-documented merits for symmetric matrix completion, yet so far unexplored in RDPG inference. GD offers flexible computation of embeddings and a pathway

Algorithm 1 Gradient Descent (GD)

Require: Initial estimate $\mathbf{X} \leftarrow \mathbf{X}_0$, tolerance $\varepsilon > 0$

repeat
  Compute gradient: $\nabla f(\mathbf{X}) = 4[\mathbf{M} \circ (\mathbf{X}\mathbf{X}^\top - \mathbf{A})]\mathbf{X}$
  Choose step size α (e.g., via Armijo rule)
  Update: $\mathbf{X} \leftarrow \mathbf{X} - \alpha \nabla f(\mathbf{X})$
until stopping criterion satisfied (e.g., $\|\nabla f(\mathbf{X})\|_{\mathrm{F}} \leq \varepsilon$)
return $\mathbf{X}$

towards tracking nodal representations in a streaming graph setting. We then show that the particular structure of the problem lends itself naturally to more efficient BCD iterations, and discuss the relative merits of the different approaches in terms of convergence properties, complexity, and empirical execution time.

3.4.1 Back to basics: Estimation via gradient descent

Recall the embedding problem for undirected graphs in (3.1), and denote by $f : \mathbb{R}^{N\times d} \to \mathbb{R}$ its smooth objective function $f(\mathbf{X}) = \|\mathbf{M} \circ (\mathbf{A} - \mathbf{X}\mathbf{X}^\top)\|_F^2$. Although the problem is not convex with respect to $\mathbf{X}$, it is convex with respect to $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$. In the broad context of matrix factorization problems where the objective function depends on the product $\mathbf{X}\mathbf{X}^\top$, the GD approach is often referred to as *factored GD* (Bhojanapalli et al., 2016).

The workhorse GD algorithm generates embedding updates via

$$\mathbf{X}_{k+1} = \mathbf{X}_k - \alpha_k \nabla f(\mathbf{X}_k), \quad k = 0, 1, 2, \ldots \tag{3.2}$$

where $\alpha_k > 0$ is the step size, which may be fixed or selected adaptively–for instance, using the Armijo backtracking rule. The gradient of the objective is given by

$$\nabla f(\mathbf{X}) = 4\left[\mathbf{M} \circ (\mathbf{X}\mathbf{X}^\top - \mathbf{A})\right]\mathbf{X},$$

where $\mathbf{A}$ and $\mathbf{M}$ are known symmetric matrices that specify the problem instance. A summary of the GD procedure is given in Algorithm 1.

There have been several noteworthy advances in the study of GD's convergence (including rates) for this non-convex setting, as well as accelerated variants (Chi et al., 2019; Luo & Garcia Trillos, 2022; Vu & Raich, 2021; L. Wang et al., 2017; Zhou et al., 2020; Z. Zhu et al., 2021). Building on that literature, we analyze the optimization landscape of our RDPG embedding, first by analyzing it for the unmasked problem (2.2), and then extending that analysis to the more general masked setting (3.1).

3.4.2 Landscape analysis of the embedding problem

In this section, we discuss the landscape of the optimization problem under consideration. We begin by analyzing the ASE optimization problem (2.2)–that is, problem (3.1) without the mask $\mathbf{M}$–and then consider efforts to generalize the results to the masked problem (3.1).

3.4.2.1 Unmasked objective

For now we consider the unmasked the objective

$$f(\mathbf{X}) = \frac{1}{4}\left\|\mathbf{A} - \mathbf{X}\mathbf{X}^\top\right\|_{\mathrm{F}}^2, \tag{3.3}$$

where the factor $\frac{1}{4}$ is included so that the gradient of f is simply $(\mathbf{X}\mathbf{X}^\top - \mathbf{A})\mathbf{X}$. We will assume $\mathbf{A}$ is a PSD matrix, a requirement fullfilled under the RDPG model with high probability (Sussman et al., 2014). The goal is to minimize $f(\mathbf{X})$ over all matrices $\mathbf{X} \in \mathbb{R}^{N\times d}$ with rank d.

Understanding the stationary points of this function helps characterize the optimization landscape–particularly whether spurious local minima or saddle points exist. In that sense, in Appendix 3.A we prove that any critical point of the objective (3.3) is of the form:

$$\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{D}}^{1/2}\mathbf{Q},$$

where $\tilde{\mathbf{U}}$ contains any d orthonormal eigenvectors of $\mathbf{A}$ as columns, $\tilde{\mathbf{D}}$ holds the corresponding eigenvalues in its diagonal, and $\mathbf{Q} \in O(d)$.

Since we have identified all critical points, we next show that the objective (3.3) has no spurious local minima. That is, the only global minimizers are those corresponding to the ASE, up to an orthogonal transformation, and all other stationary points are strict saddle points. This result underscores the effectiveness of GD algorithms for solving the RDPG embedding problem–at least in the unmasked setting.

Proposition 3.4.1. *The global minimizers of the objective (3.3) are precisely the matrices of the form*

$$\mathbf{X} = \sum_{i=1}^{d} \sqrt{\lambda_i}\mathbf{u}_i\mathbf{v}_i^\top = \hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2}\mathbf{Q},$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ are the top d eigenvalues of $\mathbf{A}$, $\hat{\mathbf{U}}$ contains the corresponding eigenvectors as columns, $\hat{\mathbf{D}} = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$, and $\mathbf{Q} \in O(d)$.

Furthermore, any other stationary point of f that is not of the form above is a strict saddle point; that is, the Hessian of f at those points has at least one negative eigenvalue and therefore there exists an escaping direction.

Proof. That $\hat{\mathbf{X}} := \hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2}$, is the best rank-$d$ approximation to $\mathbf{A}$ in Frobenius norm

is a consequence of the Eckart-Young theorem (Eckart & Young, 1936). Since for any matrix of the form $\hat{\mathbf{Y}} := \hat{\mathbf{X}}\mathbf{Q}$, $\mathbf{Q} \in O(d)$, it holds $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^\top = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top$, it follows that $\hat{\mathbf{Y}}$ also minimizes f, which proves the first part of the proposition.

Let $\tilde{\mathbf{X}}$ be a stationary point of f that is not a global minimizer. So

$$\tilde{\mathbf{X}} = \sum_{j \in J} \sqrt{\lambda_j} \mathbf{u}_j \mathbf{v}_j^\top,$$

for some index set $J \neq \{1, \dots, d\}$ with $|J| = d$; that is, $\tilde{\mathbf{X}}$ does not use the top d eigenvectors of $\mathbf{A}$. Therefore, we can choose $k \in \{1, \dots, d\} \setminus J$, corresponding to one of the top d eigenvalues that is not represented in $\tilde{\mathbf{X}}$. Since $|J| = d$, there must exist some $i \in J$ such that $\lambda_i < \lambda_k$; in other words, we pick k to be the index of an unused top eigenvalue and i to be any index currently used by $\tilde{\mathbf{X}}$ whose eigenvalue is smaller than λ_k. Now, define

$$\mathbf{Z} := \sqrt{\lambda_k} \mathbf{u}_k \mathbf{v}_i^\top.$$

We will compute the second-order directional derivative of f at $\tilde{\mathbf{X}}$ in the direction $\mathbf{Z}$. To that end, define the quadratic form

$$\nabla^2 f(\tilde{\mathbf{X}})[\mathbf{Z}, \mathbf{Z}] = \left\langle \nabla^2 f(\tilde{\mathbf{X}})\mathbf{Z}, \mathbf{Z} \right\rangle, \tag{3.4}$$

where $\nabla^2 f(\tilde{\mathbf{X}})$ denotes the Euclidean Hessian of f at $\tilde{\mathbf{X}}$, and $\langle \mathbf{B}, \mathbf{C} \rangle = \mathrm{Tr}(\mathbf{B}^\top \mathbf{C})$ is the Frobenius inner product.

To compute the Euclidean Hessian, observe that

$$\nabla^2 f(\tilde{\mathbf{X}})[\mathbf{Z}] = \frac{d}{dt} \left(\nabla f(\tilde{\mathbf{X}} + t\mathbf{Z}) \right) \Big|_{t=0}.$$

Using the expression $\nabla f(\tilde{\mathbf{X}}) = (\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A})\tilde{\mathbf{X}}$, we differentiate to obtain

$$\nabla^2 f(\tilde{\mathbf{X}})[\mathbf{Z}] = (\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A})\mathbf{Z} + (\tilde{\mathbf{X}}\mathbf{Z}^\top + \mathbf{Z}\tilde{\mathbf{X}}^\top)\tilde{\mathbf{X}}.$$

Substituting this into (3.4), we obtain

$$\nabla^2 f(\tilde{\mathbf{X}})[\mathbf{Z}, \mathbf{Z}] = \left\langle \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A}, \mathbf{Z}\mathbf{Z}^\top \right\rangle + \left\langle \tilde{\mathbf{X}}\mathbf{Z}^\top + \mathbf{Z}\tilde{\mathbf{X}}^\top, \mathbf{Z}\tilde{\mathbf{X}}^\top \right\rangle,$$

due to the cyclic property of the trace.

For the first term, we have

$$\left\langle \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A}, \mathbf{Z}\mathbf{Z}^\top \right\rangle = \left\langle \sum_{j \in J} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top - \sum_{j=1}^{N} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top, \lambda_k \mathbf{u}_k \mathbf{u}_k^\top \right\rangle$$

$$= -\left\langle \sum_{j \notin J} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top, \lambda_k \mathbf{u}_k \mathbf{u}_k^\top \right\rangle = -\lambda_k^2.$$

As for the second term, note that:

$$\tilde{\mathbf{X}}\mathbf{Z}^\top = \sum_{j \in J} \sqrt{\lambda_j} \mathbf{u}_j \mathbf{v}_j^\top \left(\sqrt{\lambda_k} \mathbf{v}_i \mathbf{u}_k^\top \right) = \sqrt{\lambda_i \lambda_k} \mathbf{u}_i \mathbf{u}_k^\top,$$

since $\mathbf{v}_j^\top \mathbf{v}_i = \delta_{ij}$. Similarly,

$$\mathbf{Z}\tilde{\mathbf{X}}^\top = \sqrt{\lambda_i \lambda_k} \mathbf{u}_k \mathbf{u}_i^\top,$$

and hence

$$\left\langle \tilde{\mathbf{X}}\mathbf{Z}^\top + \mathbf{Z}\tilde{\mathbf{X}}^\top, \mathbf{Z}\tilde{\mathbf{X}}^\top \right\rangle = \lambda_i \lambda_k \left\langle \mathbf{u}_i \mathbf{u}_k^\top + \mathbf{u}_k \mathbf{u}_i^\top, \mathbf{u}_k \mathbf{u}_i^\top \right\rangle.$$

Using the definition of the Frobenius inner product:

$$\left\langle \mathbf{u}_i \mathbf{u}_k^\top, \mathbf{u}_k \mathbf{u}_i^\top \right\rangle = \mathrm{Tr}\left(\mathbf{u}_i \mathbf{u}_k^\top \mathbf{u}_i \mathbf{u}_k^\top \right) = \mathrm{Tr}\left(\mathbf{u}_k^\top \mathbf{u}_i \mathbf{u}_k^\top \mathbf{u}_i \right) = (\mathbf{u}_k^\top \mathbf{u}_i)^2 = 0,$$

because $\mathbf{u}_i$ and $\mathbf{u}_k$ are orthogonal. Similarly,

$$\left\langle \mathbf{u}_k \mathbf{u}_i^\top, \mathbf{u}_k \mathbf{u}_i^\top \right\rangle = \mathrm{Tr}(\mathbf{u}_i \mathbf{u}_k^\top \mathbf{u}_k \mathbf{u}_i^\top) = \mathrm{Tr}(\mathbf{u}_k^\top \mathbf{u}_k \mathbf{u}_i^\top \mathbf{u}_i) = \|\mathbf{u}_i\|_2^2 \|\mathbf{u}_k\|_2^2 = 1.$$

Thus,

$$\left\langle \tilde{\mathbf{X}}\mathbf{Z}^\top + \mathbf{Z}\tilde{\mathbf{X}}^\top, \mathbf{Z}\tilde{\mathbf{X}}^\top \right\rangle = \lambda_i \lambda_k.$$

Combining both terms, we have

$$\nabla^2 f(\tilde{\mathbf{X}})[\mathbf{Z}, \mathbf{Z}] = -\lambda_k^2 + \lambda_i \lambda_k = -\lambda_k(\lambda_k - \lambda_i) < 0,$$

which shows that $\tilde{\mathbf{X}}$ is a strict saddle. □

3.4.2.2 Masked objective

We now turn to the problem (3.1), and define the masked objective as

$$f_{\mathbf{M}}(\mathbf{X}) = \frac{1}{4} \left\| \mathbf{M} \circ \left(\mathbf{A} - \mathbf{X}\mathbf{X}^\top \right) \right\|_{\mathrm{F}}^2, \tag{3.5}$$

where $\mathbf{M} = \mathbf{1}_N \mathbf{1}_N^\top - \mathbf{I}_N$ masks out the diagonal. To analyze the optimization landscape of this formulation, we build on the work of Luo and Garcia Trillos (2022) and Z. Zhu et al. (2021), who study a broad class of optimization problems expressible via the Burer–Monteiro factorization (Burer & Monteiro, 2005). These are problems involving PSD matrix variables $\mathbf{P}$ with rank constraints, which are reparameterized

as factorizations of the form $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$. Notably, our problem (3.1) is already naturally expressed in this factored form.

In particular, Luo and Garcia Trillos (2022), building upon the machinery developed by Z. Zhu et al. (2021), analyze the global landscape of optimization problems of the form:

$$\min_{\mathbf{P}\in\mathbb{R}^{N\times N}} g(\mathbf{P}) \quad \text{s. to } \operatorname{rank}(\mathbf{P}) = d \leq N \text{ and } \mathbf{P} \text{ is symmetric and PSD.} \tag{3.6}$$

Since $\mathbf{P}$ is symmetric and PSD, it admits a factorization of the form $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$– this is the so-called Burer–Monteiro factorization. Therefore, problem (3.6) can be expressed as the *factored problem*:

$$\min_{\mathbf{X}\in\mathbb{R}_*^{N\times d}} \bar{h}(\mathbf{X}) \tag{3.7}$$

where $\bar{h}(\mathbf{X}) := h(\mathbf{X}\mathbf{X}^\top)$ and $\mathbb{R}_*^{N\times d}$ denotes the space of $N \times d$ matrices with full column rank. The choice of notation in terms of $\mathbf{P}$ and $\mathbf{X}$ here is intentional: our problem (3.1) can be cast within this framework, but in the reverse direction: whereas Burer-Monteiro methods typically begin with a problem formulated in terms of $\mathbf{P}$ and factor it into $\mathbf{X}$, we start with an objective in $\mathbf{X}$ and consider its interpretation in terms of $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$.

What is particularly interesting about the work of Luo and Garcia Trillos (2022) is that they analyze the landscape of such problems using the Riemannian quotient geometry that arises from the inherent orthogonal ambiguity in the factorization. As in the case of inference under the RDPG model, if $\mathbf{X}$ is a solution to the factored problem (3.7), then so is $\mathbf{X}\mathbf{Q}$ for any orthogonal matrix $\mathbf{Q} \in O(d)$. This symmetry implies that the problem (3.7) is inherently nonconvex in the neighborhood of any stationary point. To overcome this issue, Luo and Garcia Trillos (2022) analyze the landscape of the problem

$$\min_{[\mathbf{X}]\in\mathcal{M}_{d+}^N} h([\mathbf{X}]) \tag{3.8}$$

where $[\mathbf{X}] := \{\mathbf{X}\mathbf{Q} : \mathbf{Q} \in O(d)\}$ is the equivalence class of $\mathbf{X}$ modulo orthogonal transformations, $\mathcal{M}_{d+}^N$ is the quotient manifold $\mathcal{M}_{d+}^N := \mathbb{R}_*^{N\times d}/O(d)$, and $h([\mathbf{X}]) := \bar{h}(\mathbf{X})$. Under certain regularity conditions on the unfactored objective g (made explicit below), they show that the landscape of (3.8) is benign–meaning that, for any $\mathbf{X} \in \mathbb{R}_*^{N\times d}$, at least one of the following properties hold [see (Luo & Garcia Trillos, 2022, Theorem 1)]:

(a) the Riemannian gradient of $h([\mathbf{X}])$ has large magnitude;
(b) the Riemannian Hessian of $h([\mathbf{X}])$ has a large negative eigenvalue, and there exists an explicit direction of negative curvature;

(*c*) if $\mathbf{X}\mathbf{X}^\top$ is sufficiently close to the target matrix $\mathbf{P}^*$, then $h([\mathbf{X}])$ is smooth and geodesically convex.

This has strong implications for the convergence of the GD algorithm when applied to the factored problem (3.7), because the horizontal lift of the Riemannian gradient of $h([\mathbf{X}])$ coincides with the Euclidean gradient of $\bar{h}(\mathbf{X})$ (Luo & Garcia Trillos, 2022, Lemma 3).[1] As a result, performing GD under the Riemannian quotient geometry is computationally equivalent to applying standard GD to the factored objective. This equivalence provides a foundational guarantee that our GD-based approach yields good estimates when solving the ASE problem (3.1).

In order to use Theorem 1 from Luo and Garcia Trillos (2022) we first define the unfactored function associated with our masked objective $f_{\mathbf{M}}$ as:

$$g_{\mathbf{M}}(\mathbf{P}) := \frac{1}{4}\|\mathbf{M} \circ (\mathbf{A} - \mathbf{P})\|_{\mathrm{F}}^2. \tag{3.9}$$

For that theorem to hold we require that $g_{\mathbf{M}}$ is twice continuously differentiable in the usual sense (which it is), and that it satisfies the restricted strong convexity and smoothness properties defined below.

Definition 3.4.1. A function $g : \mathbb{R}^{N\times N} \to \mathbb{R}$ is said to satisfy the $(2r, 4r)$-restricted strong convexity and smoothness properties if there exist a $\delta \in [0, 1)$ such that for all $\mathbf{P}, \mathbf{G} \in \mathbb{R}^{N\times N}$ with $\mathrm{rank}(\mathbf{P}) \leq 2r$ and $\mathrm{rank}(\mathbf{G}) \leq 4r$ the following holds:

$$(1 - \delta)\|\mathbf{G}\|_{\mathrm{F}}^2 \leq \nabla^2 g(\mathbf{P})[\mathbf{G}, \mathbf{G}] \leq (1 + \delta)\|\mathbf{G}\|_{\mathrm{F}}^2, \tag{3.10}$$

where $\nabla^2 g(\mathbf{P})[\mathbf{G}, \mathbf{G}] := \langle \nabla^2 g(\mathbf{P})\mathbf{G}, \mathbf{G}\rangle$, $\nabla^2 g(\mathbf{P})$ is the Euclidean Hessian of g, and $\langle \cdot, \cdot \rangle$ is the Frobenius inner product.

To show that $g_{\mathbf{M}}$ satisfies these properties, note that, by standard computations–similar to those carried out in the proof of Proposition 3.4.1–we have:

$$\nabla^2 g_{\mathbf{M}}(\mathbf{P})\mathbf{G} = \mathbf{M} \circ \mathbf{G} \Rightarrow \nabla^2 g_{\mathbf{M}}(\mathbf{P})[\mathbf{G}, \mathbf{G}] = \|\mathbf{M} \circ \mathbf{G}\|_{\mathrm{F}}^2.$$

Using the definition of our mask $\mathbf{M}$ yields

$$\|\mathbf{M} \circ \mathbf{G}\|_{\mathrm{F}}^2 = \sum_{i\neq j} G_{ij}^2 = \|\mathbf{G}\|_{\mathrm{F}}^2 - \sum_{i=1}^{N} G_{ii}^2 \leq \|\mathbf{G}\|_{\mathrm{F}}^2,$$

[1] Under the Riemannian quotient geometry, the tangent space at any point $[\mathbf{X}]$ can be decomposed as the direct sum of two spaces: a vertical space–which leaves the equivalence class of $\mathbf{X}$ unchanged–and a horizontal one. Any vector in the tangent space of $[\mathbf{X}]$ can be uniquely identified with its horizontal component; that component is what is called the horizonal lift. See (Luo & Garcia Trillos, 2022, Section 1.4) for a more formal discussion on the matter.

where G_{ij} is the (i,j) entry of $\mathbf{G}$. This implies that for any $\delta \in [0,1)$ the upper bound in (3.10) trivially holds for $g_{\mathbf{M}}$.

Regarding the lower bound, note that a general function $g_{\mathbf{M}}$ of the form (3.9) can never satisfy it. Indeed, for any $r > 0$ pick as $\mathbf{G}$ any diagonal matrix with $\operatorname{rank}(\mathbf{G}) \leq 2r$. Then:

$$\nabla^2 g_{\mathbf{M}}(\mathbf{P})[\mathbf{G},\mathbf{G}] = \|\mathbf{M} \circ \mathbf{G}\|_{\mathrm{F}}^2 = 0.$$

It is worth emphasizing that this behavior is consistent with our problem (3.1): for any matrix $\mathbf{P}$, a perturbation in a diagonal direction $\mathbf{G}$ does not affect the objective (3.9), since the mask $\mathbf{M}$ effectively discards the diagonal residuals. We must therefore show that, given any feasible point $\mathbf{X}$ of the factored problem (3.1), GD updates never move in such a direction, unless we are already at a stationary point.

To that end, we analyze the structure of the gradient $\nabla f_{\mathbf{M}}(\mathbf{X}) = \left[\mathbf{M} \circ (\mathbf{X}\mathbf{X}^\top - \mathbf{A})\right]\mathbf{X}$. Picking a direction $\mathbf{Z} \in \mathbb{R}^{N\times d}$, the first-order approximation of how $\mathbf{X}\mathbf{X}^\top$ changes in the direction $\mathbf{Z}$ is:

$$(\mathbf{X}+\eta\mathbf{Z})(\mathbf{X}+\eta\mathbf{Z})^\top = \mathbf{X}\mathbf{X}^\top + \eta(\mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top) + \eta^2\mathbf{Z}\mathbf{Z}^\top \approx \mathbf{X}\mathbf{X}^\top + \eta(\mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top).$$

So, to only affect the diagonal of $\mathbf{X}\mathbf{X}^\top$, the symmetric matrix $\mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top$ must be diagonal. Note that the function $f_{\mathbf{M}}(\mathbf{X})$ is invariant to such small perturbations in $\mathbf{X}$, i.e., if $\mathbf{Z}$ is such that $\mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top$ is diagonal, then $f_{\mathbf{M}}(\mathbf{X}+\eta\mathbf{Z}) = f_{\mathbf{M}}(\mathbf{X})$ for small enough η. Indeed, the directional derivative of f in the direction of $\mathbf{Z}$ vanishes, that is:

$$\frac{d}{d\eta} f_{\mathbf{M}}(\mathbf{X}+\eta\mathbf{Z})\Big|_{\eta=0} = 0.$$

To see that, let us define $\mathbf{X}_\eta := \mathbf{X} + \eta\mathbf{Z}$. Then

$$f_{\mathbf{M}}(\mathbf{X}_\eta) = \frac{1}{4}\left\|\mathbf{M} \circ \left(\mathbf{X}\mathbf{X}^\top - \mathbf{A} + \eta(\mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top) + \eta^2\mathbf{Z}\mathbf{Z}^\top\right)\right\|_{\mathrm{F}}^2.$$

Defining $\mathbf{R} := \mathbf{X}\mathbf{X}^\top - \mathbf{A}, \mathbf{S} := \mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top$, $\mathbf{T} := \mathbf{Z}\mathbf{Z}^\top$, and $\mathbf{F}(\eta) := \mathbf{M} \circ (\mathbf{R} + \eta\mathbf{S} + \eta^2\mathbf{T})$ we have

$$f_{\mathbf{M}}(\mathbf{X}_\eta) = \frac{1}{4}\|\mathbf{F}(\eta)\|_{\mathrm{F}}^2.$$

Therefore

$$\frac{d}{d\eta} f_{\mathbf{M}}(\mathbf{X}+\eta\mathbf{Z}) = \frac{1}{2}\left\langle \mathbf{F}(\eta), \frac{d}{d\eta}\mathbf{F}(\eta)\right\rangle \Rightarrow \frac{d}{d\eta} f_{\mathbf{M}}(\mathbf{X}+\eta\mathbf{Z})\Big|_{\eta=0} = \frac{1}{2}\left\langle \mathbf{M} \circ \mathbf{R}, \mathbf{M} \circ \mathbf{S}\right\rangle.$$

Since for the chosen $\mathbf{Z}$, $\mathbf{S} = \mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top$ is diagonal, and $\mathbf{M}$ is zero on the diagonal,

we have $\mathbf{M} \circ \mathbf{S} = \mathbf{0}$ and:

$$\frac{d}{d\eta} f_{\mathbf{M}}(\mathbf{X} + \eta \mathbf{Z})\Big|_{\eta=0} = 0.$$

So any such direction $\mathbf{Z}$ must lie in the nullspace of the differential of f. But the gradient of f is always orthogonal to the nullspace of the differential, meaning that

$$\langle \nabla f_{\mathbf{M}}(\mathbf{X}), \mathbf{Z} \rangle = 0 \quad \text{for all directions } \mathbf{Z} \text{ such that } \mathbf{X}\mathbf{Z}^\top + \mathbf{Z}\mathbf{X}^\top \text{ is diagonal.}$$

Therefore, GD will never produce updates in those directions. We will next show that, outside of this degenerate subspace, the function $g_{\mathbf{M}}$ does satisfy the required curvature properties (3.10).

In particular, to establish restricted strong convexity and smoothness for $g_{\mathbf{M}}$ under the RDPG model, we assume that the latent positions are sampled i.i.d. from a distribution F whose second moment matrix is full rank. This assumption is standard in the RDPG literature (see, for instance, (Athreya et al., 2017; Sussman et al., 2014)) and is necessary for the asymptotic consistency and normality of the ASE embeddings.

Assumption 1. Let F be a probability distribution in $\mathbb{R}^d$ such that $\mathbf{x}^\top \mathbf{y} \in [0, 1]$ for all $\mathbf{x}, \mathbf{y}$ in its support. Then, for any $\mathbf{x} \sim F$, the second moment matrix $\mathbf{\Delta} = \mathbb{E}\left[\mathbf{x}\mathbf{x}^\top\right]$ has full rank d.

If the adjacency matrix $\mathbf{A}$ is sampled from an RDPG model whose latent positions are drawn i.i.d. from a distribution F satisfying Assumption 1, then the nonzero eigenvalues of $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$ are, with high probability, of order N.[1] This result was first established by Sussman et al. (2014, Proposition 4.3); we also provide an analogous proof for a weighted extension of the RDPG model in Appendix 4.A.

Since under the RDPG model we have that $\mathbf{X} = \mathbf{U}_{\mathbf{P}} \mathbf{D}_{\mathbf{P}}^{1/2} \mathbf{Q}$ for some $\mathbf{Q} \in O(d)$, and the entries in the diagonal of $\mathbf{D}_{\mathbf{P}}$ are nonzero with high probability, we have

$$\begin{aligned} \mathbf{U}_{\mathbf{P}} = \mathbf{X}\mathbf{Q}^\top \mathbf{D}_{\mathbf{P}}^{-1/2} \overset{(a)}{\Rightarrow} \|\mathbf{U}_{\mathbf{P}}\|_{2\to\infty} &\leq \|\mathbf{X}\|_{2\to\infty} \left\|\mathbf{Q}^\top \mathbf{D}_{\mathbf{P}}^{-1/2}\right\|_2 \\ &\overset{(b)}{\leq} \|\mathbf{X}\|_{2\to\infty} \left\|\mathbf{Q}^\top\right\|_2 \left\|\mathbf{D}_{\mathbf{P}}^{-1/2}\right\|_2 \\ &\overset{(c)}{\leq} \|\mathbf{X}\|_{2\to\infty} \left\|\mathbf{D}_{\mathbf{P}}^{-1/2}\right\|_2, \end{aligned} \tag{3.11}$$

where (a) is due to the property $\|\mathbf{A}\mathbf{B}\|_{2\to\infty} \leq \|\mathbf{A}\|_{2\to\infty} \|\mathbf{B}\|_2$, (b) is due to the fact that the matrix spectral norm is submultiplicative, and (c) follows because $\mathbf{Q} \in O(d)$, so $\left\|\mathbf{Q}^\top\right\|_2 = \|\mathbf{Q}\|_2 = 1$. Since the nonzero eigenvalues of $\mathbf{P}$ are $\Theta_{\mathbb{P}}(N)$, we have $\left\|\mathbf{D}_{\mathbf{P}}^{-1/2}\right\|_2 = \Theta_{\mathbb{P}}\left(N^{-1/2}\right)$. Recalling that the $\|\mathbf{X}\|_{2\to\infty}$ norm is the maximum

[1] Remember that by this we mean that $\lambda_i(\mathbf{P}) = \Theta_{\mathbb{P}}(N)$ for all $i = 1, \ldots, d$, where $\lambda_1(\mathbf{P}) \geq \lambda_2(\mathbf{P}) \geq \ldots \lambda_d(\mathbf{P}) \geq 0$ are the largest d eigenvalues of $\mathbf{P}$.

Euclidean norm of the rows of $\mathbf{X}$, and noting that these must be bounded by 1 to ensure that all inner products between latent positions lie in $[0, 1]$, (3.11) then yields $\|\mathbf{U_P}\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$.

We will use this to show that our masked unfactored function $g_{\mathbf{M}}$ in (3.9) satisfies the lower bound on the restricted strong convexity and smoothness properties definition (3.10). Recall that we had:

$$\nabla^2 g_{\mathbf{M}}(\mathbf{P})[\mathbf{G}, \mathbf{G}] = \|\mathbf{M} \circ \mathbf{G}\|_{\mathrm{F}}^2 = \|\mathbf{G}\|_{\mathrm{F}}^2 - \sum_{i=1}^{N} G_{ii}^2. \tag{3.12}$$

Since any perturbation in the feasible space $\mathbf{G}$ is a symmetric matrix, we can spectrally decompose it as $\mathbf{G} = \mathbf{U_G D_G U_G^\top}$. Therefore:

$$G_{ii} = \mathbf{e}_i^\top \mathbf{G} \mathbf{e}_i = \mathbf{e}_i^\top \mathbf{U_G D_G U_G} \mathbf{e}_i = \sum_{l=1}^{N} \lambda_l (\mathbf{u}_l^\top \mathbf{e}_i)^2$$

where $\mathbf{e}_i$ is the i-th vector in the standard basis of $\mathbb{R}^d$, λ_l is the l-th eigenvalue of $\mathbf{G}$, and $\{\mathbf{u}_l\}$ are the corresponding orthonormal eigenvectors. Letting λ_1 denote the largest eigenvalue, we get

$$G_{ii} \leq \lambda_1 \sum_{l=1}^{N} (\mathbf{u}_l^\top \mathbf{e}_i)^2 = \lambda_1 \left\|\mathbf{U_G^\top} \mathbf{e}_i\right\|_2^2 \leq \lambda_1 \|\mathbf{U_G}\|_{2\to\infty}^2,$$

where the last inequality is because $\mathbf{U_G^\top} \mathbf{e}_i$ equals the i-th row of $\mathbf{U_G}$. Now, if $\mathbf{G}$ stems from an RDPG model, we have that $\|\mathbf{U_G}\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$, so with high probability $|G_{ii}| \leq C\lambda_1 N^{-1}$ for some $C > 0$, and therefore

$$\sum_{i=1}^{N} G_{ii}^2 \leq C\lambda_1^2 N^{-1}, \tag{3.13}$$

with high probability. Now, since $\|\mathbf{G}\|_{\mathrm{F}}^2 = \sum_{i=1}^{N} \lambda_i^2$, we have $\lambda_1^2 \leq \|\mathbf{G}\|_{\mathrm{F}}^2$, so combining (3.12) and (3.13) yields:

$$\nabla^2 g_{\mathbf{M}}(\mathbf{P})[\mathbf{G}, \mathbf{G}] \leq \|\mathbf{G}\|_{\mathrm{F}}^2 \left(1 - \frac{C}{N}\right) \text{ with high probability.}$$

Therefore, provided N is sufficiently large, we can choose $\delta = \frac{C}{N} \in [0, 1)$ so that $g_{\mathbf{M}}$ satisfies the restricted strong convexity and smoothness properties (3.10). This, combined with the result of Luo and Garcia Trillos (2022, Theorem 1), guarantees a benign optimization landscape for our problem (3.1), thereby justifying the use of GD to solve it.

3.4.3 A local convergence result for general masks

Our previous analysis relies on the mask matrix having the specific form $\mathbf{M} = \mathbf{1}_N\mathbf{1}_N^\top - \mathbf{I}_N$. As mentioned earlier, redefining $\mathbf{M}$ also enables us to handle missing data in the observed adjacency matrix. In this more general setting, however, the benign landscape guarantees for the GD algorithm no longer apply. Nevertheless, by building on prior work, we can still ensure local convergence of the resulting estimates.

The next proposition asserts that if the initial condition $\mathbf{X}_0$ is close to the solution of (3.1), the GD iteration (3.2) converges linearly to $\hat{\mathbf{X}}$; see (Bhojanapalli et al., 2016, Corollary 7), (Vu & Raich, 2021, Theorem 1), as well as (Chi et al., 2019, Lemma 4) and references therein for a similar version of this proposition.

Proposition 3.4.2. *Let $\hat{\mathbf{X}}$ be a solution of (3.1). Then there exist $\delta > 0$ and $0 < \kappa < 1$ such that, if $d(\mathbf{X}_0, \hat{\mathbf{X}}) \leq \delta$, we have*

$$d(\mathbf{X}_k, \hat{\mathbf{X}}) \leq \delta\kappa^k, \quad \forall\, k > 0 \tag{3.14}$$

where $\{\mathbf{X}_k\}$ are GD iterates (3.2) with appropriate constant stepsize, and $d(\mathbf{X}, \hat{\mathbf{X}}) := \min_{\mathbf{Q}} \left\|\mathbf{X}\mathbf{Q} - \hat{\mathbf{X}}\right\|_F^2$ s. to $\mathbf{Q} \in O(d)$ is a matrix distance accounting for the rotational invariance.

Although there are specific $\mathbf{X}_0$ which correspond to sub-optimal stationary points, in our experience GD converges to the global optimum when initialized randomly. For further details on the initialization of factored GD, strong convexity assumptions, and the choice of the stepsize α, see e.g., (Bhojanapalli et al., 2016, Section 5).

Remark 3.4.1 (Warm restarts to embed multiple graphs). On top of being flexible to handle missing data encoded in $\mathbf{M}$, this approach also allows to track the latent positions of a graph sequence $\{\mathbf{A}_t\}$ using warm restarts. That is, after computing the embeddings $\mathbf{X}_t$ of the graph with adjacency matrix $\mathbf{A}_t$, we initialize the GD iterations (3.2) with $\mathbf{X}_t$ to compute $\mathbf{X}_{t+1}$ corresponding to $\mathbf{A}_{t+1}$. This way, unless the graphs at times t and $t+1$ are uncorrelated, the embedding of each node will transition smoothly to its new position (up to noise). Moreover, if the embeddings of the graph at time $t+1$ are sufficiently close to the embeddings at time t, say for slowly time-varying graphs where $d(\mathbf{X}_t, \mathbf{X}_{t+1}) \leq \delta$, then the GD iterates for computing $\mathbf{X}_{t+1}$ also have the same convergence guarantees and rates $(\delta\kappa^k)$, by virtue of Proposition 3.4.2. This was also observed empirically; indeed, experiments demonstrating this so-called longitudinal stability property (Gallagher et al., 2021) are presented in Sections 3.6.3 and 3.6.4.

3.4.4 Block coordinate descent

Here we develop a block-coordinate descent (BCD) method for updating GD iterates (3.2) for our problem (3.1), an alternative to GD which turns out to be quite efficient. The algorithm updates one row of $\mathbf{X}$ at a time in a cyclic fashion, by minimizing the objective function $f_{\mathbf{M}}$ with respect to the corresponding row (treating all other entries of $\mathbf{X}$ as constants, evaluated at their most recent updates). In general, this row-wise sub-problem may not admit a simple solution; however, we show that due to the structure of the mask matrix $\mathbf{M}$ the updates are given in closed form.

Dropping the subscript, let $f_(\mathbf{X}) = \|\mathbf{M} \circ (\mathbf{A} - \mathbf{X}\mathbf{X}^\top)\|_F^2$ and recall $\mathbf{x}_i^\top$ is the i-th row of $\mathbf{X}$. The gradient $\nabla_i f$ of f with respect to $\mathbf{x}_i$ is

$$\begin{aligned} \nabla_i f(\mathbf{X}) &= \big(-(\mathbf{M} \circ \mathbf{A})_i \mathbf{X} + ((\mathbf{M} \circ \mathbf{X}\mathbf{X}^\top)\mathbf{X})_i\big)^\top \\ &= -\mathbf{X}^\top (\mathbf{M} \circ \mathbf{A})_i^\top + \mathbf{X}^\top (\mathbf{M} \circ \mathbf{X}\mathbf{X}^\top)_i^\top, \end{aligned} \tag{3.15}$$

where $(\cdot)_i$ stands for the i-th row of the matrix argument.

Note that since the graph has no self-loops (i.e., the diagonal entries of $\mathbf{A}$ are zero), then the entry-wise product of $\mathbf{A}$ with $\mathbf{M}$ is vacuous over the diagonal. Also because $A_{ii} = 0$, the term $\mathbf{X}^\top (\mathbf{M} \circ \mathbf{A})_i^\top = \mathbf{X}^\top (\mathbf{A})_i^\top$ in (3.15) does not depend on $\mathbf{x}_i$. More importantly, $\mathbf{X}\mathbf{X}^\top$ clearly depends on $\mathbf{x}_i$, and this would challenge solving $\nabla_i f(\mathbf{X}) = \mathbf{0}_d$ to obtain a minimizer [due to the trilinear form of the second term in (3.15)]. However, a close re-examination of $\mathbf{X}^\top (\mathbf{M} \circ \mathbf{X}\mathbf{X}^\top)_i^\top$ suggests this purported challenge can be overcome. First, observe that

$$(\mathbf{M} \circ \mathbf{X}\mathbf{X}^\top)_i = \mathbf{x}_i^\top \mathbf{X}^\top - \mathbf{r}_i^\top,$$

where $\mathbf{r}_i \in \mathbb{R}^N$ is a column vector with zeros everywhere except in entry i, where it takes the value $\mathbf{x}_i^\top \mathbf{x}_i$. Hence,

$$\mathbf{X}^\top (\mathbf{M} \circ \mathbf{X}\mathbf{X}^\top)_i^\top = \mathbf{X}^\top (\mathbf{X}\mathbf{x}_i - \mathbf{r}_i) = (\mathbf{X}^\top \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^\top)\mathbf{x}_i.$$

All in all, we have a simplified expression for the gradient

$$\nabla_i f(\mathbf{X}) = -\mathbf{X}^\top (\mathbf{A}_i)^\top + (\mathbf{X}^\top \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^\top)\mathbf{x}_i. \tag{3.16}$$

Now, define $\mathbf{R} = \mathbf{X}^\top \mathbf{X} - \mathbf{x}_i \mathbf{x}_i^\top$ and notice this matrix does not depend on $\mathbf{x}_i$. Therefore, from (3.16) it follows that the equation $\nabla_i f(\mathbf{x}_i) = \mathbf{0}_d$ is *linear* in $\mathbf{x}_i$, namely $\mathbf{R}\mathbf{x}_i = \mathbf{X}^\top (\mathbf{A}_i)^\top$. The pseudo-code of the algorithm is tabulated under Algorithm 2.

Algorithm 2 Block coordinate descent (BCD)

Require: Initial $\mathbf{X} \leftarrow \mathbf{X}_0$, tolerance $\varepsilon > 0$
Compute $\mathbf{R} = \mathbf{X}^\top \mathbf{X}$
repeat
  for $i = 1 : N$ **do**
    $\mathbf{R} \leftarrow \mathbf{R} - \mathbf{x}_i \mathbf{x}_i^\top$
    $\mathbf{b} \leftarrow \mathbf{X}^\top (\mathbf{A}_i)^\top$
    $\mathbf{x}_i \leftarrow$ solution of $\mathbf{R}\mathbf{x} = \mathbf{b}$
    $\mathbf{R} \leftarrow \mathbf{R} + \mathbf{x}_i \mathbf{x}_i^\top$
  end for
until stopping criterion satisfied (e.g., $\|\nabla f(\mathbf{X})\|_{\mathrm{F}} \leq \varepsilon$)
return $\mathbf{X}$.

The $d \times d$ matrix $\mathbf{R}$ is invertible provided that $\mathbf{X}$ has rank d. This also implies that the row-wise sub-problem has a unique minimizer, which is key to establish convergence of BCD to a stationary point (Bertsekas, 1999, Prop. 2.7.1). It is worth reiterating that this favorable linear structure is lost in the absence of a mask matrix $\mathbf{M}$ (cf. ASE in (2.2)). Since in RDPG embeddings we typically have $d \ll N$, solving multiple $d \times d$ linear systems is affordable; especially when compared to matrix-vector operations of order $\Theta(N^\gamma)$, $\gamma > 1$, like in GD.

3.4.5 Complexity and execution time analyses

We compare four computational methods to obtain RDPG embeddings of undirected graphs: the ASE based on (i) full eigendecomposition, and (ii) truncated SVD as implemented in Python's `Graspologic` (Chung et al., 2019); (iii) GD initialized with the randomized-SVD (RSVD) (Halko et al., 2011) (we account for the RSVD in the execution time); and (iv) randomly initialized BCD as in Algorithm 2.

The full eigendecomposition of $\mathbf{A}$ has worst-case $\Theta(N^3)$ complexity, while for sparse graphs the d-dominant components can be obtained with $\Theta(Nd)$ per-iteration cost. For GD, the per-iteration computational cost incurred to evaluate $\nabla f(\mathbf{X})$ is dominated by the matrix multiplications, which is $\Theta(N^2 d)$ for a naïve implementation. The number of iterations depends on $\mathbf{X}_0$, but even with a favorable initialization the runtime is still higher than the $\Theta(Nd)$ of truncated SVD-based ASE. A refined convergence-rate analysis of GD for the symmetric matrix completion problem is presented in (Vu & Raich, 2021).

Although in general it is tricky to compare the complexity of GD against BCD approaches, we can evaluate the per-iteration computational cost of both methods (for BCD this means a whole cycle over the rows of $\mathbf{X}$ is completed). In both cases, each entry of the matrix $\mathbf{X}$ is updated exactly once. Each cycle consist of N instances of $d \times d$ linear systems, so this is $\Theta(Nd^3)$ in the worst case. In addition,

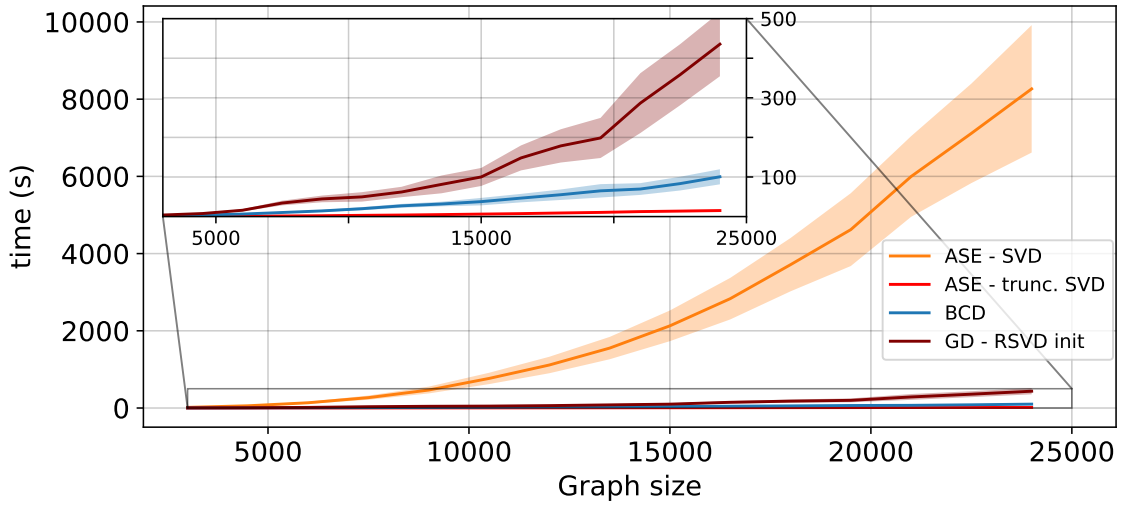

Figure 3.1: Execution time for embedding SBM graphs with up to $N = 24000$ nodes. As N grows, BCD exhibits competitive scaling to the state-of-the-art ASE algorithm implemented in the `Graspologic` package.

in our experience Algorithm 2 converges in fewer iterations than the GD method.

In Fig. 3.1 we compare the execution times of methods (i)-(iv) as a function of N, all the way to $N = 24000$. For ASE, we use the `SciPY` optimized implementation of the eigendecomposition in Python, as in state-of-the-art RDPG inference packages such as `Graspologic`. Our GD and BCD implementations are in pure Python, not optimized for performance. For each N, we sampled several 2-block SBM graphs, with connection probabilities of $p = 0.5$ (within block) and $q = 0.2$ (across blocks). Community sizes are $N/3$ and $2N/3$. We let $d = 2$ in all cases. Results are averaged over 10 Monte Carlo replicates, and corresponding standard deviations are depicted in Fig. 3.1. In all cases, the methods converge to a solution of (3.1). The obtained cost function is very similar for each run, with slightly lower values for the GD and BCD methods because they are solving the problem with the zero-diagonal restriction. As expected, BCD exhibits competitive scaling with the truncated SVD-based ASE, and can embed graphs with $N = 20000$ nodes in just over a minute.

A moderately large graph, such as the one with $N = 24000$, is ideal to assess the effect of d on the computation time. Graphs of this scale are expected to have several communities, and thus values larger than $d = 2$ (as before) will likely be required. We thus explore this scenario in more detail, embedding a d-block SBM graph using d dimensions, and measure the execution time of BCD (Algorithm 2) and the truncated SVD methods as d increases. Results are reported in Table 3.1. Interestingly, BCD yields faster results than truncated SVD when $d \geq 50$, gracefully scaling with the embedding dimension. As we mentioned before, in our experience BCD converges in very few iterations and offers competitive computation performance.

	$d = 10$	$d = 50$	$d = 100$
BCD - Algorithm 2	67 ± 4	98 ± 9	154 ± 32
ASE - Truncated SVD	14 ± 1	247 ± 21	466 ± 47

Table 3.1: Execution time (in seconds) as a function of the embedding dimension d, for d-block SBM graphs with $N = 24000$ nodes.

3.5 Embedding algorithms for digraphs

Shifting gears to embedding nodes in digraphs, we start with a close examination of the ambiguities inherent to the directed RDPG model and justify the need for orthogonality constraints on the factors' columns. To compute the desired nodal representations, we then develop a first-order feasible optimization method in the manifold of matrices with orthogonal columns.

Recall that we now embed each node with two vectors, $\mathbf{x}_i^l, \mathbf{x}_i^r \in \mathcal{X} \subset \mathbb{R}^d$. Existence of a directed edge from node i to j is modeled as the outcome of a Bernoulli trial with success probability $(\mathbf{x}_i^l)^\top \mathbf{x}_j^r$ (Athreya et al., 2017). Again, vertically stacking the embeddings into two matrices $\mathbf{X}^l, \mathbf{X}^r \in \mathbb{R}^{N \times d}$, we introduce the probability matrix $\mathbf{P} = \mathbf{X}^l (\mathbf{X}^r)^\top$ such that the expected value of the random adjacency matrix is $\mathbb{E}\left[\mathbf{A} \,\middle|\, \mathbf{X}^l, \mathbf{X}^r\right] = \mathbf{M} \circ \mathbf{P}$.

If we ignore the mask $\mathbf{M}$, the embedding problem boils down to finding the best rank-d approximation to the (possibly asymmetric) adjacency matrix. Recall that one such solution may be obtained via the SVD of $\mathbf{A}$; i.e., $\mathbf{A} = \mathbf{U}_\mathbf{A} \mathbf{D}_\mathbf{A} \mathbf{V}_\mathbf{A}^\top$. We thus have that $\hat{\mathbf{X}}^l = \hat{\mathbf{U}} \hat{\mathbf{D}}^{1/2}$ and $\hat{\mathbf{X}}^r = \hat{\mathbf{V}} \hat{\mathbf{D}}^{1/2}$, with $\hat{\mathbf{D}}$ containing only the d largest singular values, and $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ the associated singular vectors.

Remark 3.5.1 (Identifiability of directed RDPGs). As noted in Marenco et al. (2022), ambiguities with directed RDPGs can be more problematic than in the undirected case. Now given *any* invertible matrix $\mathbf{T}$ (not necessarily orthonormal), the embeddings $\mathbf{Y}^l = \mathbf{X}^l \mathbf{T}$ and $\mathbf{Y}^r = \mathbf{X}^r \mathbf{T}^{-\top}$ generate the same probability matrix as before; i.e., $\mathbf{Y}^l (\mathbf{Y}^r)^\top = \mathbf{X}^l \mathbf{T} (\mathbf{X}^r \mathbf{T}^{-\top})^\top = \mathbf{X}^l (\mathbf{X}^r)^\top = \mathbf{P}$. As we show in Section 3.5.1, constraining matrices $\mathbf{X}^l$ and $\mathbf{X}^r$ to being orthogonal and having the same column-wise norms[1] effectively limits this ambiguity without compromising the model's expressivity (now an admissible $\mathbf{T}$ may only be orthonormal; see Proposition 3.5.1), all while preserving its interpretability. Given these considerations, our approach to embedding digraphs is to solve the following manifold-constrained op-

[1] Let $\bar{\mathbf{x}}_i^l, \bar{\mathbf{x}}_i^r \in \mathbb{R}^N$ be the i-th columns of $\mathbf{X}^l$ and $\mathbf{X}^r$, respectively. When we say $\mathbf{X}^l$ and $\mathbf{X}^r$ have the same column-wise norms we mean that $\|\bar{\mathbf{x}}_i^l\|_2 = \|\bar{\mathbf{x}}_i^r\|_2$ holds for all $i = 1, \ldots, d$.

timization problem

$$\{\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r\} \in \underset{\{\mathbf{X}^l, \mathbf{X}^r\} \in \mathbb{R}^{N \times d}}{\operatorname{argmin}} \left\| \mathbf{M} \circ (\mathbf{A} - \mathbf{X}^l (\mathbf{X}^r)^\top) \right\|_F^2$$
$$\text{s. to } (\mathbf{X}^l)^\top \mathbf{X}^l = (\mathbf{X}^r)^\top \mathbf{X}^r \text{ diagonal.} \tag{3.17}$$

In the absence of a mask, a solution of (3.17) is the legacy ASE. Indeed, $\hat{\mathbf{X}}^l$ and $\hat{\mathbf{X}}^r$ are obtained from orthonormal singular vectors and have the same column-wise norms because both $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ are right-multiplied by $\hat{\mathbf{D}}^{1/2}$. To tackle the general case, a novel Riemannian GD method over the manifold of matrices with orthogonal columns is developed in Section 3.5.2.

3.5.1 On the interpretability of the directed RDPG

Let us now discuss why $\mathbf{X}^l$ and $\mathbf{X}^r$ should be constrained to be orthogonal and have the same column-wise norms–in other words, why do we need the constraints in (3.17) to obtain useful embeddings when the graph is directed.

To gain some insights, suppose we ignore these constraints altogether and use GD to minimize $f(\mathbf{X}^l, \mathbf{X}^r)$. Similar to (3.2), at iteration $k+1$ we update $\{\mathbf{X}^l_{k+1}, \mathbf{X}^r_{k+1}\}$ as follows

$$\mathbf{X}^l_{k+1} = \mathbf{X}^l_k - \alpha \nabla_{\mathbf{X}^l} f(\mathbf{X}^l_k, \mathbf{X}^r_k), \tag{3.18}$$
$$\mathbf{X}^r_{k+1} = \mathbf{X}^r_k - \alpha \nabla_{\mathbf{X}^r} f(\mathbf{X}^l_k, \mathbf{X}^r_k), \tag{3.19}$$

where $\nabla f_{\mathbf{X}^l}(\mathbf{X}^l, \mathbf{X}^r) = 4 \left[\mathbf{M} \circ (\mathbf{X}^l (\mathbf{X}^r)^\top - \mathbf{A})\right] \mathbf{X}^l$ and a similar expression holds for $\nabla f_{\mathbf{X}^r}(\mathbf{X}^l, \mathbf{X}^r)$.

The ASE offers an alternative baseline, which requires to discard the mask $\mathbf{M}$. ASE estimates $\{\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r\}$ have orthogonal columns because they are derived from the SVD of $\mathbf{A}$. Same index columns in $\hat{\mathbf{X}}^l$ and $\hat{\mathbf{X}}^r$ have the same norm as well, since the orthonormal matrices $\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ are *both* right-multiplied by $\hat{\mathbf{D}}^{1/2}$. However, if we minimize $f(\mathbf{X}^l, \mathbf{X}^r)$ iteratively as in (3.18)-(3.19) to accommodate missing and streaming data, we may loose column-wise orthogonality with detrimental consequences we illustrate in the following example.

Example 3.5.1 (Bipartisan senate). Consider a synthetic political dataset of votes cast by senators in support of laws, over a certain period of time. This may be regarded as a bipartite digraph where nodes correspond to senators and laws, and the fact that senator i has voted affirmatively for law j is indicated by the existence of edge (i, j). We examine a polarized scenario, where two political parties

put forth laws for voting. Affirmative votes are more likely for senators from the party that introduced the law, and less likely for senators from the opposing party. There are also a few bipartisan laws, for which affirmative votes tend to be more balanced across parties. We simulated such a graph with 50 senators of each party, where Party 1 proposed 50 laws and Party 2 proposed 200 laws, and there were 40 additional bipartisan laws under consideration (i.e., $N = 390$ in total). Furthermore, the inter-community block probability matrix is

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0.9 & 0.01 & 0.2 \\ 0 & 0 & 0.1 & 0.8 & 0.3 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

where communities are ordered as Party 1 senator, Party 2 senator, Party 1 law, Party 2 law, and finally bipartisan law. In other words, senators of Party 1 are very supportive of their own laws and unlikely to vote for those introduced in the other side of aisle, whereas Party 2 senators are more moderate.

We compare the embeddings estimated through ASE and by GD [i.e., iterating (3.18) and (3.19) until convergence]. Recall that interpretation of these results should rely on the geometry induced by the RDPG model. Similarity among nodes is captured by their colinearity in latent space, not by their Euclidean distance being small –as it is the case with e.g., Laplacian eigenmaps (Belkin & Niyogi, 2001). Accordingly, in this particular example we expect that the embeddings of Party 1 senators and laws will be almost perfectly aligned, while slightly less so for Party 2. ASE results using $d = 2$ are shown in Figure 3.2 (left). As expected, the outward embeddings for laws and inward embeddings for senators are zero (since the former do not vote and the latter are not voted). Furthermore, the outward embeddings corresponding to senators of each party are close and roughly orthogonal to senators of the other party, reflecting the polarized landscape. Finally, the inward embeddings of laws submitted by each party are aligned with the corresponding cluster of senators, whereas bipartisan laws lie somewhere between both groups. The difference in magnitude between embeddings of senators and laws is due to the different number of such nodes in the graph, and the column-wise norm constraint imposed to the embeddings.

On the other hand, inspection of Fig. 3.2 (right) reveals that GD converges to a solution where laws are not aligned with the corresponding party senators. Accordingly, the affinity of parties to their laws is less evident than before. In fact, it appears as if Party 1 is not as supportive of its laws as in the ASE-based visualization which, as we discussed before, is the opposite of what we expected.

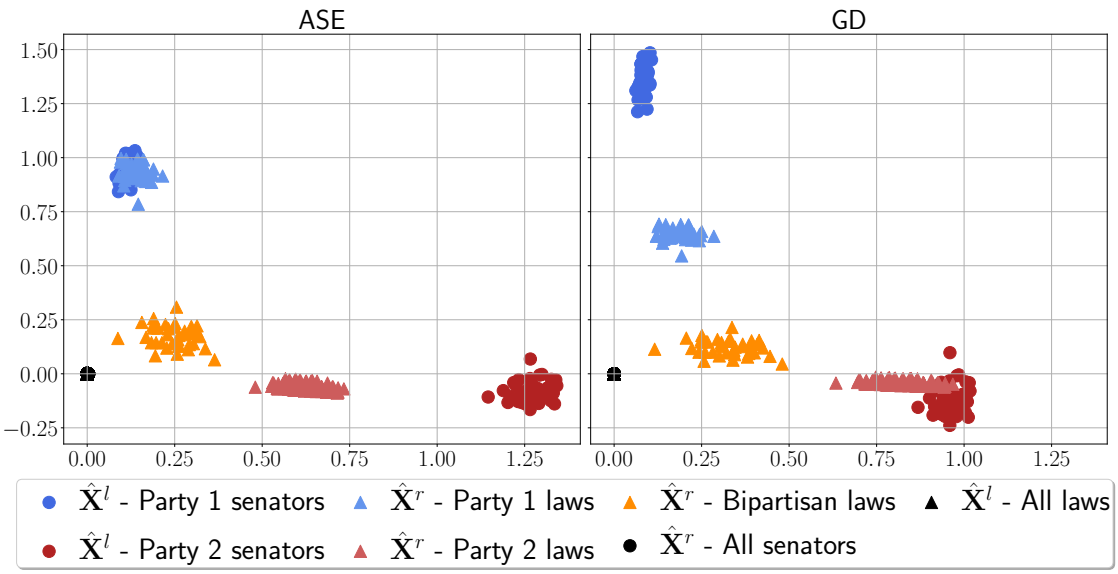

Figure 3.2: Bipartisan senate example. ASE (left) and GD (right). Since ASE implicitly imposes equally normed orthogonal columns (as it is derived from an SVD), it produces interpretable embeddings. On the other hand, GD may result in laws and parties that are not aligned, and thus loses interpretability if no further constrains are imposed in the formulation.

While the input graph is the same for both methods and the total cost $f(\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r)$ is smaller for the GD method, interpretability is hindered because of the larger ambiguity set in the absence of additional constraints.

Example 3.5.2 (Digraph with symmetric expectation). To further justify why orthogonality constraints are essential, consider a digraph sampled from a symmetric $\mathbf{P}$ (i.e., the probability of both edge directions is the same, but each arc is independently sampled). It would be desirable that in this case the model enforced $\mathbf{X}^l = \mathbf{X}^r$, since the outgoing and incoming behaviour of the nodes is the same. The general directed model should recover the subsumed undirected one and, naturally, the same should hold for the embedding method.

However, these desiderata are not necessarily met. As stated earlier, given an invertible matrix $\mathbf{T}$, the embeddings $\mathbf{Y}^l = \mathbf{X}^l\mathbf{T}$ and $\mathbf{Y}^r = \mathbf{X}^r\mathbf{T}^{-\top}$ yield the same probability matrix $\mathbf{P} = \mathbf{X}^l(\mathbf{X}^r)^\top$. This impies that unless $\mathbf{T} = \mathbf{T}^{-\top}$ (meaning $\mathbf{T}$ is an orthonormal matrix corresponding to a rotation), we could apply *different* transformations to the inward and outward embeddings and still obtain the same RDPG.

Given these observations, consider a directed RDPG model where the embedding matrices $\mathbf{X}^l$ and $\mathbf{X}^r$ are constrained to be orthogonal and of the same column-wise norm. The following result asserts that this suffices to ensure an admissible $\mathbf{T}$ is

orthonormal, hence reducing the model's ambiguity to a global rotation–just like in the undirected case.

Proposition 3.5.1. *Let* $\mathbf{P} = \mathbf{X}^l(\mathbf{X}^r)^\top$ *be the probability matrix of a directed RDPG model, where* $\{\mathbf{X}^l, \mathbf{X}^r\}$ *are* $N \times d$ *matrices with rank* d *such that* $(\mathbf{X}^l)^\top\mathbf{X}^l = (\mathbf{X}^r)^\top\mathbf{X}^r = \mathbf{D}_X$ *is diagonal. Let* $\mathbf{T} \in \mathbb{R}^{d\times d}$ *be an invertible matrix such that* $\mathbf{Y}^l = \mathbf{X}^l\mathbf{T}$ *and* $\mathbf{Y}^r = \mathbf{X}^r\mathbf{T}^{-\top}$ *are also orthogonal with the same column-wise norms; i.e.,* $(\mathbf{Y}^l)^\top\mathbf{Y}^l = (\mathbf{Y}^r)^\top\mathbf{Y}^r = \mathbf{D}_Y$ *is diagonal. Then,* $\mathbf{T}$ *is an orthonormal matrix.*

Proof. Combining $\mathbf{Y}^r = \mathbf{X}^r\mathbf{T}^{-\top}$ with $(\mathbf{X}^r)^\top\mathbf{X}^r = \mathbf{D}_X$ and $(\mathbf{Y}^r)^\top\mathbf{Y}^r = \mathbf{D}_Y$ we find that $\mathbf{D}_X = \mathbf{T}\mathbf{D}_Y\mathbf{T}^\top$. Proceeding analogously with $\mathbf{Y}^l$ we further obtain that $\mathbf{D}_X = \mathbf{T}^{-\top}\mathbf{D}_Y\mathbf{T}^{-1}$. Multiplying both identities results in $\mathbf{D}_X^2 = \mathbf{T}\mathbf{D}_Y^2\mathbf{T}^{-1}$. Thus, the columns of $\mathbf{T}$ are linearly independent eigenvectors of a diagonal matrix. Furthermore, given $\mathbf{D}_X = \mathbf{T}\mathbf{D}_Y\mathbf{T}^\top$ it follows that the above eigendecomposition is necessarily one with orthonormal eigenvectors. □

The constraints in (3.17) do not limit the expressiveness of the model, since they are compatible with those ASE implicitly imposes. Next, we develop a feasible first-order method that enforces the orthogonality constraint at all iterations. After convergence it is straightforward to equalize the resulting column-wise norms so that they are the same for both $\hat{\mathbf{X}}^l$ and $\hat{\mathbf{X}}^r$, without affecting the generated $\mathbf{P}$; see Remark 3.5.3.

3.5.2 Optimizing on a manifold

We have concluded that for the sake of interpretability and quality of the representation, it is prudent to impose the matrices $\mathbf{X}^l$ and $\mathbf{X}^r$ have orthogonal columns. One classical way to tackle this is by adding these constraints to the optimization problem as in (3.17), and solve it via Lagrangian-based methods. For some constraints with geometric properties, a more suitable and timely approach is to pose the optimization problem on a smooth manifold. One can then resort to feasible methods that search exclusively over the manifold, i.e., the constraints are satisfied from the start and throughout the entire iterative minimization process (Absil et al., 2009; Boumal, 2023). This way, we can think of the optimization as being *unconstrained* because the manifold is all there is. In the sequel we explore this last idea.

Interestingly, the space of matrices having orthogonal columns does not form any known and well-studied manifold. Yet, we show the required geometric structure is present in our problem and thus we have to define several objects as well as compute various operators to facilitate optimization (Absil et al., 2009; Boumal, 2023). The conceptual roadmap is as follows. Recall that a smooth manifold $\mathcal{M}$ can be locally approximated by a linear space, the so-called *tangent space*. If we consider the

objective function $f : \mathcal{M} \to \mathbb{R}$ defined from the (Riemannian) manifold to $\mathbb{R}$, then the *Riemannian gradient* of the function is an element of the tangent space. This Riemaninann gradient, which will be denoted as $\operatorname{grad} f$, can be computed as the projection of the Euclidean gradient ∇f to the tangent space. Having computed the gradient, a classical descent method consists of taking a certain step in the opposite direction. However, this step typically leads to a point outside the manifold. Therefore, it is necessary to map the result back onto the manifold, which is achieved through a so-called *retraction* map.

Next, we formally define our manifold, derive its tangent space, characterize the projection to that space, and derive a useful retraction. The manifold that resembles ours the most is the so-called Stiefel manifold, which consist of matrices with orthogonal and unit-norm (i.e., orthonormal) columns

$$St(d, N) := \{\mathbf{X} \in \mathbb{R}^{N \times d} : \mathbf{X}^\top \mathbf{X} = \mathbf{I}_d\}. \tag{3.20}$$

The key difference is that in our setup we do not require unit-norm columns. Thus, let $\mathbb{R}^N_* = \mathbb{R}^N \setminus \{\mathbf{0}_N\}$ be the set of N dimensional vectors without the null vector, and let $\mathbb{R}^{N \times d}_*$ be the product of d copies of $\mathbb{R}^N_*$. This open set is the set of $N \times d$ matrices without null columns. We are interested in matrices with orthogonal columns, namely

$$\begin{aligned} \mathcal{M}(d, N) &:= \{\mathbf{X} \in \mathbb{R}^{N \times d}_* : \mathbf{X}^\top \mathbf{X} \text{ is diagonal}\} \\ &= \{\mathbf{X} \in \mathbb{R}^{N \times d}_* : \mathbf{M} \circ (\mathbf{X}^\top \mathbf{X}) = \mathbf{0}_{d \times d}\}, \end{aligned} \tag{3.21}$$

where once more $\mathbf{M} = \mathbf{1}_d \mathbf{1}_d^\top - \mathbf{I}_d$ is a particular mask matrix, with zeros in the diagonal and ones everywhere else.

The following proposition establishes that $\mathcal{M}(d, N)$ is actually a differential manifold (for notational convenience, we henceforth use $\mathcal{M}$ instead of $\mathcal{M}(d, N)$ since both d and N are fixed throughout). Moreover, $\mathcal{M}$ is a Riemannian manifold since $\mathcal{M} \subset \mathbb{R}^{N \times d}$ is a vector space equipped with the usual trace inner product.

Proposition 3.5.2. *The set $\mathcal{M}$ in (3.21) is a differential manifold and its dimension is $Nd - d(d-1)/2$. Furthermore, the tangent space at $\mathbf{X} \in \mathcal{M}$ is given by*

$$\mathrm{T}_{\mathbf{X}}\mathcal{M} = \{\boldsymbol{\zeta} \in \mathbb{R}^{N \times d} : \mathbf{M} \circ \left(\boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta}\right) = \mathbf{0}_{N \times d}\}.$$

Proof. In order to show that $\mathcal{M}$ is a manifold and to further understand its differential structure, consider the function $F : \mathbb{R}^{N \times d}_* \to \mathcal{S}_d$ defined as $F(\mathbf{X}) = \mathbf{M} \circ (\mathbf{X}^\top \mathbf{X} - \mathbf{I}_d)$. Observe that $\mathcal{M}$ is defined as the preimage of zero through F, so we will prove that this is a regular value.

For $\boldsymbol{\zeta} \in \mathbb{R}^{N \times d}$, the derivative of F in $\mathbf{X} \in F^{-1}(\mathbf{0}_{d \times d})$ along $\boldsymbol{\zeta}$ is $DF(\mathbf{X})\boldsymbol{\zeta} =$

$\mathbf{M} \circ (\boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta})$. Next we establish that $DF(\mathbf{X})$ is onto. Indeed, let $\boldsymbol{\eta}$ be a matrix in the orthogonal complement of the image, i.e., $\boldsymbol{\eta} \in ImDF(\mathbf{X})^\perp \subset \mathcal{S}_d$. Then

$$\langle \boldsymbol{\eta}, \mathbf{M} \circ (\boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta}) \rangle = 0, \, \forall \boldsymbol{\zeta} \in \mathbb{R}^{N \times d}.$$

Now, since the diagonal of $\boldsymbol{\eta}$ is null, we may drop the Hadamard product with $\mathbf{M}$ and obtain

$$\langle \boldsymbol{\eta}, \mathbf{M} \circ (\boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta}) \rangle = \langle \boldsymbol{\eta}, \boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta} \rangle = 0, \, \forall \boldsymbol{\zeta} \in \mathbb{R}^{N \times d}.$$

So we have $\mathrm{tr}\left(\boldsymbol{\eta} \boldsymbol{\zeta}^\top \mathbf{X}\right) + \mathrm{tr}\left(\boldsymbol{\eta} \mathbf{X}^\top \boldsymbol{\zeta}\right) = 0$ and these two summands are equal to each other by virtue of the circular property of the trace operator. Hence, we obtain $2\mathrm{tr}\left(\boldsymbol{\eta} \mathbf{X}^\top \boldsymbol{\zeta}\right) = 0$, $\forall \boldsymbol{\zeta} \in \mathbb{R}^{N \times d}$, and since this trace vanishes for all $\boldsymbol{\zeta}$, we have that $\boldsymbol{\eta} \mathbf{X}^\top = \mathbf{0}_{d \times N}$. Multiplying by $\mathbf{X}$ we obtain $\boldsymbol{\eta} \mathbf{X}^\top \mathbf{X} = \mathbf{0}_{d \times d}$. Becasue $\mathbf{X}^\top \mathbf{X}$ is diagonal, necessarily $\boldsymbol{\eta} = \mathbf{0}_{d \times d}$ and therefore $DF(\mathbf{X})$ is onto. The conclusion is that $\mathcal{M}$ is a differential manifold, of dimension $Nd - \frac{d(d-1)}{2}$.

The tangent space at $\mathbf{X}$ can be obtained as the kernel of $DF(\mathbf{X})$, so we have

$$T_{\mathbf{X}} \mathcal{M} = \{\boldsymbol{\zeta} \in \mathbb{R}^{N \times d} : \mathbf{M} \circ \left(\boldsymbol{\zeta}^\top \mathbf{X} + \mathbf{X}^\top \boldsymbol{\zeta}\right) = \mathbf{0}_{d \times d}\}, \tag{3.22}$$

completing the proof. □

To perform a manifold GD step, one needs to compute the Riemmanian gradient of the function defined in $\mathcal{M}$. We obtain $\mathrm{grad}\, f$ as the projection of the Euclidean gradient [cf. (3.18)-(3.19)] onto the tangent space. A natural way to compute said projection is to first characterize and compute the projection to the normal space. Given $\mathbf{X} \in \mathcal{M}$, the normal space at $\mathbf{X}$ is

$$T_{\mathbf{X}} \mathcal{M}^\perp = \{\mathbf{N} \in \mathbb{R}^{N \times d} : \langle \mathbf{N}, \boldsymbol{\zeta} \rangle = \mathrm{tr}(\mathbf{N}^\top \boldsymbol{\zeta}) = 0, \forall \boldsymbol{\zeta} \in T_{\mathbf{X}} \mathcal{M}\}.$$

A useful alternative characterization is given next.

Lemma 3.5.3. *The normal space at* $\mathbf{X}$ *is*

$$T_{\mathbf{X}} \mathcal{M}^\perp = \{\mathbf{X} \boldsymbol{\Lambda} \in \mathbb{R}^{N \times d} : \boldsymbol{\Lambda} \in \mathcal{S}_d\},$$

where $\mathcal{S}_d = \{\mathbf{X} \in \mathbb{R}^{d \times d} : \mathbf{X} = \mathbf{X}^\top, \mathrm{diag}(\mathbf{X}) = \mathbf{0}_{d \times d}\}$ *is the set of* $d \times d$ *symmetric matrices with null diagonal.*

Proof. Consider a matrix of the form $\mathbf{X} \boldsymbol{\Lambda}$ with $\boldsymbol{\Lambda} \in \mathcal{S}_d$, and let us show that it is orthogonal to a matrix of the tangent space. Now, observe that $\mathrm{tr}\left((\mathbf{X} \boldsymbol{\Lambda})^\top \boldsymbol{\zeta}\right) =$

$\operatorname{tr}\left(\boldsymbol{\Lambda}\boldsymbol{\zeta}^\top\mathbf{X}\right)$. Therefore,

$$\operatorname{tr}\left((\mathbf{X}\boldsymbol{\Lambda})^\top\boldsymbol{\zeta}\right) = \frac{1}{2}\operatorname{tr}\left(\boldsymbol{\Lambda}(\mathbf{X}^\top\boldsymbol{\zeta} + \boldsymbol{\zeta}^\top\mathbf{X})\right) = 0.$$

The last trace is zero since $\boldsymbol{\Lambda} \in \mathcal{S}_d$ and $\mathbf{X}^\top\boldsymbol{\zeta} + \boldsymbol{\zeta}^\top\mathbf{X}$ is diagonal, because $\boldsymbol{\zeta} \in T_{\mathbf{X}}\mathcal{M}$. Note that, as expected, the dimension of the normal space is $\frac{d(d-1)}{2}$, which is the dimension of $\mathcal{S}_d$. □

Computing the projection to the normal space requires some work due to the null diagonal constraint in $\mathcal{S}_d$, which is not present in the normal space to $St(d, N)$ (Boumal, 2023, p. 161). The result is given in the next lemma.

Lemma 3.5.4. *Let $\mathbf{X} \in \mathcal{M}$ and let $\pi^{\perp}_{\mathbf{X}} : \mathbb{R}^{N\times d} \to T_{\mathbf{X}}\mathcal{M}^{\perp}$ be the projection to the normal space. Then*

$$\pi^{\perp}_{\mathbf{X}}(\mathbf{Z}) = \mathbf{X}s(2\mathbf{D}\mathbf{L}), \tag{3.23}$$

where $s : \mathbb{R}^{d\times d} \to \mathcal{S}_d$ is the symmetrizing function $s(\mathbf{Z}) = \frac{\mathbf{Z}+\mathbf{Z}^\top}{2} - \operatorname{diag}(\mathbf{Z})$, $\mathbf{D} = \left(\mathbf{X}^\top\mathbf{X}\right)^{1/2}$ and $\mathbf{L} = \left(\mathbf{D}^{-1}\mathbf{X}^\top\mathbf{Z}\right)\circ\mathbf{F}$, where $\mathbf{E} = \mathbf{1}_d\mathbf{1}_d^\top\mathbf{D}^2 + \mathbf{D}^2\mathbf{1}_d\mathbf{1}_d^\top$ and $\mathbf{F}$ has entries $F_{ij} = E_{ij}^{-1}$.

Remark 3.5.2. Note that (3.23) is of the form $\mathbf{X}\boldsymbol{\Lambda}$, with $\boldsymbol{\Lambda} \in \mathcal{S}_d$. It thus belongs to the normal space by virtue of the characterization in Lemma 3.5.3. The calculations to show it is indeed the projection boil down to proving that $\mathbf{Z} - \pi^{\perp}_{\mathbf{X}}(\mathbf{Z})$ lives in the tangent space. Specifically, to establish (3.23) we take $\mathbf{X}\boldsymbol{\Lambda}$ and derive conditions that $\boldsymbol{\Lambda}$ has to verify when imposing that $\mathbf{Z} - \mathbf{X}\boldsymbol{\Lambda} \in \mathrm{T}_{\mathbf{X}}\mathcal{M}$.

Proof. To compute the projection to the normal space, recall note that matrices $\mathbf{D}$ and $\mathbf{E}$ above defined allow us to re-write the operation $\varphi(\mathbf{A}) = \mathbf{A}\mathbf{D}^2 + \mathbf{D}^2\mathbf{A}$ as $\varphi(\mathbf{A}) = \mathbf{A}\circ\mathbf{E}$. In particular, this allows us to obtain an expression for the inverse operation, which will be needed. Indeed, if $\mathbf{F}$ is the matrix with entries $F_{ij} = E_{ij}^{-1}$, then $(\mathbf{A}\mathbf{D}^2 + \mathbf{D}^2\mathbf{A})\circ\mathbf{F} = \mathbf{A}$, for all $\mathbf{A} \in \mathbb{R}^{d\times d}$. We can now prove the expression of the projection as follows.

From the characterization of Lemma 3.5.3, it is clear that $\mathbf{X}s(2\mathbf{D}\mathbf{L}) \in T_{\mathbf{X}}\mathcal{M}^{\perp}$, since $s(2\mathbf{D}\mathbf{L}) \in \mathcal{S}_d$. Let us see that $\mathbf{Z} - \mathbf{X}s(2\mathbf{D}\mathbf{L}) \in T_{\mathbf{X}}\mathcal{M}$. From (3.22), we have to show that

$$\left(\mathbf{Z} - \mathbf{X}s(2\mathbf{D}\mathbf{L})\right)^\top\mathbf{X} + \mathbf{X}^\top\left(\mathbf{Z} - \mathbf{X}s(2\mathbf{D}\mathbf{L})\right) \text{ is diagonal.}$$

Indeed,

$$\left(\mathbf{Z} - \mathbf{X}s(2\mathbf{D}\mathbf{L})\right)^\top\mathbf{X} + \mathbf{X}^\top\left(\mathbf{Z} - \mathbf{X}s(2\mathbf{D}\mathbf{L})\right) =$$

$$
\begin{aligned}
&= \mathbf{Z}^\top\mathbf{X} - (s(2\mathbf{DL}))^\top\mathbf{X}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \mathbf{X}^\top\mathbf{X}s(2\mathbf{DL}) = \\
&= \mathbf{Z}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \frac{1}{2}\left[2\mathbf{DL} + 2(\mathbf{DL})^\top\right]\mathbf{X}^\top\mathbf{X} + \operatorname{diag}(2\mathbf{DL})\mathbf{X}^\top\mathbf{X} + \\
&\quad + \mathbf{X}^\top\mathbf{X}\operatorname{diag}(2\mathbf{DL}) - \frac{1}{2}\mathbf{X}^\top\mathbf{X}\left[2\mathbf{DL} + 2(\mathbf{DL})^\top\right].
\end{aligned}
$$

Now, since $\operatorname{diag}(2\mathbf{DL})$ and $\mathbf{X}^\top\mathbf{X}$ are diagonal matrices, they commute, and their product is diagonal. So we can forget those two terms in the expression, and continue with the rest. We will use the expression of $\mathbf{L}$ and the fact that $\mathbf{X}^\top\mathbf{X} = \mathbf{D}^2$. Hence,

$$
\begin{aligned}
&\mathbf{Z}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \frac{1}{2}\left[2\mathbf{DL} + 2(\mathbf{DL})^\top\right]\mathbf{X}^\top\mathbf{X} - \frac{1}{2}\mathbf{X}^\top\mathbf{X}\left[2\mathbf{DL} + 2(\mathbf{DL})^\top\right] = \\
&= \mathbf{Z}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \left[\mathbf{D}\left(\mathbf{L}\mathbf{D}^2 + \mathbf{D}^2\mathbf{L}\right) + \left(\mathbf{L}^\top\mathbf{D}^2 + \mathbf{D}^2\mathbf{L}^\top\right)\mathbf{D}\right] = \\
&= \mathbf{Z}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \left[\mathbf{D}\left(\mathbf{L}\circ\mathbf{E}\right) + \left(\mathbf{L}^\top\circ\mathbf{E}\right)\mathbf{D}\right].
\end{aligned}
$$

Now, observe that $\mathbf{L}\circ\mathbf{E} = \left(\left(\mathbf{D}^{-1}\mathbf{X}^\top\mathbf{Z}\right)\circ\mathbf{F}\right)\circ\mathbf{E} = \mathbf{D}^{-1}\mathbf{X}^\top\mathbf{Z}$, and the same happens with $\mathbf{L}^\top$. We end up with

$$
\mathbf{Z}^\top\mathbf{X} + \mathbf{X}^\top\mathbf{Z} - \left[\mathbf{D}\left(\mathbf{D}^{-1}\mathbf{X}^\top\mathbf{Z}\right) + \left(\mathbf{Z}^\top\mathbf{X}\mathbf{D}^{-1}\right)\mathbf{D}\right] = \mathbf{0}_{d\times d},
$$

which in particular is diagonal. Therefore, $\mathbf{Z} - \mathbf{X}s(2\mathbf{DL}) \in T_{\mathbf{X}}\mathcal{M}$, which completes the proof. □

Once we have characterized the projection onto the normal space, the projection onto the tangent space is simply given by $\pi_{\mathbf{X}}(\mathbf{Z}) = \mathbf{Z} - \pi^{\perp}_{\mathbf{X}}(\mathbf{Z})$. For ease of reference, we state this fact as the following proposition.

Proposition 3.5.5. *Let $\mathbf{X} \in \mathcal{M}$. The projection to the tangent space $\pi_{\mathbf{X}} : \mathbb{R}^{N\times d} \to T_{\mathbf{X}}\mathcal{M}$ can be computed as:*

$$
\pi_{\mathbf{X}}(\mathbf{Z}) = \mathbf{Z} - \pi^{\perp}_{\mathbf{X}}(\mathbf{Z}) = \mathbf{Z} - \mathbf{X}s(2\mathbf{DL}).
$$

When we take a small step in the opposite direction of $\operatorname{grad} f$, we generally end up outside $\mathcal{M}$. We therefore need a way to map points in the tangent bundle of the manifold back to the manifold itself. Informally, this is the role of a retraction.

Given a full rank matrix $\mathbf{Z} \in \mathbb{R}^{N\times d}$, consider its decomposition $\mathbf{Z} = \widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}$, where $\widetilde{\mathbf{Q}}$ is a matrix with orthogonal columns and $\widetilde{\mathbf{R}}$ is upper triangular with ones in the diagonal. This decomposition is unique. Indeed, one may obtain $\widetilde{\mathbf{Q}}$ by a Gram-Schmidt process, but skipping the normalization steps. A more efficient approach is to consider the classical QR decomposition ($\mathbf{Z} = \mathbf{QR}$, with $\mathbf{Q}$ orthonormal and $\mathbf{R}$ upper triangular), and compute $\widetilde{\mathbf{Q}} = \mathbf{Q}\mathbf{D}_R$, where $\mathbf{D}_R = \operatorname{diag}(\mathbf{R})$ is the diagonal matrix with the diagonal entries of $\mathbf{R}$. In a way, this modification of the QR decomposition shifts the "normalization" of the columns from the upper triangular factor

towards the orthogonal factor.

Note that $\widetilde{\mathbf{Q}} \in \mathcal{M}$ and this decomposition will serve to define a retraction to the manifold in the next proposition. Again, this procedure differs from the popular Q-factor retraction to the Stiefel manifold (Boumal, 2023, p. 160).

Proposition 3.5.6. *Let $\mathbf{X} \in \mathcal{M}$ and $\boldsymbol{\zeta} \in T_{\mathbf{X}}\mathcal{M}$ a tangent vector. Then, the mapping*

$$R_{\mathbf{X}}(\boldsymbol{\zeta}) = \widetilde{qf}(\mathbf{X} + \boldsymbol{\zeta})$$

is a retraction, where $\widetilde{qf}(\mathbf{A})$ denotes the $\widetilde{\mathbf{Q}}$ factor of the modified QR decomposition described above, and the sum $\mathbf{X} + \boldsymbol{\zeta}$ stands for the usual abuse of notation for embedded manifolds on vector spaces.

Proof. Denoting by $\mathbb{R}^{N\times d}_{fr}$ the set of $N \times d$ full-rank matrices, and by $Supp_1(d)$ the set of upper triangular matrices with ones in the diagonal, let us consider the mapping

$$\phi : \mathcal{M} \times Supp_1(d) \to \mathbb{R}^{N\times d}_{fr}, \text{ with } \phi(\widetilde{\mathbf{Q}}, \widetilde{\mathbf{R}}) = \widetilde{\mathbf{Q}}\widetilde{\mathbf{R}}.$$

From the discussion above we have that ϕ is bijective. Furthermore, ϕ is smooth since its the restriction of the matrix multiplication to a submanifold. Now, given a full rank matrix $\mathbf{M}$, the first component of ϕ^{-1} can be obtained as the result of a modified Gram-Schmidt process, which is is differentiable. The second component can then be obtained as $\widetilde{\mathbf{R}} = \widetilde{\mathbf{Q}}^{-1}\mathbf{M}$, and therefore it is also differentiable. It follows that ϕ is a diffeomorphism.

We also have that $\phi(\widetilde{\mathbf{Q}}, \mathbf{I}_d) = \widetilde{\mathbf{Q}}$. Following Absil et al. (Proposition 4.1.2 from 2009) we have that the projection onto the first component of ϕ^{-1} is a retraction, which is exactly the $\widetilde{qf}$ mapping.

□

We now have all the ingredients for the GD method to minimize $f(\mathbf{X}^l, \mathbf{X}^r) = \|\mathbf{M} \circ (\mathbf{A} - \mathbf{X}^l(\mathbf{X}^r)^\top)\|_F^2$ over $\mathcal{M}$, which is tabulated under Algorithm 3. The convergence rate of Riemannian GD is the same as the unconstrained counterpart (i.e., producing points with grad f smaller than ε in $\mathcal{O}(1/\varepsilon^2)$ iterations) (Boumal et al., 2018). The computational complexity of each iteration is dominated by the QR decomposition in the retraction.

We extended the `Pymanopt` package (Townsend et al., 2016) with a class for the manifold $\mathcal{M}$ defined in Proposition 3.5.2, which forms part of the code available at https://github.com/marfiori/efficient-ASE.

Remark 3.5.3 (Rescaling the factors' columns). Algorithm 3 does not quite solve (3.17). While both $\{\mathbf{X}^l_k, \mathbf{X}^r_k\}$ belong to $\mathcal{M}$, the constraint $(\mathbf{X}^l_k)^\top \mathbf{X}^l_k =$

Algorithm 3 Riemannian Gradient Descent (GD) on $\mathcal{M}$

Require: Initial $\mathbf{X}_0^l$ and $\mathbf{X}_0^r$

repeat
  Compute Euclidean gradients
  $\nabla f_{\mathbf{X}^l}(\mathbf{X}_k^l, \mathbf{X}_k^r)$ and $\nabla f_{\mathbf{X}^r}(\mathbf{X}_k^l, \mathbf{X}_k^r)$
  Compute Riemannian gradients as
  $\operatorname{grad} f_{\mathbf{X}^l}(\mathbf{X}_k^l, \mathbf{X}_k^r) = \pi_{\mathbf{X}_k^l}(\nabla f_{\mathbf{X}^l}(\mathbf{X}_k^l, \mathbf{X}_k^r))$
  $\operatorname{grad} f_{\mathbf{X}^r}(\mathbf{X}_k^l, \mathbf{X}_k^r) = \pi_{\mathbf{X}_k^r}(\nabla f_{\mathbf{X}^r}(\mathbf{X}_k^l, \mathbf{X}_k^r))$
  Compute retraction with α chosen via the Armijo rule
  $\mathbf{X}_{k+1}^l = R_{\mathbf{X}_k^l}\left(-\alpha \operatorname{grad} f_{\mathbf{X}^l}(\mathbf{X}_k^l, \mathbf{X}_k^r)\right)$,
  $\mathbf{X}_{k+1}^r = R_{\mathbf{X}_k^r}\left(-\alpha \operatorname{grad} f_{\mathbf{X}^r}(\mathbf{X}_k^l, \mathbf{X}_k^r)\right)$
until convergence
return $\{\mathbf{X}_k^l, \mathbf{X}_k^r\}$.

$(\mathbf{X}_k^r)^\top \mathbf{X}_k^r$ will in general not be satisfied upon convergence. Dropping the iteration index for simplicity, let $\bar{\mathbf{x}}_i^l, \bar{\mathbf{x}}_i^r \in \mathbb{R}^N$ be the i-th columns of $\mathbf{X}^l$ and $\mathbf{X}^r$, respectively. To obtain a feasible solution from the output of Algorithm 3, for each dimension $i = 1, \ldots, d$ we define scaling factors $s_i = \|\bar{\mathbf{x}}_i^l\|_2 / \|\bar{\mathbf{x}}_i^r\|_2$ and collect them in the diagonal matrix $\mathbf{S} = \operatorname{diag}(s_1, \ldots, s_d)$. We then rescale the columns of the embedding matrices via $\mathbf{X}_k^l \leftarrow \mathbf{X}_k^l \mathbf{S}^{-1/2}$ and $\mathbf{X}_k^r \leftarrow \mathbf{X}_k^r \mathbf{S}^{1/2}$, without affecting the value of the objective function but now satisfying the constraint in (3.17).

Example 3.5.3 (Bipartisan senate revisited). Going back to the bipartisan senate from Example 3.5.1, Fig. 3.3 depicts the solution of (3.17) for the same simulated bipartite senator-law digraph (imposing the orthogonality constraints and rescaling in Remark 3.5.3). Unlike in Example 3.5.1, the Riemannian GD algorithm on the manifold $\mathcal{M}$ is able to recover the same structure as the ASE. Laws are now correctly aligned with their corresponding party, thus faithfully revealing the structure in the data.

3.6 Numerical experiments and applications

In this section we illustrate our embedding algorithms' ability to produce accurate and informative estimates of nodal latent position vectors. We explore a variety of GRL applications and consider synthetic and real network data. Our test cases are designed to target ASE challenges outlined in Section 3.1, namely: i) missing data; ii) embedding multiple networks; and iii) graph streams (with fixed and varying number of nodes). For each case we assess the results with respect to estimation accuracy, interpretability, and stability/alignment in dynamic environments. The suitability of

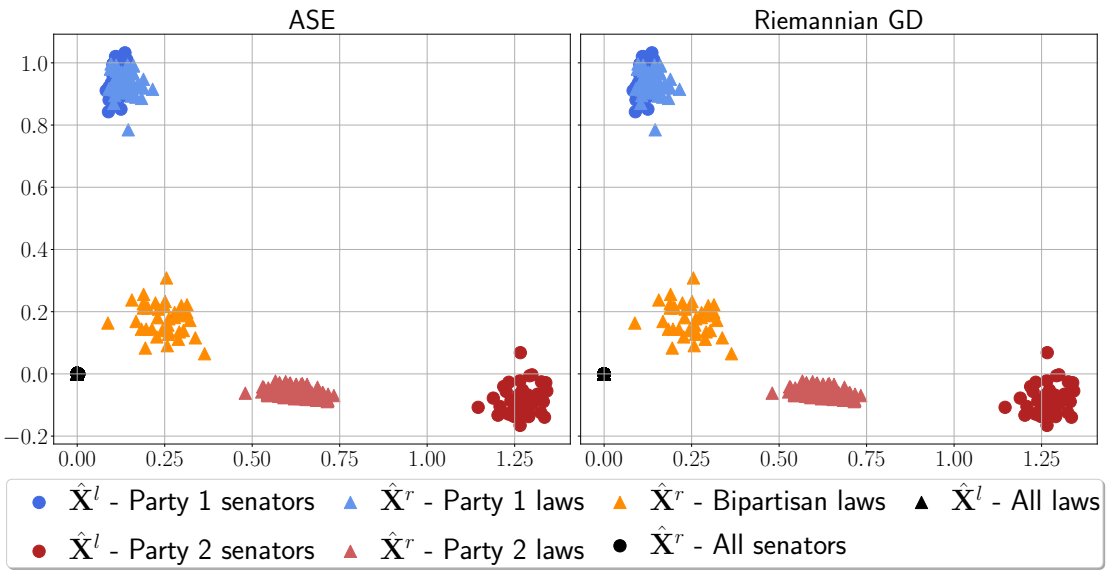

Figure 3.3: Solution to the embedding problem (3.17) for the bipartisan senate example. ASE (left) and Riemannian GD (right). Notice how both solutions are nearly identical [cf. unconstrained GD in Fig. 3.2 (right)], underscoring the importance of the orthogonality constraints.

spectral embeddings (rooted in the RDPG model) for downstream tasks has already been well-documented (Athreya et al., 2017; Levin et al., 2018). For this reason, the goal here is to demonstrate the effectiveness of our algorithms in generating node embeddings that faithfully represent network structure in novel settings i)-iii). Additional results exploring algorithm sensitivity to random initialization are in Section 3.6.1. The code for all these experiments is publicly available at our GitHub repository.

3.6.1 Robustness to initialization

Since the objective functions we optimize are non-convex and convergence guarantees provided for general mask matrices are to stationary solutions, it is prudent to study the algorithms' sensitivity to the initialization. As discussed in Section 3.4.1, except for specific problematic initializations corresponding to sub-optimal stationary points, in our experience all algorithms converge to the optimum when they are initialized at random. The following experiment illustrates this desirable property, in particular for the Riemannian GD (i.e., Algorithm 3) method developed to embed digraphs. Similar results are obtained for the other algorithms, but not included here to avoid repetition.

We consider a Lancichinetti–Fortunato–Radicchi (LFR) (Lancichinetti et al., 2008) benchmark graph with $N = 1000$ nodes, randomly initialize Algorithm 3 and plot the evolution of the cost function $f(\mathbf{X}_k^l, \mathbf{X}_k^r)$ in (3.17). The LFR model

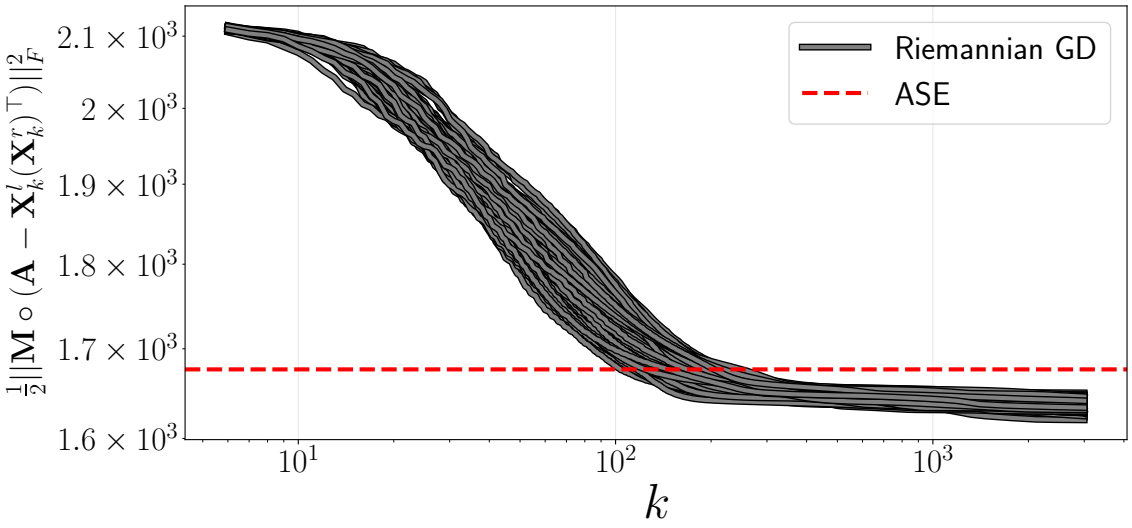

Figure 3.4: The evolution of $f(\mathbf{X}_k^l, \mathbf{X}_k^r) = \frac{1}{2}\left\|\mathbf{M} \circ (\mathbf{A} - \mathbf{X}_k^l(\mathbf{X}_k^r)^\top)\right\|_F^2$ using Algorithm 3 to embed an LFR graph, starting from 75 different random initializations. The first 5 iterations are omitted for clarity. Note how the algorithm systematically produces estimates of the embeddings with a lower cost than ASE, and marginal variability regardless of the initialization.

is a widely adopted benchmark that produces graphs with properties observed in real-world networks, particularly in terms of the resulting degree distribution and community sizes. Here, the LFR graph was generated using parameters $\tau_1 = 3$ and $\tau_2 = 2$ as exponents of the power law distributions for the degree and community size, respectively, and mixing parameter $\mu = 0.1$. The resulting graph has 16 communities, the larger one with 142 nodes, and the smaller one with 30 nodes. The largest hub has 157 neighbors, and there are several nodes with degree 2. Fig. 3.4 shows the results over 75 randomly initialized runs where $d = 16$, and also the cost function value 1676.49 obtained by ASE. Regardless of the initialization, the limiting objective values obtained via Riemannian GD exhibit marginal variability (mean = 1635.66, std = 6.52) and always outperform ASE. In terms of timing, embedding each of these LFR graphs with $N = 1000$ nodes takes 40 seconds on average.

3.6.2 Inference with missing data

Next we illustrate how GD-based inference can be useful for GRL with missing data. The setup is similar to that of Example 3.5.1, but here we rely on real United Nations (UN) General Assembly voting data (Voeten et al., 2009). For each roll call and country, the dataset includes if the country was present and if so the corresponding vote (either 'Yes', 'No', or 'Abstain') for each proposal. We analyze the associated bipartite digraph pertaining to a particular year, where nodes

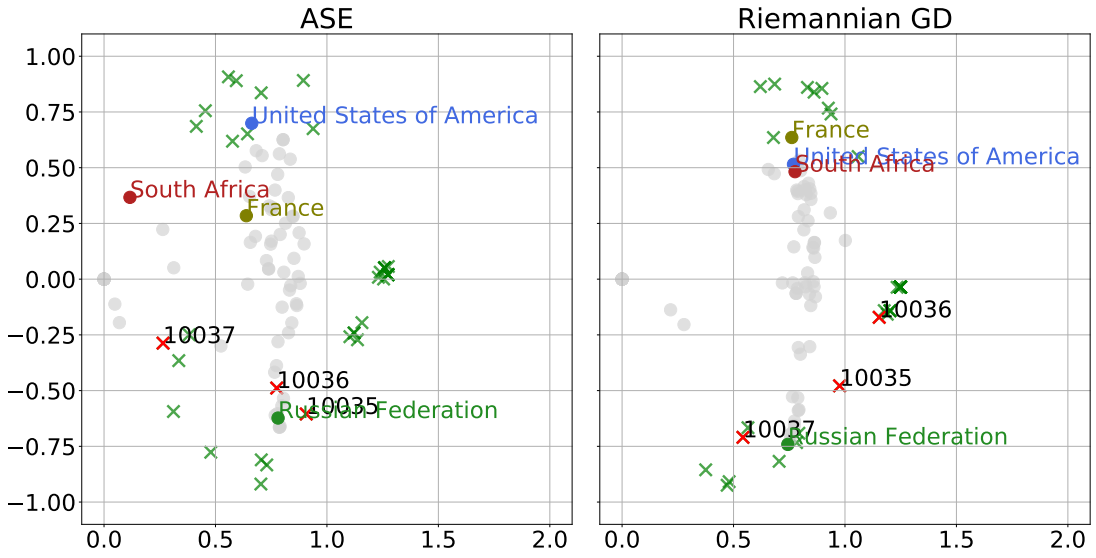

Figure 3.5: UN General Assembly voting data for 1955. ASE (left) and Riemannian GD (i.e., Algorithm 3) with mask matrix encoding present and absent (or abstained) voters (right). Our approach is able to assign the absent voters to the correct group (e.g., South Africa) and offers a more clear clustering of roll calls.

correspond to countries and roll calls, and an edge from a country to a roll call exists if it voted affirmatively. If the country was absent or abstained, we will tag that edge as unknown ($M_{ij} = 0$).

Fig. 3.5 depicts the node embeddings ($d = 2$) of the graph from 1955, estimated by ASE (naively assuming unknown edges do not exist, $A_{ij} = 0$) and Riemannian GD (i.e. Algorithm 3). Consider the countries, which are displayed as circles. We highlight four interesting cases: Russia, USA, France, and South Africa. At the time, the first two represented two poles of the world, and are naturally almost orthogonal to each other for both methods. Note furthermore how the ASE seems to indicate that South Africa is less likely to vote in agreement with Russia than (even) the USA, whereas the opposite is true for France.

The problem comes from equating an absence or abstention to a negative vote. For instance, South Africa was only present in roughly one third of the roll calls, and it voted almost identically to the USA. The Riemannian GD method, which acknowledges unknown edges via the mask $\mathbf{M}$, provides an embedding that reflects this agreement. Something similar happens with France, which differed from the USA only in six roll calls. Four correspond to USA abstentions and France voting 'Yes', another one where the opposite happened (and thus both cases should not be accounted for in the optimization problem), and finally the roll call 10036 where France was one of only two countries to vote 'No' (the USA voted 'Yes').

Regarding the embeddings of roll calls marked with a cross in Fig. 3.5, note how 10036 is aligned with the Russian block of countries by ASE, but it is better

placed as an intermediate proposal in Fig. 3.5 (right)–equally likely to be voted by all countries. Something similar occurs with roll call 10035, which dealt with the same subject of 10036, but met resistance from more countries (roughly a dozen, including the USA and France). In both cases several countries were not present or abstained during the voting. Incorrectly assuming these votes as negative by ASE leads to biased results. Much more can be said about the roll calls and their associated UN resolutions, but let us conclude the discussion by noting that roll call embeddings generated by Algorithm 3 form three clusters reflecting the geopolitical landscape at the time. There is a cluster for each pole (American and Russian), plus an intermediate one where both poles tend to vote similarly. On the other hand, ASE generates roll call embeddings that are incorrectly aligned (e.g., 10036), and a loose grouping of intermediate roll calls with shared voting from both poles.

3.6.3 Embedding multiple graphs: the batch case

Suppose now that we observe $m > 1$ graphs $\{\mathbf{A}_t\}_{t=1}^m$ and we are interested, for instance, in testing whether they are drawn from the same RDPG model, or, in tracking the embeddings over time. Assume that we can identify nodes across different observations; e.g., they correspond to labeled users in a social network and so a matching algorithm is not needed. Independently obtaining the ASE for each graph is undesirable because it yields arbitrarily rotated embeddings, a challenge that has motivated several recent research efforts.

Indeed, a hypothesis test which involves solving a Procrustes problem to align the embeddings was put forth in M. Tang et al. (2017). An alignment alternative is to jointly embed all m graphs via a single 'super-matrix' decomposition. The so-called *Omnibus* embedding first forms an $mN \times mN$ matrix derived from all $\{\mathbf{A}_t\}_{t=1}^m$, and then computes its ASE which enjoys asymptotic normality (Levin et al., 2017). The Unfolded ASE (UASE) also constructs an auxiliary matrix, but by horizontally stacking all $\{\mathbf{A}_t\}_{t=1}^m$ (Gallagher et al., 2021; Jones & Rubin-Delanchy, 2020). Nodal representations are then extracted from the SVD of this $N \times mN$ matrix. Under some technical assumptions, the UASE provably offers desirable longitudinal and cross-sectional stability (defined below; see also Gallagher et al. (2021)). However, the complexity and memory footprint of these *batch* approaches grow linearly with m, and they are only applicable to undirected graphs.

In the context of the algorithms proposed in this paper, we may leverage their iterative nature and initialize them using the estimated embeddings of another related (e.g., contiguous in time) graph. Unless radical changes take place from one graph to the other, this so-called warm restart is expected to produce embeddings that are closely aligned, with the added benefit of converging in few iterations.

Example 3.6.1 (Stability of GD estimates). Let us illustrate this (desirable) behaviour through a numerical example. We borrow the setting and code from Gallagher et al. (2021). Consider two graph samples drawn from a dynamic SBM with inter-community probability matrices

$$\mathbf{\Pi}_1 = \begin{pmatrix} 0.08 & 0.02 & 0.18 & 0.10 \\ 0.02 & 0.20 & 0.04 & 0.10 \\ 0.18 & 0.04 & 0.02 & 0.02 \\ 0.10 & 0.10 & 0.02 & 0.06 \end{pmatrix}, \mathbf{\Pi}_2 = \begin{pmatrix} 0.16 & 0.16 & 0.04 & 0.10 \\ 0.16 & 0.16 & 0.04 & 0.10 \\ 0.04 & 0.04 & 0.09 & 0.02 \\ 0.10 & 0.10 & 0.02 & 0.06 \end{pmatrix}$$

Initially there are four communities. At time 2, the first two communities merge, community 3 moves, and community 4 has its connection probabilities unchanged.

Ideally, when embedding both graphs: i) the representations of nodes in community 4 should not change (longitudinal stability); and ii) the time 2 embeddings of members of communities 1 and 2 should be similar, up to noise (cross-sectional stability). Fig. 3.6 displays the results for UASE (Gallagher et al., 2021), Omnibus embedding (Levin et al., 2017), independent ASE for each graph, and BCD (i.e., Algorithm 2 warm-restarted at time 2 with the result of time 1). As expected, independent ASE lacks longitudinal stability, and the Omnibus embedding fails to exhibit cross-sectional stability. Note how the time 2 Omnibus estimates of communities 1 and 2 remain separate, due to time 1 'interference' affecting this joint embedding.

UASE and BCD produce embeddings that fulfill both stability requirements i) and ii). However, BCD yields a better overall representation for both graphs. This is quantified via the cost function (3.1) evaluated at each solution; see above each plot for the numerical values. Unlike the batch UASE, gradient descent methods as the ones we present here offer a pathway towards tracking nodal representations in a streaming graph setting –the subject dealt with next.

3.6.4 Model tracking for graph streams

Consider now a monitoring scenario, where we observe a stream of time-indexed graphs $\{\mathbf{A}_t\}$ and the goal is to track the underlying model. Different from the batch setting of the previous section, we are now unable to jointly process the entire graph sequence. This may be due to memory constraints or stringent delay requirements. We will still assume that nodes are identifiable across time, but the algorithm's computational cost and memory footprint may not increase with t.

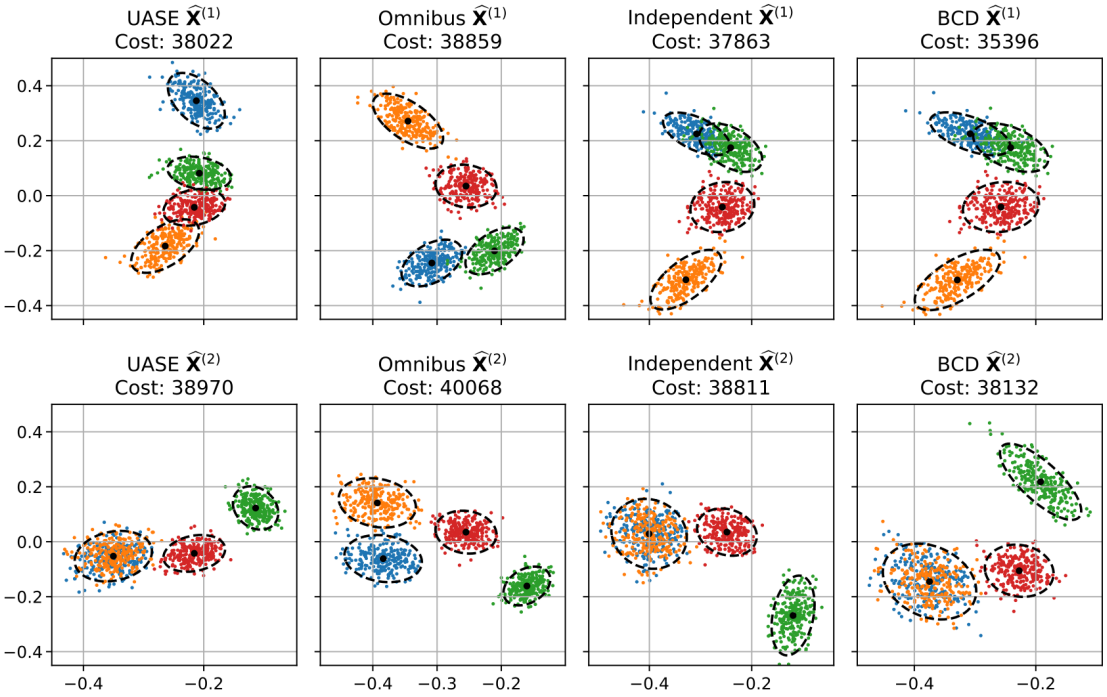

Figure 3.6: Embeddings of two SBM graph realizations, where communities 1 and 2 merge, while community 4 keeps the connection probabilities with other groups. Observe how the BCD approach (far right) manages to capture this behaviour, while providing the best representation for each graph individually (quantified by the smallest cost function values). Example adapted from Gallagher et al. (2021).

3.6.4.1 Fixed vertex set

We first consider the setting where N is fixed and we would like to track the latent vectors $\mathbf{X}_t \in \mathbb{R}^{N\times d}$.[1] Previous efforts in this direction have been mainly motivated by the change-point detection problem; i.e., detecting if and when the generative model of the observed graph sequence changes (Chen, 2019; Marenco et al., 2022; Yu et al., 2021; M. Zhang et al., 2020). Our focus is on the related problem of estimating the embeddings' evolution. A couple noteworthy applications include recommender systems (where rankings are revealed, or even change, over time) (Campos et al., 2014) or, as we discuss below, monitoring wireless networks (Mateos & Rajawat, 2013).

Independent ASE computation for each $\mathbf{A}_t$ suffers from the alignment issue already discussed. Instead, and supposing for now that the mask $\mathbf{M}$ can be ignored, it may well be the case that recursive methods to update the SVD of a perturbed matrix $\mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{\Delta}_t$ suffice (Brand, 2006). However, as we show in the following synthetic example, these approaches may also produce arbitrarily rotated estimates from one time-step to the next, and suffer from catastrophic error accumulation (Z.

[1] We stick to undirected graphs for ease of exposition, but extensions to digraphs are straightforward and presented in the numerical experiments.

Zhang et al., 2018).

Tracking of a dynamic SBM. Our idea is instead to proceed as in Remark 3.4.1, and warm-restart the GD iterations with the previous time-step's estimate $\hat{\mathbf{X}}_{t-1}$ (analogously to the example in Fig. 3.6). Consider a dynamic SBM graph with $N = 200$ nodes and two communities. At each time-step $t = 0, 1, 2, \dots$ a single randomly chosen node changes its community affiliation. We compare the tracking performance of warm-restarted GD [i.e., several iterations of GD in (3.2) initialized with the previous time-step's estimate] and the fast, recursive SVD algorithm in Brand (2006). The nodal embeddings for $t = 0$ and 1 (i.e., a single node changed affiliation) are depicted in Fig. 3.7 (top). Notice how online GD produces stable results, with a single vector moving from one cluster to the other. The rest of the nodes' embeddings remain virtually unchanged. On the other hand, the recursive SVD in Brand (2006) fails to preserve a common angular reference for $\hat{\mathbf{X}}_0$ and $\hat{\mathbf{X}}_1$. Another well-documented drawback of these incremental SVD methods is that, since they update only the d most significant components, the error $\|\hat{\mathbf{X}}_t\hat{\mathbf{X}}_t^\top - \mathbf{P}_t\|_F$ increases with t (Z. Zhang et al., 2018). Fig. 3.7 (bottom) illustrates this error-accumulation behavior, to be contrasted with online GD that keeps the error in check for all $t \geq 0$.

Wireless network monitoring. We may further leverage the fact that $\mathbf{X}_t$ is typically correlated with $\mathbf{X}_{t-1}$ in order to improve the embeddings' accuracy over time. For example, suppose $\mathbf{X}_{t-m} = \ldots = \mathbf{X}_t = \mathbf{X}$ over some interval of length m. It is then prudent to estimate $\mathbf{X}$ by solving (3.1), but using the average $\bar{\mathbf{A}}_t = 1/m \sum_{k=t-m}^{t} \mathbf{A}_k$ instead (R. Tang et al., 2017). Note that $\bar{\mathbf{A}}_t$ may be interpreted as the adjacency matrix of a weighted graph. Edge weights can also be modeled by an RDPG, where now the embeddings are such that $\mathbf{X}\mathbf{X}^\top = \mathbb{E}[\mathbf{A}]$. Unlike the unweighted case, $\mathbb{E}[\mathbf{A}]$ are not probabilities. Still, under mild assumptions on the weights' distribution, the solution of (3.1) for weighted $\mathbf{A}$ is a consistent estimator of $\mathbf{X}$ as $N \to \infty$; see Theorem 4.4.1.

These observations motivate well the two-stage tracking system depicted in Fig. 3.8. The stream of adjacency matrices $\{\mathbf{A}_t\}$ is fed to the entry-wise filter $\mathbf{F}(z)$, which outputs $\{\mathbf{B}_t\}$. For instance, $\mathbf{F}(z)$ may be a moving average of fixed length m as before. If memory is at a premium, we may use a single-pole infinite impulse response (IIR) filter instead so that $\{\mathbf{B}_t\}$ is an exponentially-weighted moving average of the input adjacency matrices. We may even drop the filtering stage altogether (setting $m = 1$) to yield a least mean squares (LMS)-type online GD algorithm.

We now empirically demonstrate this simple tracking system yields accurate and stable embeddings of dynamic RDPG graphs. Consider a Wi-Fi network from which a monitoring system periodically acquires the Received Signal Strength Indicator (RSSI) between Access Points (APs) – a feature typically available in enterprise-

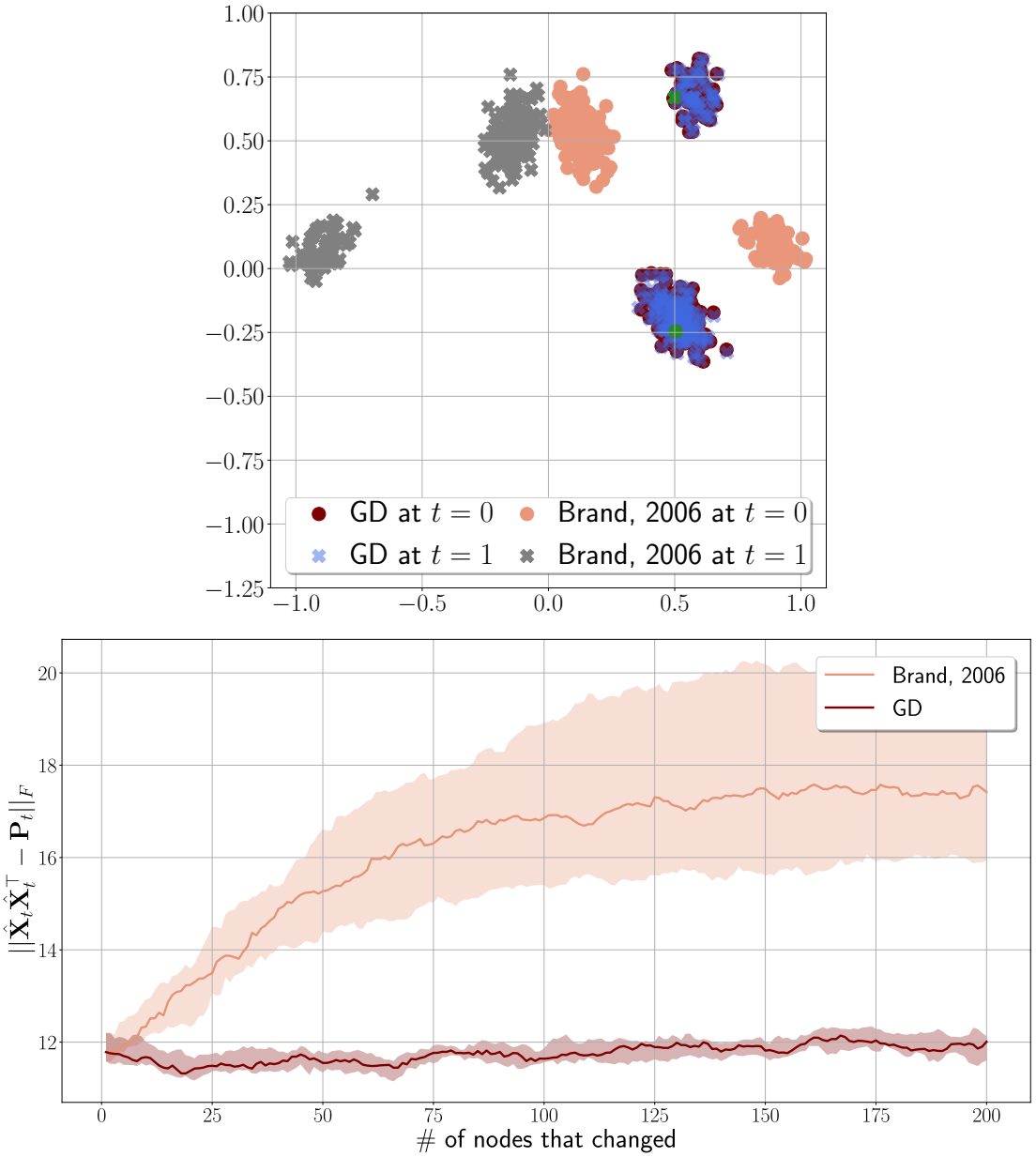

Figure 3.7: Two-block dynamic SBM in which a single node changes affiliation at each t. Comparison between online GD and recursive SVD (Brand, 2006). (top) Embeddings for the first two time-steps ($d = 2$); the node that changed communities is highlighted in green. Best viewed in a color display. Note how the change of a single node produces markedly different results for Brand (2006), whereas online GD offers stable estimates. (bottom) Evolution of $\|\hat{\mathbf{X}}_t\hat{\mathbf{X}}_t^\top - \mathbf{P}_t\|_F$. Solid line indicates median across ten realizations, with the range between first and third quartiles shown in a lighter color. Online GD exhibits uniformly bounded error, whereas Brand (2006) accumulates error as t grows.

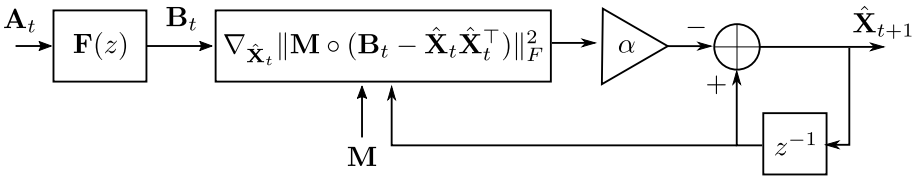

Figure 3.8: A diagram of the proposed tracking system. The entry-wise filter $\mathbf{F}(z)$ implements an averaging operator, e.g., a fixed-length moving average.

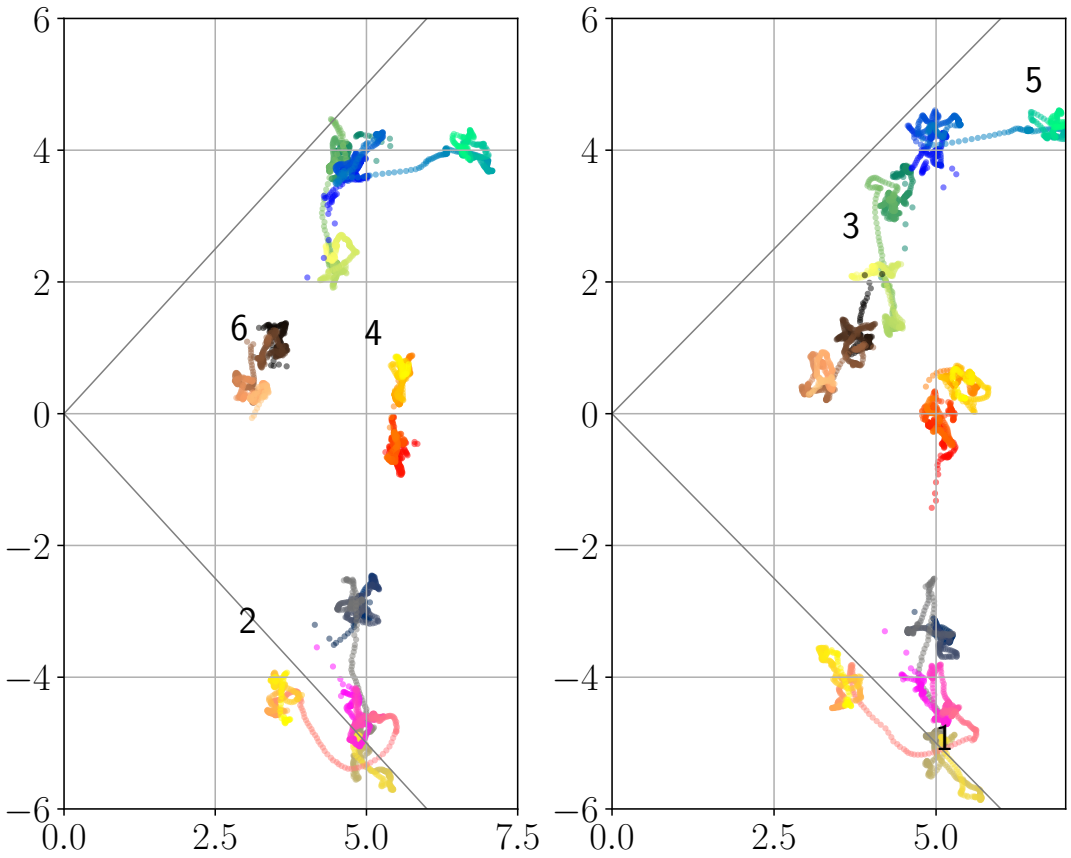

Figure 3.9: Embeddings $\hat{\mathbf{X}}_t^l$ (left) and $\hat{\mathbf{X}}_t^r$ (right) for the RSSI digraph ($d = 2$). Color palettes distinguish the APs and a lighter tone indicates larger values of t. Best viewed in a color display. The network's change at $t \approx 310$ is apparent. AP 4 was moved ($i = 4$) closer to the upper cluster of APs.

level deployments. We will use our GRL framework to flag network changes and eventually diagnose them. We analyze graphs $\mathbf{A}_t$ whose nodes are the APs and the edge weights are the measured RSSI values (plus a constant offset so that all values are positive). Since these measurements are typically not symmetric, we have a digraph sequence. We rely on the dataset described in Capdehourat et al. (2020), which consists of hourly measurements between $N = 6$ APs at a Uruguayan school, over almost four weeks ($m = 655$ graphs). During the monitoring period, the network administrator moved an AP ($i = 4$) at $t \approx 310$.

To track the AP embeddings, we run an online version of Algorithm 3 as schematically shown in the diagram of Fig. 3.8, but adapted to digraphs. This entails a retraction after the Riemannian GD step, not shown in the diagram. We use an IIR filter $\mathbf{F}(z)$ with a pole at 0.9. Furthermore, we adopt a fix stepsize $\alpha = 0.01$ instead of choosing it via the Armijo rule.

The evolution of the online Riemannian GD estimates $\hat{\mathbf{X}}_t^l$ and $\hat{\mathbf{X}}_t^r$ for $d = 2$

is shown in Fig. 3.9. Different color palettes are used to distinguish the nodes, and as t increases the colors become lighter. Note how at all times there are two (almost) orthogonal cluster of nodes: c1) APs 1 and 2 (in the lower part of the plots); and c2) APs 3, 5 and (to a lesser extent) 6. AP 4 is embedded between both communities for all t. Moreover, note how the *trajectory of each AP* can be split into a couple clear states, discernable as the colors transition from darker to lighter. This is indicative of the change in AP 4's position at roughly the middle of the monitoring period. Finally, movement within both AP clusters is mostly radial, hence dot products between cluster members are preserved. On the other hand, AP 4 moves tranversally closer to c2, consistent with the information provided by the network administrator. It appears as if it was moved closer to AP 5, and AP 1 remains its closest member from c1.

3.6.4.2 Time-varying node set

In dynamic environments it is not uncommon for nodes to join or leave the network. Going back to the wireless network test case, the question remains on how to proceed should an AP fail, or if the administrator decides to add a new one to improve coverage. Dealing with the former case is straightforward; if a node leaves the network at time t, we simply drop the corresponding row in $\hat{\mathbf{X}}_{t-1}$ and re-run the GD algorithm (warm-restarted from there).

Node additions require more thought. Suppose that a single node $i = N+1$ joins the network at time t. Let $\mathbf{a}_{N+1} = [A_{1,N+1}, \ldots, A_{N,N+1}]^\top \in \{0,1\}^N$ be the $(N+1)$-th column of $\mathbf{A}_t \in \{0,1\}^{N+1 \times N+1}$, excluding $A_{N+1,N+1} = 0$ and dropping the subindex t for notational convenience. Then given $\hat{\mathbf{X}}_{t-1} \in \mathbb{R}^{N \times d}$, we can embed node i by solving

$$\hat{\mathbf{x}}_{N+1} = \underset{\boldsymbol{\theta} \in \mathbb{R}^d}{\operatorname{argmin}} \|\mathbf{a}_{N+1} - \hat{\mathbf{X}}_{t-1}\boldsymbol{\theta}\|_2^2. \tag{3.24}$$

This simple but intuitive out-of-sample embedding procedure was studied in Levin et al. (2018), and shown to recover the true latent positions as $N \to \infty$. If several nodes are added at a given time-step, they can all be embedded by solving multiple LS problems like (3.24). However, this procedure disregards the information from the connections between new nodes. Furthermore, if the embeddings of existing nodes are not updated, their growing inaccuracies as $\mathbf{A}_t$ evolves will negatively impact future nodes' representations.

As we show in the following numerical experiments, these drawbacks can be overcome by running our online GD-based algorithms to update *all embeddings* $\hat{\mathbf{X}}_t$, initializing existing nodes with $\hat{\mathbf{X}}_{t-1}$ and new one(s) with $\hat{\mathbf{x}}_{N+1}$ as in (3.24).

Dynamic random graph with growing vertex set. Consider an Erdös-Rényi graph with a fixed connection probability $p = 0.1$, and initial number of nodes

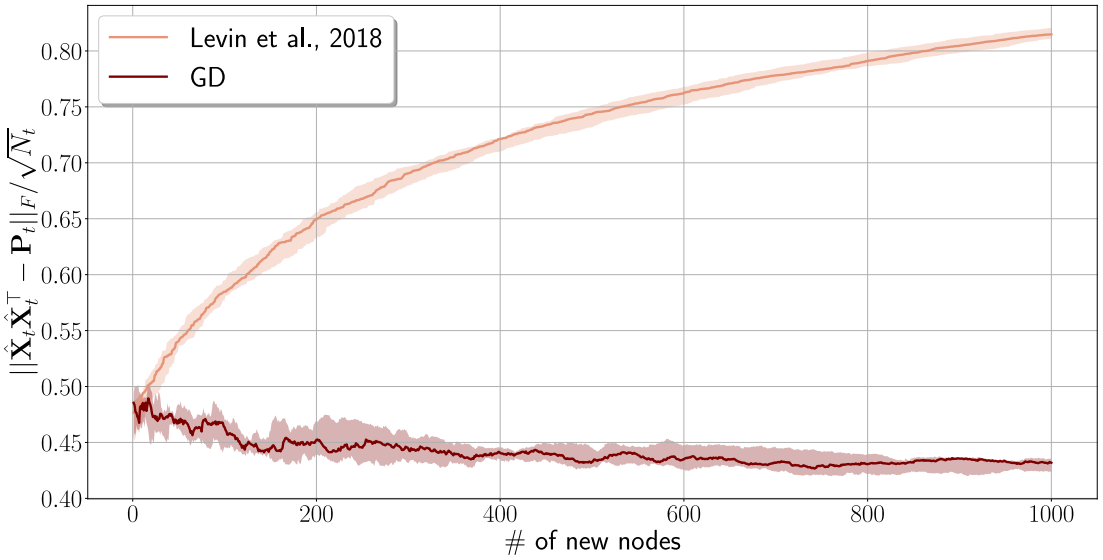

Figure 3.10: Dynamic Erdös-Rényi graph in which a single node is added at each t. Comparison between online GD and out-of-sample LS embedding Levin et al. (2018). Evolution of $\|\hat{\mathbf{X}}_t\hat{\mathbf{X}}_t^\top - \mathbf{P}_t\|_F/\sqrt{N_t}$. Solid line indicates median across ten realizations, with range between first and third quartiles shown in a lighter color. Once more, online GD exhibits uniformly bounded error, whereas the baseline method Levin et al. (2018) accumulates error as t grows.

$N_0 = 100$. At each time-step t we add a single node so that $N_t = N_{t-1} + 1$. The evolution of the error $\|\hat{\mathbf{X}}_t\hat{\mathbf{X}}_t^\top - \mathbf{P}_t\|_F/\sqrt{N_t}$ is shown in Fig. 3.10. Note how (carefully warm-restarted) online GD exhibits bounded error behavior, in stark contrast with repeated LS-based embeddings as in Levin et al. (2018). Admittedly, this gain in accuracy comes with a modest increase in computation (few GD steps), and identical memory footprint (i.e., storing the current embeddings and the new adjacency matrix) as the baseline in Levin et al. (2018).

Tracking international relations from UN voting data. Here we revisit the UN General Assembly voting data from Section 3.6.2. Following the same bipartite digraph construction procedure, we study all yearly graphs from 1955 to 2015. In this dynamic network we have a time-varying node set. Roll calls change from one year to the next, and also several countries joined the UN later (while others have ceased to exist). We embed the first graph from 1955 using Riemannian GD initialized at random (as before, using $d = 2$). For each successive year, we warm-restart Algorithm 3 with the embeddings from the previous year, while new nodes are initialized using the LS solution (3.24).

Fig. 3.11 depicts the embeddings of four countries: USA, Israel, Cuba, and the USSR (later, the Russian Federation). We use a similar visualization style as in Fig. 3.9, with different color palettes used to distinguish among countries, and lighter tones indicating more recent years. Observe how the representations for the USA and Israel remain strongly aligned over the entire time horizon, which is consistent

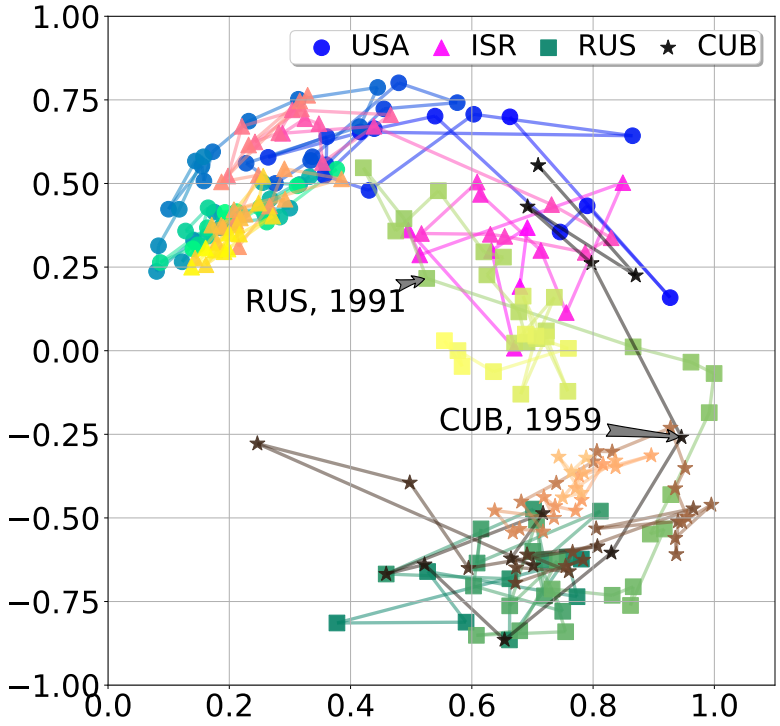

Figure 3.11: UN General Assembly voting data from 1955 to 2015. Evolution of nodal positions for the USA, Israel, Cuba, and the USSR (or, after 1991, the Russian Federation) estimated via online Riemannian GD. Color palettes distinguish the countries and a lighter tone indicates later years. Best viewed in a color display. Note how the USA and Israel remain strongly aligned over the entire span, with Cuba and the USSR shifting alignments depending of their political views.

with their longstanding agreement on UN resolution matters. The embedding for the USSR is initially (nearly) orthogonal to the USA and Israel, with Cuba initially showing a greater affinity to the USA/Israel block. This is consistent with Cold War geopolitics of the time. Then, after 1959, Cuba's position shifts to the lower half-plane, becoming more aligned with the USSR. This is expected given Cuba's sharp shift in foreign policy as a result of the Cuban revolution, with its ideology being in agreement with that of the USSR. This polarized scenario remained unchanged until 1991. That year the embedding for the USSR (now the Russian Federation) moves closer to the USA/Israel block, which reflects the politics of the Russian Federation in the aftermath of USSR's dissolution. Cuba remains at an (almost) orthogonal position from the USA/Israel block, with Russia eventually shifting to a middle ground after the mid-2000's.

3.7 Concluding remarks

We developed a gradient-based spectral-embedding framework to estimate latent positions of RDPGs. Relative to prior art our algorithmic approaches offer better representation at a competitive computational cost, and they are more broadly applicable to settings with incomplete, dynamic, and directed network data. We motivated and proposed a novel manifold-constrained formulation to embed directed RDPGs, and developed novel Riemannian GD iterations to estimate interpretable latent nodal positions. The effectiveness of the GRL framework is demonstrated via reproducible experiments with both synthetic and real (wireless network and United Nations voting) data. We made all our codes publicly available.

Appendix 3.A: Critical points for the unmasked objective

In this appendix we provide a proposition that describes all stationary points of the unmasked objective (3.3). To that end, we appeal to the eigendecomposition of $\mathbf{A}$. In that proposition we prove that any such critical point is of the form:

$$\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{D}}^{1/2}\mathbf{Q},$$

where $\tilde{\mathbf{U}}$ contains any d orthonormal eigenvectors of $\mathbf{A}$, $\tilde{\mathbf{D}}$ holds the corresponding eigenvalues as columns, and $\mathbf{Q} \in O(d)$.

Proposition 3.A.1. *Let $\mathbf{A} \in \mathbb{R}^{N\times N}$ be a symmetric positive semidefinite matrix with eigendecomposition*

$$\mathbf{A} = \sum_{i=1}^{N} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top,$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \geq 0$ and $\{\mathbf{u}_i\}_{i=1}^N, \mathbf{u} \in \mathbb{R}^N$ is an orthonormal basis of eigenvectors.

Then $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times d}$ is a stationary point of (3.3) if and only if there exists an index set $J \subseteq \{1,\ldots,N\}$ with $|J| = d$ and an orthonormal set $\{\mathbf{v}_j\}_{j\in J}, \mathbf{v}_j \in \mathbb{R}^d$, such that

$$\tilde{\mathbf{X}} = \sum_{j\in J} \sqrt{\lambda_j}\, \mathbf{u}_j \mathbf{v}_j^\top.$$

Proof. We proceed in two directions:
($\Leftarrow$) If $\tilde{\mathbf{X}}$ is of the given form, computing $\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$ yields

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top = \sum_{j,k\in J} \sqrt{\lambda_j\lambda_k}\, \mathbf{u}_j \mathbf{v}_j^\top \mathbf{v}_k \mathbf{u}_k^\top = \sum_{j\in J} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top,$$

because $\mathbf{v}_j^\top \mathbf{v}_k = \delta_{jk}$ by orthonormality. Thus,

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A} = \sum_{j\in J} \lambda_j \mathbf{u}_j \mathbf{u}_j^\top - \sum_{i=1}^{N} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top = -\sum_{i\notin J} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top.$$

Remember the gradient of our objective is $\nabla f(\mathbf{X}) = (\mathbf{X}\mathbf{X}^\top - \mathbf{A})\mathbf{X}$. Therefore

$$\nabla f(\tilde{\mathbf{X}}) = \left(-\sum_{i\notin J} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top\right)\left(\sum_{j\in J} \sqrt{\lambda_j}\mathbf{u}_j \mathbf{v}_j^\top\right) = -\sum_{i\notin J, j\in J} \lambda_i \sqrt{\lambda_j}\mathbf{u}_i \mathbf{u}_i^\top \mathbf{u}_j \mathbf{v}_j^\top.$$

Now, for any $i \notin J$ and $j \in J$, we have $\mathbf{u}_i^\top \mathbf{u}_j = 0$, so each term in the above sum vanishes, and thus $(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A})\tilde{\mathbf{X}} = \mathbf{0}$. Therefore, $\tilde{\mathbf{X}}$ is a stationary point.

($\Rightarrow$) Assume that $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times d}$ is a stationary point of f, so

$$\nabla f(\tilde{\mathbf{X}}) = (\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A})\tilde{\mathbf{X}} = \mathbf{0}. \tag{3.25}$$

Since $\operatorname{rank}(\tilde{\mathbf{X}}) = d$, $W := \operatorname{Im}(\tilde{\mathbf{X}})$ is a d-dimensional subspace of $\mathbb{R}^N$. Let $\mathbf{w} \in W$. Then $\mathbf{w} = \tilde{\mathbf{X}}\mathbf{v}$ for some $\mathbf{v} \in \mathbb{R}^d$. Multiplying both sides of (3.25) by $\mathbf{v}$, we get

$$(\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top - \mathbf{A})\,\tilde{\mathbf{X}}\mathbf{v} = \mathbf{0} \Rightarrow \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top \mathbf{w} = \mathbf{A}\mathbf{w}. \tag{3.26}$$

Hence, W is invariant under $\mathbf{A}$. So we can choose an orthonormal basis of V consisting of eigenvectors of $\mathbf{A}$, i.e., there exist indices $J \subset \{1, \ldots, N\}$ with $|J| = d$ such that:

$$W = \operatorname{span}\{\mathbf{u}_i : i \in J\}.$$

Note that $\tilde{\mathbf{X}}$ has rank d, so we can write

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top = \sum_{i\in J} \mu_i\, \mathbf{u}_i \mathbf{u}_i^\top,$$

where the $\mu_i > 0$ are the nonzero eigenvalues of $\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$. Equation (3.26) then implies that for $\mathbf{w} = \mathbf{u}_i$

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top \mathbf{u}_i = \mu_i \mathbf{u}_i = \mathbf{A}\mathbf{u}_i = \lambda_i \mathbf{u}_i \Rightarrow \mu_i = \lambda_i.$$

Hence

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top = \sum_{i\in J} \lambda_i\, \mathbf{u}_i \mathbf{u}_i^\top. \tag{3.27}$$

But since $W = \operatorname{Im}(\tilde{\mathbf{X}})$, we can write:

$$\tilde{\mathbf{X}} = \sum_{i\in J} \mathbf{u}_i\, \mathbf{c}_i^\top,$$

for some $\mathbf{c}_i \in \mathbb{R}^d$. Then

$$\tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top = \sum_{i,j \in J} \mathbf{u}_i \, \mathbf{c}_i^\top \mathbf{c}_j \, \mathbf{u}_j^\top .$$

Comparing to (3.27) we see that $\mathbf{c}_i^\top \mathbf{c}_j = 0$ for $i \neq j$ and $\|\mathbf{c}_i\|_2^2 = \lambda_i$. Hence $\mathbf{c}_i = \sqrt{\lambda_i}\mathbf{v}_i$ for some orthonormal vectors $\mathbf{v}_i \in \mathbb{R}^d$. Thus:

$$\tilde{\mathbf{X}} = \sum_{j \in J} \sqrt{\lambda_j} \, \mathbf{u}_j \mathbf{v}_j^\top ,$$

as desired. □

Chapter 4

A weighted RDPG model

In this chapter we extend the RDPG model to account for *weighted* graphs, endowed with a weight map $w : E \mapsto \mathbb{R}_+$ that assigns a nonnegative value to each edge. The adjacency matrix entries for the weighted graph are $W_{ij} = W_{ji} = w(i, j)$ for all $(i, j) \in E$, while the absence of an edge is represented as $W_{ij} = W_{ji} = 0$. This formulation naturally encompasses unweighted graphs as a special case, where edge weights are constrained to binary values (i.e., $w \equiv 1$).

This chapter is adapted from the paper Marenco et al. (2025).

4.1 Related work

Several extensions of the standard (or *vanilla*) RDPG model have been developed to incorporate edge weights. A common approach is to introduce a parametric distribution $F_{\boldsymbol{\theta}}$ with parameter $\boldsymbol{\theta} \in \mathbb{R}^L$ to model weighted adjacency entries (see DeFord & Rockmore, 2016; R. Tang et al., 2017). In this formulation, each node $i \in V$ is assigned a collection of latent vectors $\mathbf{x}_i[l] \in \mathbb{R}^d$ for $l = 1, \ldots, L$, such that the weight of an edge between nodes i and j follows the distribution

$$W_{ij} \sim F_{(\mathbf{x}_i^\top[1]\mathbf{x}_j[1], \ldots, \mathbf{x}_i^\top[L]\mathbf{x}_j[L])},$$

independently across edges. To model sparse graphs, this distribution may include a point mass at zero. The classic RDPG model is recovered by setting $F_{\boldsymbol{\theta}}$ as a Bernoulli distribution with success probability $\theta \in [0, 1]$. Despite its flexibility, this approach has some noteworthy limitations. A key drawback is that all edges must share the same parametric family of weight distributions, differing only in their parameters. While mixture models can introduce some heterogeneity, they still require an explicit assumption about the underlying distribution families. This requirement significantly restricts the model's applicability, particularly if edges follow heterogeneous, unknown, or multimodal weight distributions.

To address this shortcoming, recent work by Gallagher et al. (2023) proposes a nonparametric alternative. Therein each node has a single associated latent position $\mathbf{z}_i \in \mathcal{Z}$, which is endowed with a probability distribution F. It is postulated that, given a family $\{H(\mathbf{z}_1, \mathbf{z}_2) : \mathbf{z}_1, \mathbf{z}_2 \in \mathcal{Z}\}$ of symmetric real-valued distributions, there exists a map $\boldsymbol{\phi} : \mathcal{Z} \mapsto \mathbb{R}^d$ such that if $W_{ij} \sim H(\mathbf{z}_i, \mathbf{z}_j)$, then $\mathbb{E}[W_{ij}] = \boldsymbol{\phi}^\top(\mathbf{z}_i)\mathbf{I}_{p,q}\boldsymbol{\phi}(\mathbf{z}_j)$, where $\mathbf{I}_{p,q}$ is a diagonal matrix of p ones and q minus ones, such that $p + q = d$. This diagonal matrix facilitates modeling heterophilic (or disassortative) behavior, as in Rubin-Delanchy et al. (2022). Interestingly, one can consistently recover $\mathbf{x}_i = \boldsymbol{\phi}(\mathbf{z}_i)$ via the ASE of $\mathbf{W}$, and the estimated $\hat{\mathbf{x}}_i$'s are asymptotically normal. While this method improves flexibility and avoids restrictive parametric assumptions, it can only recover $\boldsymbol{\phi}(\mathbf{z}_i)$, i.e., the latent positions for the mean adjacency matrix $\mathbb{E}[\mathbf{W}]$. Consequently, the model is unable to differentiate between pairs of edges that stem from distinct distributions sharing an identical mean. These challenges highlight the ongoing need for more expressive, discriminative, and robust latent space models for weighted graphs.

4.2 Contributions and chapter outline

Instead, we propose that the sequence of nodal vectors be related to the weight distribution's moment-generating function (MGF). Assume each node has a sequence of latent positions $\{\mathbf{x}_i[k]\}_{k\geq 0}$ such that the inner products $\{\mathbf{x}_i^\top[k]\mathbf{x}_j[k]\}_{k\geq 0}$ form an admissible moment sequence. Our weighted (W)RDPG model specifies the k-th order moments of the random entries in the adjacency matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ are given by $\mathbb{E}\left[W_{ij}^k\right] = \mathbf{x}_i^\top[k]\mathbf{x}_j[k]$, for all $k \geq 0$ (see Section 4.3.1).

The WRDPG model offers several key advantages: (i) it is nonparametric; (ii) it considers higher-order moments beyond the mean; while providing (iii) statistical guarantees for the associated estimator; and (iv) a generative mechanism that can reproduce both the structure and the weights' distribution of real networks. Next, we elaborate on these attractive features to better position our technical contributions in context. Unlike DeFord and Rockmore (2016) and R. Tang et al. (2017), the nonparametric WRDPG graph modeling framework does not require prior assumptions about a specific weight distribution. Nodal vectors can be used to describe high-order moments of said distribution, thus enhancing the model's representation and discriminative power. For example, different from Gallagher et al. (2023) the WRDPG model can distinguish between distributions with the same mean but differing standard deviations; see also Section 4.3.4 and Figure 4.5 for an illustrative example.

Furthermore, framing our model in terms of (symmetric) adjacency matrices allows us to define latent position estimators based on their spectral decomposition;

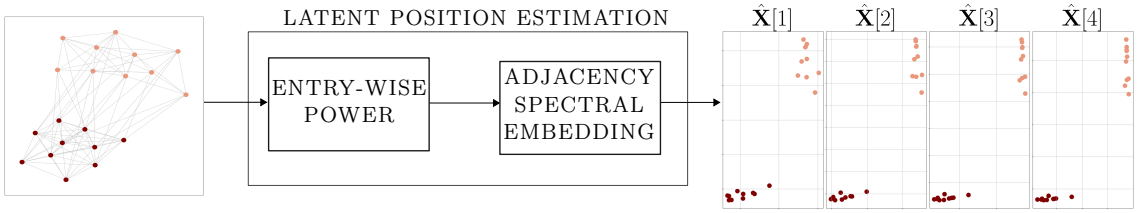

Figure 4.1: Latent position estimation. Given an adjacency matrix $\mathbf{A}$ we compute its k-th entry-wise power $\mathbf{A}^{(k)}$. The ASE of $\mathbf{A}^{(k)}$ yields the estimates $\hat{\mathbf{X}}[k]$; see also Section 4.3.2.

see Section 4.3.2. Statistical guarantees can be derived using tools to control the matrix spectrum under perturbations. As shown in Marenco et al. (2022), for each fixed moment index k, the latent position matrix $\mathbf{X}[k]$ can be consistently recovered (up to an orthogonal transformation) via the ASE of $\mathbf{W}^{(k)}$, the matrix of entry-wise k-th powers of $\mathbf{W}$; see Figure 4.1 for a schematic diagram of the latent position estimation procedure. In Section 4.4 we generalize the result in Marenco et al. (2022) by proving that asymptotic consistency holds for a stronger convergence metric that ensures uniform convergence for all $\hat{\mathbf{x}}_i$'s. We also establish that the estimator is asymptotically normal as $N \to \infty$, a result that is novel in our WRDPG setup.

In Section 4.5 we show how one can generate graphs adhering to the proposed WRDPG model, which is non-trivial in a nonparameteric setting and was not considered in Gallagher et al. (2023). We develop a methodology to sample random weighted graphs whose edge weight distribution is defined by a sequence of moments given by inner products of connected nodes' latent positions. Depending on the characteristics of the weight distribution–whether discrete, continuous, or a mixture–we can solve for the latent position sequence to ensure their inner products match the prescribed moments. For real-valued (continuous) weights, we rely on the maximum entropy principle subject to moment constraints. We introduce a primal-dual approach to find a probability density function that maximizes entropy, offering improved numerical stability relative to previous algorithms (Saad & Ruai, 2019). Results for mixture distributions allow us to generate graphs that simultaneously replicate the sparsity pattern of the network and the edge weights. In Figure 4.2 we present a schematic diagram for the graph generation pipeline.

Moreover, given an observed weighted network (instead of the ground-truth latent position sequence as in Figure 4.2), we show how we can use this generative procedure to sample graphs that are similar to the original one, in a well-defined statistical sense. This can be useful for several statistical inference tasks involving network data. If, for instance, one would like to assess the significance of some observed graph characteristic, this procedure allows to generate several 'comparable' graphs and construct judicious reference distributions (see Kolaczyk, 2009, Section 6.2). In a related use case, one could perform hypothesis testing to determine if a

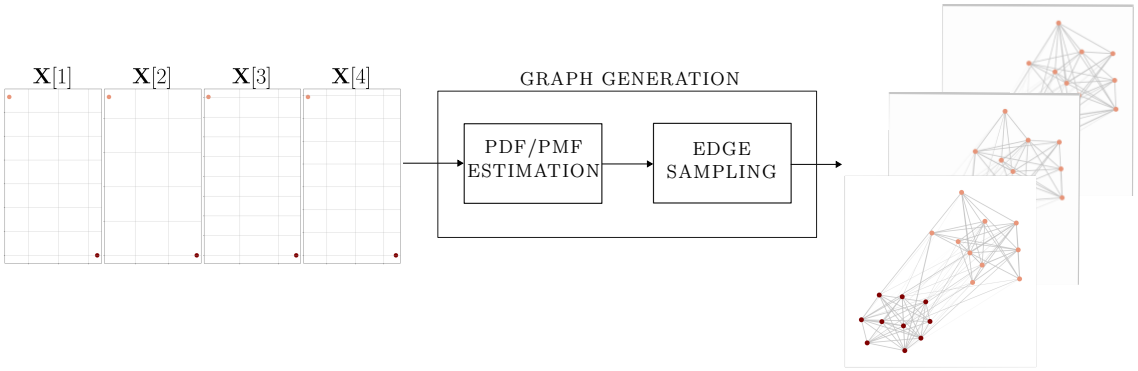

Figure 4.2: Graph generation. Given the latent positions of each vertex $\{\mathbf{X}[k]\}_{k\geq 0}$, we estimate a weight distribution whose sequence of moments is given by the corresponding dot products. Edge weights are then sampled from this estimated distribution; see also Section 4.5.

given graph adheres to the WRDPG model.

The rest of this chapter is organized as follows. In Section 4.3 we formally define the WRDPG model, describe the embedding method, and present several illustrative examples. The estimation problem is addressed in Section 4.4, and the proofs of the asymptotic statistical guarantees are presented in detail. Finally, in Section 4.5, we present our methodology for generating WRDPG graphs. We include several reproducible examples to demonstrate the ability of the generative framework to produce weighted graphs with desired characteristics. All figures in the remainder of this chapter can be reproduced using the code available at https://github.com/bmarenco/wrdpg.

4.3 Weighted RDPG model

Here we define the WRDPG model and the ASE-based latent position estimator. Illustrative examples are presented to ground these concepts. Furthermore, we show the discriminative power for community detection inherited from considering higher-order moments beyond the mean. Finally, we discuss the impact of the number of nodes on the accuracy of the moment sequence reconstruction.

4.3.1 Model specification

In this section, we formally define our WRDPG model. We follow the rationale of the vanilla RDPG and define the model in terms of random latent position sequences per node. As previewed in Section 4.2, the inner product of the latent positions will be related to the MGF of the edge weights' distribution, thus requiring some preliminary definitions.

Definition 4.3.1 (Admissible moment sequence)**.** A sequence $\{m[k]\}_{k\geq 0}$ of real numbers is an admissible moment sequence if $m[0] = 1$ and for all $p \geq 0$ the matrix

$$\mathbf{M} = \begin{pmatrix} m[0] & m[1] & m[2] & \dots & m[p] \\ m[1] & m[2] & m[3] & \dots & m[p+1] \\ m[2] & m[3] & m[4] & \dots & m[p+2] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m[p] & m[p+1] & m[p+2] & \dots & m[2p] \end{pmatrix} \in \mathbb{R}^{(p+1)\times(p+1)}$$

whose entries are $M_{ij} = m[i+j-2]$ is PSD.

Remark 4.3.1. Definition 4.3.1 is enough to guarantee that there exists a probability measure μ in $\mathbb{R}$ such that $m[k]$ is the k-th moment of μ.

Definition 4.3.2 (Weighted inner-product distribution)**.** Let F be a probability distribution with support $\text{supp}F = \mathcal{X} \subset (\mathbb{R}^d)^\infty$. We say that F is a weighted inner-product distribution if for all $\{\mathbf{x}[k]\}_{k\geq 0}, \{\mathbf{y}[k]\}_{k\geq 0} \in \mathcal{X}$, the sequence $\{\mathbf{x}^\top[k]\mathbf{y}[k]\}_{k\geq 0}$ is an admisible moment sequence.

Definition 4.3.3 (WRDPG)**.** Let F be a weighted inner-product distribution and let $\{\mathbf{x}_1[k]\}_{k\geq 0}$, $\{\mathbf{x}_2[k]\}_{k\geq 0}$, $\dots$, $\{\mathbf{x}_N[k]\}_{k\geq 0}$ be i.i.d. with distribution F. Given the sequence $\mathbf{X}_k := \{\mathbf{X}[k]\}_k$, with $\mathbf{X}[k] = [\mathbf{x}_1[k], \dots, \mathbf{x}_N[k]]^\top \in \mathbb{R}^{N\times d}$, the weighted (W)RDPG model specifies the MGF of a random adjacency matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ as

$$\mathbb{E}\left[e^{tW_{ij}}|\mathbf{X}_k\right] = \sum_{k=0}^{\infty} \frac{t^k \mathbb{E}\left[W_{ij}^k\right]}{k!} = \sum_{k=0}^{\infty} \frac{t^k \mathbf{x}_i^\top[k]\mathbf{x}_j[k]}{k!} \tag{4.1}$$

and the entries W_{ij} are independent, i.e., edge independent. In such a case, we write $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$.

Remark 4.3.2 (Nonidentifiability of latent positions). As is the case for the vanilla RDPG model, latent positions are invariant to orthogonal transformations, since for any $\mathbf{Q} \in O(d)$, $\mathbf{X}[k]\mathbf{X}^\top[k] = \mathbf{X}[k]\mathbf{Q}\left(\mathbf{X}[k]\mathbf{Q}\right)^\top$. This implies that given a latent position sequence $\mathbf{X}_k$, for each index k one may choose a matrix $\mathbf{Q}_k \in O(d)$ (which may vary with k) and construct a sequence $\mathbf{Y}_k := \{\mathbf{X}[k]\mathbf{Q}_k\}_k$, which will result in the same distribution of graphs as that given by $\mathbf{X}_k$.

4.3.2 Estimation of latent positions

The vectors $\mathbf{x}_i[k]$ can be estimated via an ASE of the matrix $\mathbf{W}^{(k)}$, the entry-wise k-th power of an observed symmetric weighted adjacency matrix $\mathbf{W}$. Indeed, for each index k, we stack the latent positions of all nodes into the matrix $\mathbf{X}[k] = [\mathbf{x}_1[k], \dots, \mathbf{x}_N[k]]^\top \in \mathbb{R}^{N \times d}$. If one had access to the moment matrix $\mathbf{M}_k := \mathbf{X}[k]\mathbf{X}^\top[k]$, it would be straightforward to recover $\mathbf{X}[k]$ (up to an orthogonal transformation) from the eigendecomposition of $\mathbf{M}_k$. However, since $\mathbf{M}_k$ is typically unobserved, we instead rely on the observed weighted adjacency matrix $\mathbf{W}$. Because each entry of $\mathbf{W}^{(k)}$ has expectation equal to the corresponding entry of $\mathbf{M}_k$, we approximate $\mathbf{X}[k]$ by solving [cf. (2.2)]

$$\hat{\mathbf{X}}[k] \in \operatorname*{argmin}_{\mathbf{X} \in \mathbb{R}^{N \times d}} \left\| \mathbf{W}^{(k)} - \mathbf{X}\mathbf{X}^\top \right\|_{\mathrm{F}}^2,$$

for all $k \geq 0$. A solution is readily obtained by setting $\hat{\mathbf{X}}[k] = \hat{\mathbf{U}}_k \hat{\mathbf{D}}_k^{1/2}$, where $\mathbf{W}^{(k)} = \mathbf{U}_{W_k} \mathbf{D}_{W_k} \mathbf{U}_{W_k}^\top$ is the eigendecomposition of $\mathbf{W}^{(k)}$, $\hat{\mathbf{D}}_k \in \mathbb{R}^{d \times d}$ is the diagonal matrix of the d largest-magnitude eigenvalues, and the columns of $\hat{\mathbf{U}}_k \in \mathbb{R}^{N \times d}$ contain the corresponding eigenvectors. In Appendix 4.A, we show that, under mild assumptions on the weighted inner-product distribution F (Assumptions 2 and 3 in Section 4.4), the top d eigenvalues of $\mathbf{W}^{(k)}$ are nonnegative, ensuring that $\hat{\mathbf{X}}[k]$ is well-defined. We refer to this estimator as the ASE of $\mathbf{X}[k]$.

4.3.3 Examples

Next, we derive expressions for the latent position vectors in some simple, yet classical, models. We also simulate such WRDPG instances and show that latent position estimation via the ASE yields satisfactory results. We will derive statistical guarantees for ASE in Section 4.4, justifying our empirical findings.

4.3.3.1 Erdös-Rényi graph with Gaussian weights

Consider a graph with $N = 1000$ nodes. We first sample the presence or absence of each edge independently as Bernoulli(p). Then, for each edge we independently sample its weight from a $\mathcal{N}(\mu, \sigma^2)$ distribution. This means that W_{ij} is either 0 with probability $1 - p$ or follows a $\mathcal{N}(\mu, \sigma^2)$ distribution with probability p. In this setup, it is enough to choose $d = 1$, and the latent position sequence for every node is $x[0] = 1$ and $x[k] = \sqrt{p m_{\mathcal{N}}[k]}$ for $k \geq 1$, where $m_{\mathcal{N}}[k]$ is the k-th moment of the normal distribution with parameters μ and σ^2.

Figure 4.3 shows a histogram of the recovered latent positions up to order $k = 6$, with $p = 0.5$, $\mu = 1$ and $\sigma = 0.1$. The red dashed line indicates the true latent

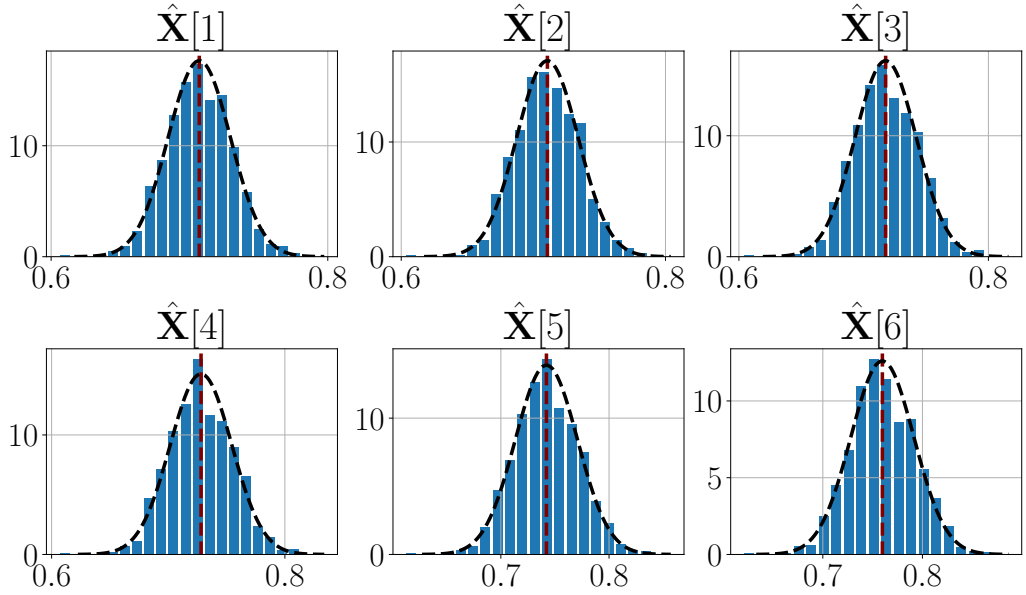

Figure 4.3: True (dashed vertical line) and estimated (histograms) latent positions for an Erdös-Rényi model with $\mathcal{N}(1, 0.01)$ weights. Pdf for limiting Gaussians, as given by Corollary 4.4.4, are plotted with dashed lines in each panel.

positions in each case. Apparently, for each k the estimated positions follow a normal distribution centered around the ground-truth value $x[k]$. This observation is supported by plotting the probability density function (pdf) of the limiting Gaussian distribution–that is, the distribution the estimated embeddings are expected to follow as $N \to \infty$, according to Theorem 4.4.2, which is proven later in this chapter; see also Corollary 4.4.4.

4.3.3.2 Two-block SBM with arbitrary weights' distribution

In this case, the setup is similar as before; only now is the presence or absence of an edge given by a 2-block SBM. That is, each node belongs to exactly one of two communities, and an edge is formed between two nodes with probability p_1 if both belong to community 1, with probability p_2 if both belong to community 2, and with probability q if they belong to different communities. So, conditioned on individual node assignments to communities, the presence or absence of edges is given by the block probability matrix:

$$\mathbf{B} = \begin{pmatrix} p_1 & q \\ q & p_2 \end{pmatrix}. \tag{4.2}$$

After sampling edges, all weights are independently sampled from the same arbitrary edge-weight distribution.

Let $\mathbf{x}_{C_1}, \mathbf{x}_{C_2} \in (\mathbb{R}^2)^\infty$ denote the latent position sequence for each community. To compute the analytical latent positions for this model, we begin by looking at the mean, i.e., $\mathbf{x}_{C_1}[1], \mathbf{x}_{C_2}[1] \in \mathbb{R}^2$. If two nodes belong to community 1, then an edge

is formed between them with probability p_1, and its weight follows the prescribed distribution. This implies that

$$\mathbf{x}_{C_1}^\top[1]\mathbf{x}_{C_1}[1] = p_1 m_d[1] \Rightarrow \|\mathbf{x}_{C_1}[1]\| = \sqrt{p_1 m_d[1]},$$

where $m_d[1]$ is the first moment (the mean) of the chosen weights' distribution. Due to the nonidentifiability of latent positions, we can arbitrarily place the latent positions for community 1 along the x-axis, and therefore choose:

$$\mathbf{x}_{C_1}[1] = (\sqrt{p_1 m_d[1]}, 0)^\top. \tag{4.3}$$

Similarly, we conclude that for community 2 we must have $\|\mathbf{x}_{C_2}[1]\| = \sqrt{p_2 m_d[1]}$. Also, an edge between two nodes in different communities is present with probability q and its weight follows the prescribed distribution, so

$$\mathbf{x}_{C_1}^\top[1]\mathbf{x}_{C_2}[1] = q m_d[1].$$

From these two constraints, using (4.3) we conclude that

$$\mathbf{x}_{C_2}[1] = \left(q\sqrt{\frac{m_d[1]}{p_1}}, \sqrt{m_d[1]\left(p_2 - \frac{q^2}{p_1}\right)}\right)^\top.$$

Following the analysis above, one can derive the higher-order terms in the per-community sequences. Indeed, if $m_d[k]$ is the k-th moment for the chosen weights' distribution, imposing that the inner products between latent positions equals the moments of inter- and intra-communities connections leads to:

$$\begin{aligned}
\|\mathbf{x}_{C_1}[k]\|^2 &= p_1 m_d[k] \\
\|\mathbf{x}_{C_2}[k]\|^2 &= p_2 m_d[k] \\
\mathbf{x}_{C_1}^\top[k]\mathbf{x}_{C_2}[k] &= q m_d[k].
\end{aligned}$$

Again, arbitrarily placing the latent positions for community 1 along the x-axis we can find the latent positions for each community:

$$\begin{aligned}
\mathbf{x}_{C_1}[k] &= \left(\sqrt{p_1 m_d[k]}, 0\right)^\top \\
\mathbf{x}_{C_2}[k] &= \left(q\sqrt{\frac{m_d[k]}{p_1}}, \sqrt{m_d[k]\left(p_2 - \frac{q^2}{p_1}\right)}\right)^\top.
\end{aligned} \tag{4.4}$$

Note that (4.4) is valid for $k \geq 1$. For $k = 0$ we can arbitrarily set $\mathbf{x}_{C_i}[0] = (1, 0)^\top$ for $i = 1, 2$ in order to: *a*) maintain the dimensionality of the embeddings; and *b*)

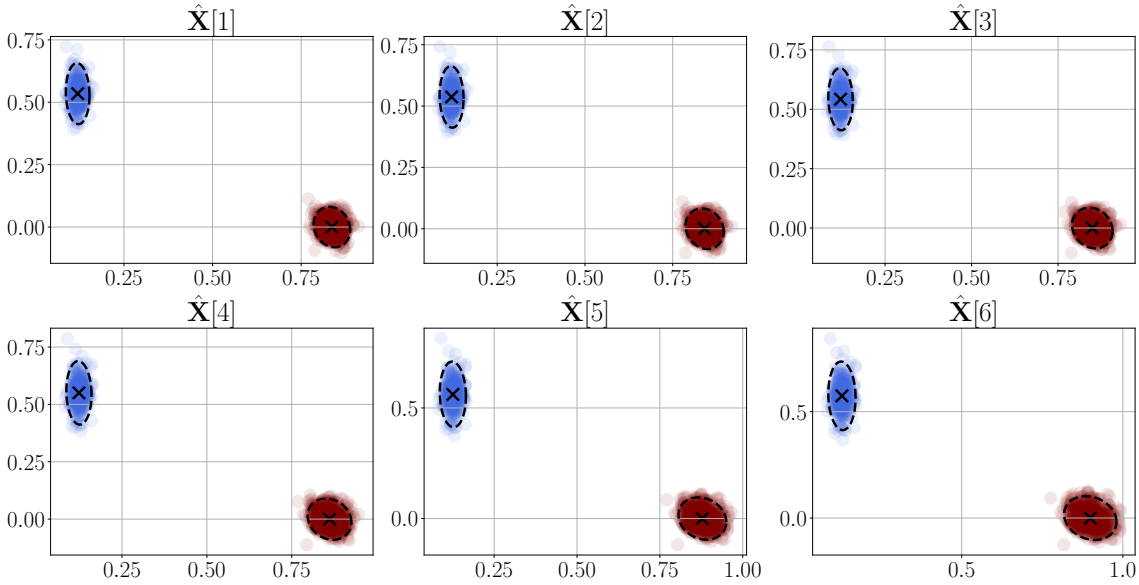

Figure 4.4: Estimated (blue and red circles) and true latent positions (black crosses) for a two-block SBM with $\mathcal{N}(1, 0.01)$ weights. The 95% confidence level sets for the limiting normal distributions, as given by Corollary 4.4.4, are shown as dashed lines.

force the 0-th moment of each edge to be equal to 1 [cf. Definition 4.3.1].

We simulated the above setup for a network with $N = 1000$ nodes (700 in community 1 and 300 in community 2), block probability matrix

$$\mathbf{B} = \begin{pmatrix} 0.7 & 0.1 \\ 0.1 & 0.3 \end{pmatrix},$$

and weights sampled from a $\mathcal{N}(\mu, \sigma^2)$ distribution, with $\mu = 1$ and $\sigma = 0.1$. Results for the ASE of $\mathbf{W}^{(k)}$ up to order $k = 6$ are shown in Figure 4.4. As expected, the estimated embeddings for each community are centered around the analytically derived ones in (4.4), with an ellipse-like outline that suggests an approximately normal distribution. As with the previous example, this is corroborated by plotting the 95% confidence level sets for the limiting Gaussians for each community, as given by Corollary 4.4.4.

4.3.4 Discriminative power of higher-order spectral embeddings

To demonstrate the ability of our model to distinguish between communities, we simulate a two-block weighted SBM consisting of $N = 2000$ nodes. Edges are established independently with probability $p = 0.5$. Edge weights follow a Gaussian distribution with mean $\mu = 5$ and standard deviation $\sigma = 0.1$, except for those between the second block of nodes, indexed $i = 1001, \ldots, 2000$, where the weights instead follow a Poisson distribution with rate parameter $\lambda = 5.1$.

For the same reasons as in the previous example, in this setting the matrix $\mathbf{X}[k]$ has at most two distinct columns for each k. Figure 4.5 displays the estimated embeddings $\hat{\mathbf{x}}_i[k]$ obtained via the ASE of $\mathbf{W}^{(k)}$ for $k = 1, 2, 3$ with embedding dimension $d = 2$, where nodes are color-coded by community membership. The 95% confidence level sets for the limiting normal distributions, as given by Corollary 4.4.4, are shown in Figure 4.5 as dashed lines. Notably, for each value of k, the simulated points closely follow the normal distribution predicted by the theorem in the large-N limit, providing further empirical support for the asymptotic result.

Observe that for $k = 1$, the node embeddings are nearly indistinguishable across the two communities. This is expected, as the vectors $\hat{\mathbf{x}}_i[1]$ cluster around the points $(\sqrt{\mu p}, 0)^\top \approx (1.58, 0)^\top$ for the gaussian community, and $(\sqrt{\mu p}, \sqrt{p(\lambda - \mu)})^\top \approx (1.58, 0, 22)^\top$ for the Poisson community, reflecting the almost identical expected weight of edges in both distributions. When $k = 2$, the embeddings begin to reveal community structure, as shown in the center panel of Figure 4.5. In this simplified example, closed-form expressions for higher-order embeddings can be readily derived as well. Assuming, without loss of generality, that the embeddings $\mathbf{x}_i[k]$ lie along the x-axis for $i = 1, \ldots, 1000$, we obtain:

$$\mathbf{x}_i[k] = \begin{cases} \left(\sqrt{pm_k^N}, 0\right)^\top, & i \leq 1000, \\ \left(\sqrt{pm_k^N}, \sqrt{p(m_k^P - m_k^N)}\right)^\top, & i > 1000, \end{cases}$$

where m_k^N and m_k^P are the k-th moments of the univariate $\mathcal{N}(\mu, \sigma^2)$ and Poisson distributions, respectively. For $k = 2$, these correspond to the approximate coordinates $(3.54, 0)^\top$ and $(3.54, 1.75)^\top$ for the two groups. However, since the confidence sets of the limiting multivariate Gaussians are still close, the separation between the communities is not yet clearly pronounced. At $k = 3$, the confidence sets no longer intersect, and the embeddings exhibit a clear separation between the two blocks, as shown in the right panel of Figure 4.5.

Recall that the model proposed in Gallagher et al. (2023) is limited to embeddings derived solely from the mean of the weight distribution, specifically, the ASE of the matrix $\mathbf{W}^{(1)} = \mathbf{W}$. In the weighted SBM described above, both blocks are constructed to have almost identical expected edge weights. As a consequence, the embeddings $\hat{\mathbf{x}}_i[1]$ obtained via an ASE of $\mathbf{W}$ are concentrated around the same point in latent space regardless of the underlyingblock membership, and meaningful separation between the two blocks begins to emerge only when considering higher-order embeddings. Therefore, approaches restricted to first-order information, such as Gallagher et al. (2023), are inherently incapable of discriminating between communities whose structure is encoded in the higher-order moments of edge weights.

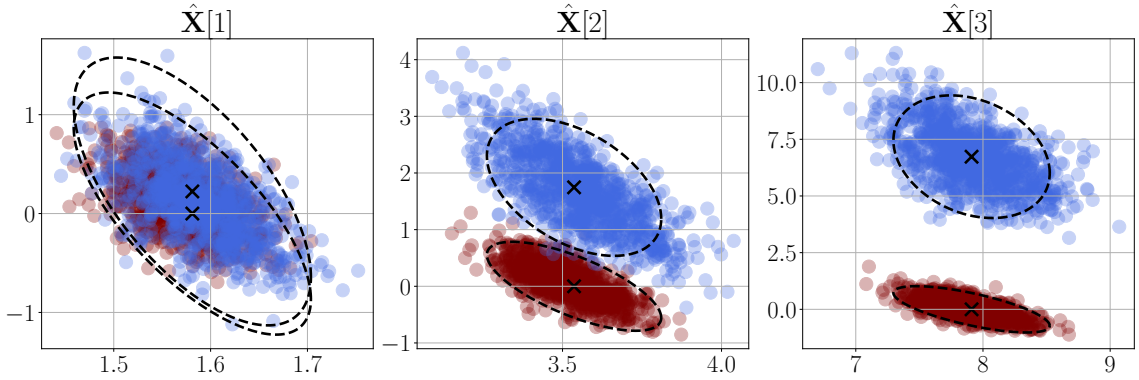

Figure 4.5: Theoretical latent positions (black crosses) and ASE embeddings of $\mathbf{W}^{(k)}$ for Gaussian ($\mu = 5$ and $\sigma = 0.1$; in red) and Poisson ($\lambda = 5.1$; in blue) distributed weights for $d = 2$ and $k = 1$ (left), $k = 2$ (center), and $k = 3$ (right). Nodes with different weight distributions are clearly revealed for $k = 3$, but they overlap for $k = 1$. The 95% confidence level sets for the limiting normal distributions, as given by Theorem 4.4.2, are shown as dashed lines.

4.3.5 Accuracy of moment recovery with varying number of nodes

Accurate estimation of higher-order moments is known to be highly sensitive to the amount of data available. In particular, the variance of moment estimators tends to grow rapidly with the order of the moment, requiring markedly larger sample sizes to obtain stable estimates (Bourin & Bondon, 1998). This is because higher-order moments are dominated by extreme values in the data, making them especially prone to noise and outliers. Admittedly, this a challenge facing the proposed model.

To illustrate how this behavior affects the WRDPG model, we simulate a two-block weighted SBM with $N = 2000$ nodes, where 70% of them are assigned to community 1. The interconnection probabilities are given by the matrix

$$\mathbf{B} = \begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.5 \end{pmatrix},$$

and edge weights are sampled from a Gaussian distribution with mean $\mu = 1$ and standard deviation $\sigma = 0.5$.

Figure 4.6 depicts the estimated latent position matrices $\hat{\mathbf{X}}[k]$ corresponding to moments $k = 1$ and $k = 4$. To assess the quality of the embeddings, we compare the entries of the empirical moment matrices $\hat{\mathbf{M}}[k] = \hat{\mathbf{X}}[k]\hat{\mathbf{X}}^{\top}[k]$ with the true moments. The results, shown in the first two columns of Figure 4.6, indicate that the embeddings closely follow a mixture of multivariate Gaussian distributions and that, as expected, the accuracy of $\hat{\mathbf{M}}[k]$ degrades as the moment order increases.

This observation is further reinforced in a second experiment presented in the third and fourth columns of Figure 4.6, where the number of nodes (i.e., the sample

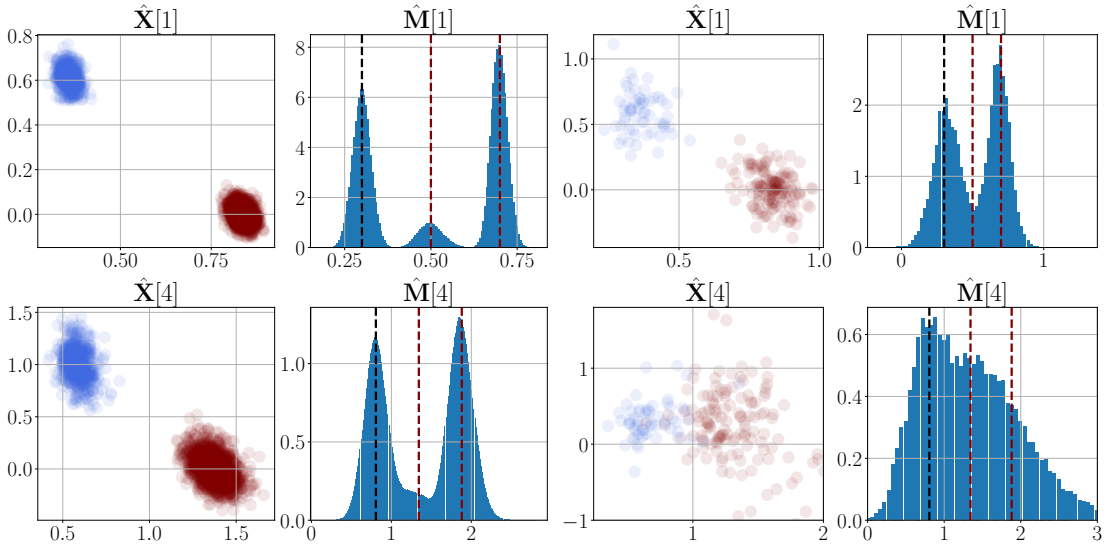

Figure 4.6: Inference results for a two-class SBM with Gaussian weights and $N = 2000$ nodes (first and second columns) and $N = 200$ nodes (third and fourth columns). The plots on the second and fourth columns show histograms of the estimated $\hat{\mathbf{M}}[k]$ and the vertical lines indicate the true moments. For $N = 2000$ embeddings and moments are accurately estimated up to order $k = 4$, while accuracy degrades in the $N = 200$ setting. Also, for fixed sample size performance degrades as the order increases from $k = 1$ (top row) to $k = 4$ (bottom row).

size) is reduced to $N = 200$. Although the embeddings corresponding to $k = 1$ remain fairly accurate, the limited sample size noticeably impairs the estimation for $k = 4$, illustrating the practical limitations imposed by high-order moment estimation. These examples underscore the importance of accounting for finite-sample effects when working with moment-based estimators in network models such as WRDPG.

4.4 Asymptotic results

Recall the ASE estimator introduced in Section 4.3.2. Here we establish the asymptotic results that, for a fixed index k, characterize the behavior of the estimated latent positions $\hat{\mathbf{X}}[k]$ when the number of nodes N goes to infinity. Given the inherent rotational ambiguity in the WRDPG model, the latent position sequence is estimable up to an unknown orthogonal transformation. Thus, our main results (consistency and asymptotic normality) are stated in terms of a sequence of orthogonal matrices $\mathbf{Q}_k \in O(d)$.

For these results to hold, we make the following two assumptions. Here, F is a weighted inner-product distribution as in Definition 4.3.2, $\mathbf{X}_k := \{\mathbf{X}[k]\}_{k\geq 0}$ is a sequence of latent positions matrices, and $\mathbf{W}$ is the adjacency matrix of a WRDPG graph, i.e., $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$ as in Definition 4.3.3.

Assumption 2. Let $\{\mathbf{x}[k]\}_{k\geq 0} \sim F$. Then for each k, the second moment matrix $\mathbf{\Delta}_k = \mathbb{E}\left[\mathbf{x}[k]\mathbf{x}^\top[k]\right]$ has full rank d.

Remark 4.4.1. As mentioned in Chapter 3, Assumption 2 is standard in the RDPG literature (cf. Assumption 1) and is necessary to ensure a separation between the top d eigenvectors and the trailing ones of an observed adjacency matrix following any RDPG model. See Appendix 4.A for a more formal discussion of this topic within the WRDPG model setup.

Assumption 3 (Sub-Weibull weights)**.** There exists a constant $\theta > 0$ such that for each pair $1 \leq i < j \leq N$, there is a constant $C_{ij} > 0$ satisfying:

$$\mathbb{P}\left\{|W_{ij}| \geq t|\mathbf{X}_k\right\} \leq 2\exp\left(-\left(\frac{t}{C_{ij}}\right)^{1/\theta}\right) \text{ for all } t \geq 0.$$

Remark 4.4.2. Since the class of sub-Weibull random variables (rvs) is closed under multiplication (Vladimirova et al., 2020, Proposition 2.3), it follows that W_{ij}^k is also sub-Weibull, with parameter $k\theta$. This will allow us to establish our asymptotic results by invoking a concentration bound for sums of sub-Weibull variables, which is proven in Proposition 4.4.2.

4.4.1 Asymptotic consistency

We first establish the asymptotic consistency of the estimated latent positions (up to an unknown orthogonal transformation).

Theorem 4.4.1. *Let F be a weighted inner-product distribution satisfying Assumption 2 and consider $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$ satisfying Assumption 3. Then, for each index k there exists a matrix $\mathbf{Q}_k \in O(d)$ such that*

$$\left\|\hat{\mathbf{X}}[k]\mathbf{Q}_k - \mathbf{X}[k]\right\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right).$$

Theorem 4.4.1 generalizes a previous consistency result for the WRDPG model (Marenco et al., 2022, Theorem 1) in two key aspects. First, it extends the result to accommodate unbounded edge weights. Second, it provides a bound on the difference between the estimated and actual latent positions (up to an unknown orthogonal transformation) in terms of the $2 \to \infty$ norm rather than the

Frobenius norm. As noted in Cape et al. (2019), the $2 \to \infty$ norm offers tighter control over the estimation error. This can be seen from the identity

$$\|\mathbf{A}\|_{2\to\infty} = \max_{1\le i\le n} \left\|\mathbf{a}_i^\top\right\|_2,$$

where $\mathbf{a}_i^\top$ denotes the i-th row of $\mathbf{A} \in \mathbb{R}^{n\times m}$. This means that the $2 \to \infty$ norm corresponds to the maximum Euclidean norm of the rows of $\mathbf{A}$, so Theorem 4.4.1 bounds the maximum error between the estimated and true latent vectors for *any* node in a WRDPG graph. In contrast, controlling the Frobenius norm does not guarantee such a uniform bound.

Our consistency result aligns with previous findings for the vanilla RDPG model (see Athreya et al., 2017; Lyzinski et al., 2017), in that we provide a bound for the $2 \to \infty$ norm. It also generalizes the consistency result in Gallagher et al. (2023), as their theorem applies only to $\hat{\mathbf{X}}[1]$, meaning they establish consistency solely for the estimated latent positions associated with the mean. Nevertheless, our proof follows their methodology, adapting the necessary arguments to accommodate our more general setup. We note that their proof scheme closely follows that of the RDPG model, as first established in Lyzinski et al. (Theorem 15 of 2017). Additionally, it is worth mentioning that a more general result on matrix perturbations can be obtained using similar techniques; see Cape et al. (2019).

As in the unweighted case, the proof of Theorem 4.4.1 relies on expressing the difference between $\hat{\mathbf{X}}[k]\mathbf{Q}_k$ and $\mathbf{X}[k]$ as the sum of a dominant term and a series of remainder terms. Proposition 4.4.8 shows that these remainders are of a lower order than the dominant term. Its proof will be based on some technical results, which are stated and proven next.

We begin with a lemma that establishes a bound on the moment generating function of a sub-Weibull random variable near the origin.

Lemma 4.4.1. *Let Y be a sub-Weibull random variable with parameter $\theta > 1$; that is, there exists a constant $C > 0$ such that*

$$\mathbb{P}\{|Y| \ge t\} \le 2\exp\left(-\left(\frac{t}{C}\right)^{1/\theta}\right) \quad \textit{for all } t \ge 0.$$

Then there exist $T > 1$ and $C_1 > 0$, depending only on θ and C, such that:

$$\mathbb{E}\left(e^{\lambda|Y|}\right) \le \exp\left(-C_1\lambda^{1/(1-\theta)}\right) \textit{ for all } \lambda \in [0, \lambda_0],$$

where

$$\lambda_0 = \frac{1}{\theta C^{1/\theta}} T^{(1-\theta)/\theta}.$$

Proof. By Fubini's theorem, we can write:

$$\mathbb{E}\left(e^{\lambda|Y|}\right) = \int_0^\infty \lambda e^{\lambda t}\mathbb{P}\left\{|Y| \geq t\right\} dt.$$

Using the sub-Weibull tail bound, we obtain:

$$\mathbb{E}\left(e^{\lambda|Y|}\right) \leq 2\lambda \int_0^\infty \exp\left(\lambda t - \left(\frac{t}{C}\right)^{1/\theta}\right) dt.$$

Let $\phi_\lambda(t) = \lambda t - \left(\frac{t}{C}\right)^{1/\theta}$, and define:

$$I(\lambda) = \int_0^\infty \exp\left(\phi_\lambda(t)\right) dt.$$

The function $\phi_\lambda(t)$ attains its unique maximum at

$$t_\lambda = \left(\lambda\theta C^{1/\theta}\right)^{\theta/(1-\theta)}.$$

To apply Laplace's method (see De Bruijn (2014, Chapter 4)), we require $t_\lambda \geq T$ for some large enough $T > 1$, so that the integrand is sharply peaked near t_λ. Solving $t_\lambda \geq T$ yields:

$$\lambda \leq \lambda_0 = \frac{1}{\theta C^{1/\theta}} T^{(1-\theta)/\theta}.$$

Under this condition, Laplace's method gives:

$$I(\lambda) \leq K \exp\left(\phi_\lambda(t_\lambda)\right),$$

for some constant $K > 0$ depending only on θ and C. Evaluating $\phi_\lambda(t_\lambda)$, we find:

$$\phi_\lambda(t_\lambda) = \left(\frac{1}{\theta} - 1\right)(\theta C)^{1/(1-\theta)}\lambda^{1/(1-\theta)}.$$

Since $\theta > 1$, we define

$$C_1 = -\left(\frac{1}{\theta} - 1\right)(\theta C)^{1/(1-\theta)} > 0,$$

so that

$$\phi_\lambda(t_\lambda) = -C_1\lambda^{1/(1-\theta)}.$$

Putting everything together,

$$\mathbb{E}\left(e^{\lambda|Y|}\right) \leq 2\lambda I(\lambda) \leq 2\lambda K \exp\left(-C_1\lambda^{1/(1-\theta)}\right).$$

Since $\lambda_0 \to 0$ as $T \to \infty$, we can choose a T large enough so that $2\lambda_0 K \leq 1$. Therefore, for all $\lambda \in [0, \lambda_0]$ we have:

$$\mathbb{E}\left(e^{\lambda|Y|}\right) \leq 2\lambda_0 K \exp\left(-C_1\lambda^{1/(1-\theta)}\right) \leq \exp\left(-C_1\lambda^{1/(1-\theta)}\right),$$

as claimed. □

We next present a concentration result for the sum of independent, centered sub-Weibull rvs with a common parameter θ. The proof follows standard arguments used in analogous concentration results for sub-Gaussian and sub-exponential rvs (see Theorems 2.6.3 and 2.8.1 in Vershynin (2018)).

Proposition 4.4.2. *Let $Y_1, \ldots, Y_N$ be independent, mean-zero, sub-Weibull random variables with parameter $\theta > 1$. That is, for each $i = 1, \ldots, N$, there exists a constant $C_i > 0$ such that*

$$\mathbb{P}\{|Y_i| \geq t\} \leq 2\exp\left(-\left(\frac{t}{C_i}\right)^{1/\theta}\right) \quad \text{for all } t \geq 0.$$

For any $T > 0$, define $\lambda_{\min} = \frac{1}{\theta C_{\max}^{1/\theta}} T^{(1-\theta)/\theta}$, where $C_{\max} = \max_{i=1,\ldots,N} C_i$.

Then there exists a large enough T such that for all $t \geq 0$, it holds that

$$\mathbb{P}\left\{\left|\sum_{i=1}^N Y_i\right| \geq t\right\} \leq 2\exp\left(-\min\left\{\lambda_{\min} t + NK_1\lambda_{\min}^{1/(1-\theta)}, \left(\frac{t}{K_2}\right)^{1/\theta} N^{(\theta-1)/\theta}\right\}\right),$$

where $K_1, K_2 > 0$ are constants depending only on θ.

Proof. Let $S := \sum_{i=1}^N Y_i$. By Markov's inequality and independence:

$$\mathbb{P}\{S \geq t\} = \mathbb{P}\left\{e^{\lambda S} \geq e^{\lambda t}\right\} \leq e^{-\lambda t} \prod_{i=1}^N \mathbb{E}\left(e^{\lambda Y_i}\right).$$

Now, using Lemma 4.4.1, there exists a large enough $T > 1$ such that for every $i = 1, \ldots, N$ there exists a constant $C_{1,i} > 0$ (depending only on C_i and θ) such that for all $\lambda \in [0, \lambda_{\min}]$:

$$\mathbb{E}\left(e^{\lambda|Y_i|}\right) \leq \exp\left(-C_{1,i}\lambda^{1/(1-\theta)}\right).$$

Using the fact that Y_i is mean-zero, we have:

$$\mathbb{E}\left(e^{\lambda Y_i}\right) \leq \mathbb{E}\left(e^{\lambda|Y_i|}\right) \leq \exp\left(-C_{1,i}\lambda^{1/(1-\theta)}\right).$$

Hence,

$$\mathbb{P}\{S \geq t\} \leq \exp\left(-\lambda t - \sum_{i=1}^N C_{1,i}\lambda^{1/(1-\theta)}\right).$$

Let $K_1 = \min_i C_{1,i}$. Then K_1 depends only on θ, since the $C_{1,i}$'s depend only on θ. Therefore:

$$\mathbb{P}\{S \geq t\} \leq \exp\left(-\lambda t - NK_1\lambda^{1/(1-\theta)}\right). \tag{4.5}$$

Minimizing the exponent over $\lambda \in [0, \lambda_{\min}]$, the optimal choice is:

$$\lambda^* = \min\left\{\lambda_{\min}, \left(\frac{t(\theta-1)}{NK_1}\right)^{(1-\theta)/\theta}\right\}.$$

Substituting into (4.5), we obtain:

$$\mathbb{P}\{S \geq t\} \leq \exp\left(-\min\left\{\lambda_{\min}t + NK_1\lambda_{\min}^{1/(1-\theta)}, \left(\frac{t}{K_2}\right)^{1/\theta} N^{(\theta-1)/\theta}\right\}\right),$$

where $K_2 = \frac{\theta^\theta}{(\theta-1)^{\theta-1}}(K_1^{(\theta-1)/\theta})$ depends only on θ.

A symmetric argument for $-S$ gives the same bound for $\mathbb{P}\{-S \geq t\}$, and the result follows. □

We next state a lemma that establishes a bound on the difference between the matrix of entry-wise self-products, $\mathbf{W}^{(k)}$, and the true moments matrix, $\mathbf{X}[k]\mathbf{X}^\top[k]$. The proof follows the general scheme of Gallagher et al. (Proposition 1 from 2023), but differs in a key aspect: we rely on a different result for tail bounds of sums of independent matrices. This is due to the fact that the concentration result they invoke does not hold in our more general setup.

Lemma 4.4.3. *Let* $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$ *satisfy the hypotheses of Theorem 4.4.1. For each fixed integer* $k \geq 0$*, let* $\mathbf{W}_k := \mathbf{W}^{(k)}$ *and* $\mathbf{M}_k := \mathbf{X}[k]\mathbf{X}^\top[k]$*. Then*

$$\|\mathbf{W}_k - \mathbf{M}_k\|_2 = \mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right).$$

Proof. Note that the definition of our model implies that $\mathbb{E}[\mathbf{W}_k] = \mathbf{M}_k$, so we are trying to control the spectral norm of the centered matrix $\mathbf{W}_k - \mathbf{M}_k$. To that end, we will use a tail bound for sums of independent matrices (Tropp, 2012, Corollary 3.7). That result can be stated as follows: assume we have a finite sequence $\{\mathbf{Y}_l\}_{l=1}^R$ of independent, self-adjoint, random $N \times N$ matrices that satisfy

$$\mathbb{E}\left(e^{\lambda \mathbf{Y}_l}\right) \preccurlyeq e^{g(\lambda)\mathbf{A}_l} \quad \text{for all } \lambda \in [0, \lambda_0]$$

for some function $g : [0, \lambda_0] \to \mathbb{R}$ and a finite sequence $\{\mathbf{A}_l\}_{l=1}^R$ of fixed self-adjoint matrices. Here, $\succcurlyeq$ denotes the semidefinite ordering on Hermitian matrices, i.e.,

$\mathbf{A} \succcurlyeq \mathbf{B}$ if $\mathbf{A} - \mathbf{B}$ is positive semi-definite. Define the scale parameter

$$\rho = \left\| \sum_{l=1}^{R} \mathbf{A}_l \right\|_2 .$$

Then for all $t \geq 0$ it holds:

$$\mathbb{P}\left\{ \left\| \sum_{l=1}^{R} \mathbf{Y}_l \right\|_2 \geq t \right\} \leq N \inf_{\lambda \in [0, \lambda_0]} \exp\left(-\lambda t + g(\lambda)\rho\right).$$

In order to use this result, for each $1 \leq j \leq i \leq N$ let $\mathbf{Y}_{ij}$ be the $N \times N$ matrix whose (i,j) and (j,i) entries equal $W_{ij}^k - \mathbf{x}_i^\top[k]\mathbf{x}_j[k] := Y_{ij}$, with the rest being 0. Then, $\sum_{ij} \mathbf{Y}_{ij} = \mathbf{W}_k - \mathbf{M}_k$ and $\mathbb{E}\left[\mathbf{Y}_{ij}\right] = \mathbf{0}$ since $\mathbb{E}\left[Y_{ij}\right] = 0$. Also note that if $\mathbf{e}_i$ is the i-th vector of the canonical basis of $\mathbb{R}^N$, we have:

$$\mathbf{Y}_{ij} = Y_{ij}(\mathbf{e}_i\mathbf{e}_j^\top + \mathbf{e}_j\mathbf{e}_i^\top) \Rightarrow \mathbf{Y}_{ij}^p = \begin{cases} Y_{ij}^p(\mathbf{e}_i\mathbf{e}_j^\top + \mathbf{e}_j\mathbf{e}_i^\top) \text{ if } p \text{ is odd} \\ Y_{ij}^p(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top) \text{ if } p \text{ is even, } p \geq 2 \end{cases} .$$

Therefore:

$$\begin{aligned} e^{\lambda \mathbf{Y}_{ij}} &= \mathbf{I} + \sum_{p \text{ odd}} \frac{(\lambda Y_{ij})^p}{p!}(\mathbf{e}_i\mathbf{e}_j^\top + \mathbf{e}_j\mathbf{e}_i^\top) + \sum_{p \text{ even}, p \geq 2} \frac{(\lambda Y_{ij})^p}{p!}(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top) \\ &= \mathbf{I} + \sinh(\lambda Y_{ij})(\mathbf{e}_i\mathbf{e}_j^\top + \mathbf{e}_j\mathbf{e}_i^\top) + (\cosh(\lambda Y_{ij}) - 1)(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top). \end{aligned}$$

This means that $e^{\lambda \mathbf{Y}_{ij}}$ has ones along the diagonal, except at the (i,i) and (j,j) entries, which are equal to $\cosh(\lambda Y_{ij})$. Its off-diagonal entries are zero, except for the (i,j) and (j,i) entries, which are equal to $\sinh(\lambda Y_{ij})$. Using the identity $e^{|x|} - \cosh(x) = |\sinh(x)|$ for all $x \in \mathbb{R}$, together with the Gershgorin circle theorem, we obtain:

$$e^{\lambda \mathbf{Y}_{ij}} \preccurlyeq \mathbf{I} + (e^{|\lambda Y_{ij}|} - 1)(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top) = \exp\left(|\lambda Y_{ij}|(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top)\right),$$

where the last equality follows from the fact that the matrix $\mathbf{I} + \left(e^{|\lambda Y_{ij}|} - 1\right)(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top)$ is diagonal, with all entries equal to 1 except at the (i,i) and (j,j) positions, which are equal to $e^{|\lambda Y_{ij}|}$. Since the matrix expectation preserves the semidefinite order, this in turn implies

$$\mathbb{E}\left(e^{\lambda \mathbf{Y}_{ij}}\right) \preccurlyeq \mathbb{E}\left[\exp\left(|\lambda Y_{ij}|(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top)\right)\right].$$

As discussed in Remark 4.4.2, Assumption 3 implies that Y_{ij} is a sub-Weibull random variable with parameter $k\theta$. Therefore, by Lemma 4.4.1, there exists a constant $C_1^{i,j}$

and a threshold $\lambda_0^{i,j} > 0$ such that for all $\lambda \in [0, \lambda_0^{i,j}]$,

$$\mathbb{E}\left(e^{\lambda \mathbf{Y}_{ij}}\right) \preccurlyeq \mathbb{E}\left[\exp\left(-C_1^{i,j}\lambda^{1/(1-k\theta)}(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top)\right)\right].$$

Letting $K_1 = \min_{i,j} C_1^{i,j}$ and $\lambda_{\min} = \min_{i,j} \lambda_0^{i,j}$, we obtain that for all $\lambda \in [0, \lambda_{\min}]$

$$\mathbb{E}\left(e^{\lambda \mathbf{Y}_{ij}}\right) \preccurlyeq \mathbb{E}\left[\exp\left(-K_1\lambda^{1/(1-k\theta)}(\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top)\right)\right].$$

Thus, we can apply the tail bound for sums of independent random matrices stated above, with $g(\lambda) = -K_1\lambda^{1/(1-k\theta)}$ and $\mathbf{A}_{ij} = \mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top$. Therefore,

$$\mathbb{P}\left\{\|\mathbf{W}_k - \mathbf{M}_k\|_2 \geq t\right\} \leq N \inf_{\lambda \in [0, \lambda_{\min}]} \exp\left(-\lambda t - K_1\lambda^{1/(1-k\theta)}\rho\right), \tag{4.6}$$

where

$$\rho = \left\| \sum_{1 \leq i \leq j \leq N} (\mathbf{e}_i\mathbf{e}_i^\top + \mathbf{e}_j\mathbf{e}_j^\top) \right\|_2 = \|N\mathbf{I}\|_2 = N.$$

Note that the function inside the infimum in (4.6) is equivalent to that on the right-hand side of (4.5). Therefore, an argument analogous to the one in the proof of Proposition 4.4.2 yields

$$\mathbb{P}\left\{\|\mathbf{W}_k - \mathbf{M}_k\|_2 \geq t\right\} \leq N \exp\left(-\min\left\{\lambda_{\min} t + NK_1\lambda_{\min}^{1/(1-k\theta)}, \left(\frac{t}{K_2}\right)^{1/(k\theta)} N^{(k\theta-1)/(k\theta)}\right\}\right),$$

where $K_2 = \frac{(k\theta)^{k\theta}}{(k\theta-1)^{k\theta-1}}(K_1^{(k\theta-1)/(k\theta)})$. Since for large t the term involving $t^{1/(k\theta)}$ dominates the minimum, we have that, for large enough t,

$$\mathbb{P}\left\{\|\mathbf{W}_k - \mathbf{M}_k\|_2 \geq t\right\} \leq N\exp\left(-\left(\frac{t}{K_2}\right)^{1/(k\theta)} N^{(k\theta-1)/(k\theta)}\right).$$

Choosing $t = K_2 \log^{k\theta} N$ then yields the desired result.

□

The following propositions can be proven with the same arguments found in Lyzinski et al. (Proposition 16 and Lemma 17 from 2017) and Gallagher et al. (Propositions 3 and 5 from 2023), respectively. While we omit the detailed proofs, we provide a brief outline of the key ideas underlying each result.

Proposition 4.4.4 follows from a Procrustes-alignment style argument, which leverages the relationship between the principal angles of the subspaces spanned by the columns of $\mathbf{U}_k$ and $\hat{\mathbf{U}}_k$, and the eigenvalues of the difference $\hat{\mathbf{U}}_k\hat{\mathbf{U}}_k^\top - \mathbf{U}_k\mathbf{U}_k^\top$,

together with an application of the Davis–Kahan theorem (Yu et al., 2015). Proposition 4.4.6 can be proved using similar techniques.

Proposition 4.4.5 is derived via a concentration argument analogous to that used in the proof of Theorem 4.4.1. Lastly, Proposition 4.4.7 is a direct consequence of Propositions 4.4.4 and 4.4.6.

Proposition 4.4.4. *For each k, let $\mathbf{M}_k := \mathbf{X}[k]\mathbf{X}^\top[k]$ and denote its spectral decomposition as $\mathbf{M}_k = \mathbf{U}_k\mathbf{D}_k\mathbf{U}_k^\top$. Also let $\mathbf{W}_k := \mathbf{W}^{(k)}$, and define $\hat{\mathbf{D}}_k \in \mathbb{R}^{d\times d}$ as the diagonal matrix of the d largest-magnitude eigenvalues of $\mathbf{W}_k$, with $\hat{\mathbf{U}}_k \in \mathbb{R}^{N\times d}$ containing the corresponding eigenvectors as columns. For each k, let*

$$\mathbf{U}_k^\top\hat{\mathbf{U}}_k = \mathbf{S}_{k_1}\mathbf{\Sigma}_k\mathbf{S}_{k_2}^\top$$

be the singular value decomposition of $\mathbf{U}_k^\top\hat{\mathbf{U}}_k$. If $\mathbf{S}_k := \mathbf{S}_{k_1}\mathbf{S}_{k_2}^\top$, then $\mathbf{S}_k \in O(d)$ and it holds that

$$\left\|\mathbf{U}_k^\top\hat{\mathbf{U}}_k - \mathbf{S}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta} N\right).$$

Proposition 4.4.5. *For each k, let $\mathbf{M}_k, \mathbf{W}_k$ and $\mathbf{U}_k$ be as in Proposition 4.4.4. Then the following holds:*

$$\left\|\mathbf{U}_k^\top(\mathbf{W}_k - \mathbf{M}_k)\mathbf{U}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right).$$

Proposition 4.4.6. *For each k, let $\mathbf{U}_k$ be as in Proposition 4.4.4. Then the following holds:*

$$\left\|\hat{\mathbf{U}}_k\hat{\mathbf{U}}_k^\top - \mathbf{U}_k\mathbf{U}_k^\top\right\|_F = \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta} N\right)$$
$$\left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{U}_k^\top\hat{\mathbf{U}}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta} N\right)$$

Proposition 4.4.7. *For each k, let $\mathbf{D}_h, \hat{\mathbf{D}}_k$ and $\mathbf{S}_k$ be as in Proposition 4.4.4. Then the following holds:*

$$\left\|\mathbf{S}_k\hat{\mathbf{D}}_k - \mathbf{D}_k\mathbf{S}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right),$$
$$\left\|\mathbf{S}_k\hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2}\mathbf{S}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right)$$
$$\left\|\mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2}\mathbf{S}_k\right\|_F = \mathcal{O}_{\mathbb{P}}\left(N^{-3/2}\log^{k\theta} N\right).$$

We now state and prove the proposition that characterizes the behavior of the remainders that will appear in the proof of Theorem 4.4.1.

Proposition 4.4.8. *For each fixed integer $k \geq 0$, let $\mathbf{W}_k := \mathbf{W}^{(k)}$. Also let $\mathbf{M}_k := \mathbf{X}[k]\mathbf{X}^\top[k]$ and denote its spectral decomposition as $\mathbf{M}_k = \mathbf{U}_k\mathbf{D}_k\mathbf{U}_k^\top$. Let $\mathbf{S}_k \in O(d)$*

be as in Proposition 4.4.4 and define $\mathbf{R}_{k_1}$, $\mathbf{R}_{k_2}$, $\mathbf{R}_{k_3}$ *and* $\mathbf{R}_{k_4}$ *as*

$$\begin{aligned}
\mathbf{R}_{k_1} &= \mathbf{U}_k \left(\mathbf{U}_k^\top \hat{\mathbf{U}}_k \hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2} \mathbf{S}_k \right), \\
\mathbf{R}_{k_2} &= \left(\mathbf{I} - \mathbf{U}_k \mathbf{U}_k^\top \right) \left(\mathbf{W}_k - \mathbf{M}_k \right) \left(\hat{\mathbf{U}}_k - \mathbf{U}_k \mathbf{S}_k \right) \hat{\mathbf{D}}_k^{-1/2}, \\
\mathbf{R}_{k_3} &= -\mathbf{U}_k \mathbf{U}_k^\top \left(\mathbf{W}_k - \mathbf{M}_k \right) \mathbf{U}_k \mathbf{S}_k \hat{\mathbf{D}}_k^{-1/2}, \\
\mathbf{R}_{k_4} &= \left(\mathbf{W}_k - \mathbf{M}_k \right) \mathbf{U}_k \left(\mathbf{S}_k \hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2} \mathbf{S}_k \right).
\end{aligned}$$

Then the following holds:

$$\begin{aligned}
\left\| \mathbf{R}_{k_1} \right\|_{2 \to \infty} &= \mathcal{O}_{\mathbb{P}} \left(N^{-1} \log^{k\theta} N \right), \\
\left\| \mathbf{R}_{k_2} \right\|_{2 \to \infty} &= \mathcal{O}_{\mathbb{P}} \left(N^{-3/2} \log^{2k\theta} N \right), \\
\left\| \mathbf{R}_{k_3} \right\|_{2 \to \infty} &= \mathcal{O}_{\mathbb{P}} \left(N^{-1} \log^{k\theta} N \right), \\
\left\| \mathbf{R}_{k_4} \right\|_{2 \to \infty} &= \mathcal{O}_{\mathbb{P}} \left(N^{-3/2} \log^{2k\theta} N \right).
\end{aligned}$$

Proof. In order to bound $\mathbf{R}_{k_1} = \mathbf{U}_k \left(\mathbf{U}_k^\top \hat{\mathbf{U}}_k \hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2} \mathbf{S}_k \right)$, note that:

$$\begin{aligned}
\left\| \mathbf{R}_{k_1} \right\|_{2 \to \infty} &\leq \left\| \mathbf{U}_k \right\|_{2 \to \infty} \left\| \mathbf{U}_k^\top \hat{\mathbf{U}}_k \hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2} \mathbf{S}_k \right\|_2 \\
&\leq \left\| \mathbf{U}_k \right\|_{2 \to \infty} \left(\left\| \left(\mathbf{U}_k^\top \hat{\mathbf{U}}_k - \mathbf{S}_k \right) \hat{\mathbf{D}}_k^{1/2} \right\|_{\mathrm{F}} + \left\| \mathbf{S}_k \hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2} \mathbf{S}_k \right\|_{\mathrm{F}} \right).
\end{aligned}$$

Since $\mathbf{U}_k \mathbf{D}_k^{1/2} = \mathbf{X}[k] \mathbf{Q}_k$ with $\mathbf{Q}_k \in O(d)$, we have that

$$\left\| \mathbf{U}_k \right\|_{2 \to \infty} \leq \left\| \mathbf{X}[k] \right\|_{2 \to \infty} \left\| \mathbf{D}_k^{-1/2} \right\|_2.$$

Since the inner product between the rows of $\mathbf{X}[k]$ equals the k-th moment of some random variable–which, by the definition of our model, we assume to be finite for all k–this implies that all rows of $\mathbf{X}[k]$ (which lie in $\mathbb{R}^d$) must have bounded norm. Therefore, $\left\| \mathbf{X}[k] \right\|_{2 \to \infty}$ is finite and does not depend on N. In Lemma 4.A.1 of Appendix 4.A, we show that Assumption 2 implies all nonzero eigenvalues of $\mathbf{M}_k$ are $\Theta_{\mathbb{P}}(N)$, so $\left\| \mathbf{D}_k^{-1/2} \right\|_2 = \mathcal{O}_{\mathbb{P}} \left(N^{-1/2} \right)$. Consequently, $\left\| \mathbf{U}_k \right\|_{2 \to \infty} = \mathcal{O}_{\mathbb{P}} \left(N^{-1/2} \right)$. Combining this with Propositions 4.4.4 and 4.4.7 yields the desired bound for $\mathbf{R}_{k_1}$.

In a similar vein, for $\mathbf{R}_{k_3} = -\mathbf{U}_k \mathbf{U}_k^\top \left(\mathbf{W}_k - \mathbf{M}_k \right) \mathbf{U}_k \mathbf{S}_k \hat{\mathbf{D}}_k^{-1/2}$ it holds:

$$\begin{aligned}
\left\| \mathbf{R}_{k_3} \right\|_{2 \to \infty} &\leq \left\| \mathbf{U}_k \right\|_{2 \to \infty} \left\| \mathbf{U}_k^\top \left(\mathbf{W}_k - \mathbf{M}_k \right) \mathbf{U}_k \mathbf{S}_k \hat{\mathbf{D}}_k^{-1/2} \right\|_2 \\
&\leq \left\| \mathbf{U}_k \right\|_{2 \to \infty} \left\| \mathbf{U}_k^\top \left(\mathbf{W}_k - \mathbf{M}_k \right) \mathbf{U}_k \right\|_{\mathrm{F}} \left\| \hat{\mathbf{D}}_k^{-1/2} \right\|_2.
\end{aligned}$$

Using Proposition 4.4.5 and the fact that both $\left\| \mathbf{U}_k \right\|_{2 \to \infty}$ and $\left\| \hat{\mathbf{D}}_k \right\|_2$ are $\mathcal{O}_{\mathbb{P}} \left(N^{-1/2} \right)$ (the latter being a consequence of Assumptions 2 and 3, as shown in Lemma 4.A.2

of Appendix 4.A) then implies that $\|\mathbf{R}_{k_3}\|_{2\to\infty}$ is $\mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}N\right)$, as desired.

Regarding $\mathbf{R}_{k_4} = (\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k\left(\mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2}\mathbf{S}_k\right)$ we have that:

$$\begin{aligned}\|\mathbf{R}_{k_4}\|_{2\to\infty} &\leq \|\mathbf{W}_k - \mathbf{M}_k\|_{\infty}\left\|\mathbf{U}_k\left(\mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2}\mathbf{S}_k\right)\right\|_{2\to\infty}\\ &\leq \sqrt{N}\|\mathbf{W}_k - \mathbf{M}_k\|_2\|\mathbf{U}_k\|_{2\to\infty}\left\|\mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2}\mathbf{S}_k\right\|_2,\end{aligned}$$

where in the second inequality we have used that $\|\mathbf{A}\|_{\infty} \leq \sqrt{n}\|\mathbf{A}\|_2$ for all $\mathbf{A} \in \mathbb{R}^{n\times m}$. Using Lemma 4.4.3 and Proposition 4.4.7 together with the fact that $\|\mathbf{U}_k\|_{2\to\infty}$ is $\mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$ then shows the desired bound for $\mathbf{R}_{k_4}$.

This leaves us with the task of bounding the $2\to\infty$ norm of $\mathbf{R}_{k_2}$, which we remind the reader is defined as:

$$\mathbf{R}_{k_2} = \left(\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^{\top}\right)(\mathbf{W}_k - \mathbf{M}_k)\left(\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{S}_k\right)\hat{\mathbf{D}}_k^{-1/2}$$

Therefore:

$$\|\mathbf{R}_{k_2}\|_{2\to\infty} \leq \left\|\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^{\top}\right\|_{2\to\infty}\|\mathbf{W}_k - \mathbf{M}_k\|_2\left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{S}_k\right\|_2\left\|\hat{\mathbf{D}}_k^{-1/2}\right\|_2 \tag{4.7}$$

For the first term we have that:

$$\left\|\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^{\top}\right\|_{2\to\infty} \leq \|\mathbf{I}\|_{2\to\infty} + \left\|\mathbf{U}_k\mathbf{U}_k^{\top}\right\|_{2\to\infty} \leq 1 + \|\mathbf{U}_k\|_{2\to\infty}\left\|\mathbf{U}_k^{\top}\right\|_2.$$

Since $\left\|\mathbf{U}_k^{\top}\right\|_2 = \|\mathbf{U}_k\|_2 \leq \sqrt{N}\|\mathbf{U}_k\|_{2\to\infty}$ we have that $\left\|\mathbf{U}_k^{\top}\right\|_2 = \mathcal{O}_{\mathbb{P}}(1)$ because $\|\mathbf{U}_k\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$. Therefore

$$\left\|\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^{\top}\right\|_{2\to\infty} \leq 1 + \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right) = \mathcal{O}_{\mathbb{P}}(1).$$

Also note that:

$$\begin{aligned}\left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{S}_k\right\|_2 &\leq \left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{U}_k^{\top}\hat{\mathbf{U}}_k\right\|_2 + \left\|\mathbf{U}_k\left(\mathbf{U}_k^{\top}\hat{\mathbf{U}}_k - \mathbf{S}_k\right)\right\|_2\\ &\leq \left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{U}_k^{\top}\hat{\mathbf{U}}_k\right\|_2 + \|\mathbf{U}_k\|_2\left\|\mathbf{U}_k^{\top}\hat{\mathbf{U}}_k - \mathbf{S}_k\right\|_2.\end{aligned}$$

By Proposition 4.4.6 the first term is $\mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}N\right)$, whereas the second term is $\mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}N\right)$ because $\|\mathbf{U}_k\|_2 = \mathcal{O}_{\mathbb{P}}(1)$ and $\left\|\mathbf{U}_k^{\top}\hat{\mathbf{U}}_k - \mathbf{S}_k\right\|_2$ is $\mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}N\right)$ by virtue of Proposition 4.4.4. Therefore:

$$\left\|\hat{\mathbf{U}}_k - \mathbf{U}_k\mathbf{S}_k\right\|_2 = \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}N\right).$$

Combining this with Lemma 4.4.3 and the fact that $\left\|\hat{\mathbf{D}}_k^{-1/2}\right\|_2$ is $\mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$, from (4.7) we conclude $\|\mathbf{R}_{k_2}\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-3/2}\log^{2k\theta}N\right)$, which finalizes the proof.

□

Using this proposition, we are now ready to prove Theorem 4.4.1.

Proof of Theorem 4.4.1. First, assume $\mathbf{X}[k] = \mathbf{U}_k\mathbf{D}_k^{1/2}$. Fix $\mathbf{S}_k \in O(d)$ such that Proposition 4.4.8 holds. Since $\hat{\mathbf{X}}[k] = \hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2}$, we have that:

$$\begin{aligned}\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k =& \hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} - \mathbf{U}_k\mathbf{D}_k^{1/2}\mathbf{S}_k \\ =& \hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} - \mathbf{U}_k\mathbf{U}_k^\top\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} + \mathbf{U}_k\left(\mathbf{U}_k^\top\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} - \mathbf{D}_k^{1/2}\mathbf{S}_k\right) \\ =& \hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} - \mathbf{U}_k\mathbf{U}_k^\top\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} + \mathbf{R}_{k_1},\end{aligned}$$

where $\mathbf{R}_{k_1}$ is defined as in Proposition 4.4.8.

Note that $\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k = \mathbf{W}_k\hat{\mathbf{U}}_k$, which implies that $\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{1/2} = \mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2}$. So:

$$\begin{aligned}\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k =& \mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{U}_k\mathbf{U}_k^\top\mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1} \\ =& \mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{M}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{M}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \\ & - \mathbf{U}_k\mathbf{U}_k^\top\mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1} \\ =& \left(\mathbf{W}_k - \mathbf{M}_k\right)\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{U}_k\mathbf{U}_k^\top\mathbf{M}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} \\ & - \mathbf{U}_k\mathbf{U}_k^\top\mathbf{W}_k\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1} \qquad (4.8) \\ =& \left(\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^\top\right)\left(\mathbf{W}_k - \mathbf{M}_k\right)\hat{\mathbf{U}}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1}, \qquad (4.9)\end{aligned}$$

where in (4.8) we have used the fact that $\mathbf{M}_k = \mathbf{U}_k\mathbf{U}_k^\top\mathbf{M}_k$. From (4.9) we find that:

$$\begin{aligned}\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k =& \left(\mathbf{I} - \mathbf{U}_k\mathbf{U}_k^\top\right)\left(\mathbf{W}_k - \mathbf{M}_k\right)\left(\mathbf{U}_k\mathbf{S}_k - \mathbf{U}_k\mathbf{S}_k + \hat{\mathbf{U}}_k\right)\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1} \\ =& \left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} + \mathbf{R}_{k_1} + \mathbf{R}_{k_2} + \mathbf{R}_{k_3} \\ =& \left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\left(\mathbf{D}_k^{-1/2}\mathbf{S}_k + \mathbf{S}_k\hat{\mathbf{D}}_k^{-1/2} - \mathbf{D}_k^{-1/2}\mathbf{S}_k\right) + \\ & + \mathbf{R}_{k_1} + \mathbf{R}_{k_2} + \mathbf{R}_{k_3} \\ =& \left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{S}_k + \mathbf{R}_{k_1} + \mathbf{R}_{k_2} + \mathbf{R}_{k_3} + \mathbf{R}_{k_4}, \qquad (4.10)\end{aligned}$$

with $\mathbf{R}_{k_2}, \mathbf{R}_{k_3}$ and $\mathbf{R}_{k_4}$ defined as in Proposition 4.4.8. By said proposition, we conclude that

$$\left\|\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k\right\|_{2\to\infty} \leq \left\|\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{S}_k\right\|_{2\to\infty} + \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}(N)\right).$$

Since $\|\mathbf{AB}\|_{2\to\infty} \leq \|\mathbf{A}\|_{2\to\infty}\|\mathbf{B}\|_2$, we have:

$$\left\|\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k\right\|_{2\to\infty} \leq \left\|\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\right\|_{2\to\infty}\left\|\mathbf{D}_k^{-1/2}\right\|_2 + \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}(N)\right)$$

$$\leq \|(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k\|_{2\to\infty} \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right) + \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}(N)\right). \tag{4.11}$$

To bound the term $\|(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k\|_{2\to\infty}$, we first note that each entry of that matrix is a linear combination of centered, sub-Weibull rvs. Indeed,

$$[(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k]_{ij} = \sum_{l=1}^{N} (W_{il} - M_{il})\,U_{lj} = \sum_{l\neq i} (W_{il} - M_{il})\,U_{lj} - M_{ii}U_{ii},$$

where we denote the (i,j) element of $\mathbf{W}_k$, $\mathbf{M}_k$, and $\mathbf{U}_k$ by W_{ij}, M_{ij}, and U_{ij}, respectively. Since the sum in the rightmost term consists of centered, independent sub-Weibull rvs of parameter $k\theta$, using Proposition 4.4.2 we find that, for large enough t:

$$\mathbb{P}\left[\left|[(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k]_{ij}\right| \geq t\right] \leq 2\exp\left(\left(\frac{t}{K_2}\right)^{1/(k\theta)} N^{(k\theta-1)/(k\theta)}\right),$$

for some constant $K_2 > 0$ depending on $k\theta$. By choosing $t = K_2\log^{k\theta} N$ we get that $[(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k]_{ij}$ is $\mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right)$, and by summing over $j = 1,\ldots,d$ we get that each row of $(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k$ is also $\mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right)$, so its $2\to\infty$ norm is of the same order.

Equation (4.11) then implies that

$$\begin{aligned}\left\|\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{S}_k\right\|_{2\to\infty} &= \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right) + \mathcal{O}_{\mathbb{P}}\left(N^{-1}\log^{k\theta}(N)\right)\\ &= \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right).\end{aligned}$$

Choosing $\mathbf{Q}_k = \mathbf{S}_k^\top$ shows that $\left\|\hat{\mathbf{X}}[k]\mathbf{Q}_k - \mathbf{X}[k]\right\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right)$, as desired.

If $\mathbf{X}[k] \neq \mathbf{U}_k\mathbf{D}_k^{1/2}$, we have that $\mathbf{X}[k]\mathbf{T}_k = \mathbf{U}_k\mathbf{D}_k^{1/2}$ for some $\mathbf{T}_k \in O(d)$. Note that the above calculations imply that

$$\left\|\hat{\mathbf{X}}[k] - \mathbf{U}_k\mathbf{D}_k^{1/2}\mathbf{S}_k\right\|_{2\to\infty} = \mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\log^{k\theta} N\right).$$

Therefore, in this case it is enough to choose $\mathbf{Q}_k = \mathbf{S}_k^\top\mathbf{T}_k^\top \in O(d)$, since

$$\left\|\hat{\mathbf{X}}[k]\mathbf{Q}_k - \mathbf{X}[k]\right\|_{2\to\infty} = \left\|(\hat{\mathbf{X}}[k] - \mathbf{X}[k]\mathbf{T}_k\mathbf{S}_k)\mathbf{Q}_k\right\|_{2\to\infty} \leq \left\|\hat{\mathbf{X}}[k] - \mathbf{U}_k\mathbf{D}_k^{1/2}\mathbf{S}_k\right\|_{2\to\infty}.$$

□

4.4.2 Asymptotic Normality

Next, we show that latent positions behave, asymptotically as $N \to \infty$, as multivariate normal random variables. We also explicitly calculate the covariance matrix of such a normal in terms of the second-moment matrix $\mathbf{\Delta}_k$ of the latent positions and the latent positions themselves.

Theorem 4.4.2. *Let $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$ be as in Theorem 4.4.1. For each index k, define the variance function $v_k : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ as*

$$v_k(\mathbf{x}, \mathbf{y}) := \text{var}\left[W_{ij}^k \middle| \mathbf{x}_i[k] = \mathbf{x}, \mathbf{x}_j[k] = \mathbf{y}\right],$$

where W_{ij} is the (i, j) entry of $\mathbf{W}$. Let $\mathbf{\Sigma}_k : \mathbb{R}^d \mapsto \mathbb{R}^{d\times d}$ be the covariance function

$$\mathbf{\Sigma}_k(\mathbf{x}) = \mathbf{\Delta}_k^{-1}\mathbb{E}\left[v_k(\mathbf{x}, \mathbf{y}_k)\mathbf{y}_k\mathbf{y}_k^\top\right]\mathbf{\Delta}_k^{-1},$$

where $\{\mathbf{y}_k\}_{k\geq 0} \sim F$ and $\mathbf{\Delta}_k$ is the second-moment matrix $\mathbf{\Delta}_k = \mathbb{E}\left[\mathbf{y}_k\mathbf{y}_k^\top\right]$.

Then for each k there exists a sequence of orthogonal matrices $\{\mathbf{Q}_{k_N}\}_{N\geq 0}$ such that for all $\mathbf{z} \in \mathbb{R}^d$ and for any fixed row index i,

$$\lim_{N\to\infty} \mathbb{P}\left[N^{1/2}\left(\hat{\mathbf{X}}[k]\mathbf{Q}_{k_N} - \mathbf{X}[k]\right)_i^\top \leq \mathbf{z} \,\middle|\, \mathbf{x}_i[k] = \mathbf{x}\right] = \mathbf{\Phi}(\mathbf{z}; \mathbf{\Sigma}_k(\mathbf{x})),$$

where $\mathbf{\Phi}(\cdot; \mathbf{\Sigma})$ stands for the cumulative distribution function of a $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ random vector.

Proof. From the proof of Theorem 4.4.1, we have that for each fixed k there exists $\mathbf{S}_k, \mathbf{T}_k \in O(d)$ such that for $\mathbf{Q}_k = \mathbf{S}_k^\top\mathbf{T}_k^\top$ it holds

$$\hat{\mathbf{X}}[k]\mathbf{Q}_k - \mathbf{X}[k] = \left(\hat{\mathbf{X}}[k] - \mathbf{U}_k\mathbf{D}_k^{1/2}\mathbf{S}_k\right)\mathbf{Q}_k.$$

Also from that proof [see (4.10)], we have that

$$\hat{\mathbf{X}}[k] - \mathbf{U}_k\mathbf{D}_k^{1/2}\mathbf{S}_k = (\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{S}_k + \mathbf{R}_k,$$

where we have defined $\mathbf{R}_k = \mathbf{R}_{k_1} + \mathbf{R}_{k_2} + \mathbf{R}_{k_3} + \mathbf{R}_{k_4}$. Therefore:

$$N^{1/2}\left(\hat{\mathbf{X}}[k]\mathbf{Q}_k - \mathbf{X}[k]\right) = N^{1/2}\,(\mathbf{W}_k - \mathbf{M}_k)\,\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{T}_k^\top + N^{1/2}\mathbf{R}_k\mathbf{Q}_k. \tag{4.12}$$

Using Proposition 4.4.8, we have that $\left\|N^{1/2}\mathbf{R}_k\mathbf{Q}_k\right\|_{2\to\infty} \to 0$ as $N \to \infty$. Therefore, we can focus on the first summand in (4.12).

Recall from the proof of Theorem 4.4.1 that $\mathbf{T}_k$ is such that $\mathbf{X}[k]\mathbf{T}_k = \mathbf{U}_k\mathbf{D}_k^{1/2}$,

which implies that $\mathbf{U}_k\mathbf{D}_k^{-1/2} = \mathbf{X}[k]\mathbf{T}_k\mathbf{D}_k^{-1}$. Thus:

$$N^{1/2}\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{T}_k^\top = N^{1/2}\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{X}[k]\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k^\top.$$

Therefore, the i-th row of the first summand in (4.12) equals:

$$\begin{aligned}
N^{1/2}\left[\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{U}_k\mathbf{D}_k^{-1/2}\mathbf{T}_k^\top\right]_i^\top &= N^{1/2}\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k^\top\left\{\left(\mathbf{W}_k - \mathbf{M}_k\right)\mathbf{X}[k]\right\}_i^\top \\
&= N\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k^\top\left[N^{-1/2}\sum_{j=1}^{N}\left(W_{ij}^k - M_{ij}\right)\mathbf{x}_j[k]\right] \\
&= N\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k^\top\left[N^{-1/2}\sum_{j\neq i}\left(W_{ij}^k - M_{ij}\right)\mathbf{x}_j[k]\right] \\
&\quad - N\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k\left(N^{-1/2}M_{ii}\mathbf{x}_i[k]\right),
\end{aligned} \tag{4.13}$$

where as before we denote the (i,j) element of $\mathbf{W}$ and $\mathbf{M}_k$ by W_{ij} and M_{ij}, respectively. Conditioning on $\mathbf{x}_i[k] = \mathbf{x}$, we have that for the second term in (4.13)

$$\begin{aligned}
\left\|N\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k\left(N^{-1/2}M_{ii}\mathbf{x}_i[k]\right)\right\|_2 &\leq N^{1/2}|M_{ii}|\left\|\mathbf{T}_k\mathbf{D}_k^{-1}\mathbf{T}_k\right\|_{2\to\infty}\|\mathbf{x}\|_2 \\
&\leq N^{1/2}|M_{ii}|\left\|\mathbf{D}_k^{-1}\right\|_2\|\mathbf{x}\|_2,
\end{aligned}$$

where in the second inequality we have used that $\|\mathbf{A}\|_{2\to\infty} \leq \|\mathbf{A}\|_2$ and $\|\mathbf{T}_k\|_2 = 1$ because $\mathbf{T}_k \in O(d)$. Because $\left\|\mathbf{D}_k^{-1}\right\|_2 = \mathcal{O}_{\mathbb{P}}\left(N^{-1}\right)$, we conclude that the second term in (4.13) is $\mathcal{O}_{\mathbb{P}}\left(N^{-1/2}\right)$.

As for the first term in (4.13), conditional on $\mathbf{x}_i[k] = \mathbf{x}$ we have that $M_{ij} = \mathbf{x}^\top\mathbf{x}_j[k]$, so the terms in the sum

$$N^{-1/2}\sum_{j\neq i}\left(W_{ij}^k - M_{ij}\right)\mathbf{x}_j[k]$$

are centered, independent random variables, whose covariance matrix is

$$\tilde{\mathbf{\Sigma}}_k(\mathbf{x}) = \mathbb{E}\left[v_k(\mathbf{x},\mathbf{y}_k)\mathbf{y}_k\mathbf{y}_k^\top\right],$$

where $\{\mathbf{y}_k\}_{k\geq 0} \sim F$. Therefore, the multivariate central limit theorem implies that

$$N^{-1/2}\sum_{j\neq i}\left(W_{ij} - M_{ij}\right)\mathbf{x}_j[k] \xrightarrow{\mathcal{L}} \mathcal{N}(\mathbf{0}, \tilde{\mathbf{\Sigma}}_k(\mathbf{x})).$$

To conclude the proof, recall that $\mathbf{T}_k \in O(d)$ is such that $\mathbf{X}[k]\mathbf{T}_k = \mathbf{U}_k\mathbf{D}_k^{1/2}$, so

$\mathbf{X}[k] = \mathbf{U}_k \mathbf{D}_k^{1/2} \mathbf{T}_k^\top$ and therefore $\mathbf{X}^\top[k]\mathbf{X}[k] = \mathbf{T}_k \mathbf{D}_k \mathbf{T}_k^\top$. This implies that

$$\left(\mathbf{X}^\top[k]\mathbf{X}[k]\right)^{-1} = \mathbf{T}_k \mathbf{D}_k^{-1} \mathbf{T}_k^\top,$$

and therefore by the law of large numbers we have that almost surely it holds

$$N \mathbf{T}_k \mathbf{D}_k^{-1} \mathbf{T}_k^\top \to \mathbf{\Delta}_k^{-1}.$$

This implies that the first term in (4.13) converges in distribution to a multivariate normal $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}(\mathbf{x}))$, which completes the proof.

□

Remark 4.4.3. The covariance function in Theorem 4.4.2 depends on $v_k(\mathbf{x}, \mathbf{y})$, which is the conditional variance of W_{ij}^k given $\mathbf{x}_i[k] = \mathbf{x}$ and $\mathbf{x}_j[k] = \mathbf{y}$. Therefore, we can rewrite v_k as

$$\begin{aligned} v_k(\mathbf{x}, \mathbf{y}) &= \mathbb{E}\left[(W_{ij}^k)^2 \middle| \mathbf{x}_i[k] = \mathbf{x}, \mathbf{x}_j[k] = \mathbf{y}\right] - \mathbb{E}\left[W_{ij}^k \middle| \mathbf{x}_i[k] = \mathbf{x}, \mathbf{x}_j[k] = \mathbf{y}\right]^2 \\ &= \mathbb{E}\left[W_{ij}^{2k} \middle| \mathbf{x}_i[k] = \mathbf{x}, \mathbf{x}_j[k] = \mathbf{y}\right] - (\mathbf{x}^\top \mathbf{y})^2, \end{aligned}$$

since the WRDPG model by definition imposes $\mathbb{E}\left[W_{ij}^k \middle| \mathbf{x}_i[k] = \mathbf{x}, \mathbf{x}_j[k] = \mathbf{y}\right] = \mathbf{x}^\top \mathbf{y}$. The expectation of W_{ij}^{2k} need not have a closed-form expression given $\mathbf{x}_i[k] = \mathbf{x}$ and $\mathbf{x}_j[k] = \mathbf{y}$. However, if we further condition on the events $\mathbf{x}_i[2k] = \mathbf{x}_2$ and $\mathbf{y}_i[2k] = \mathbf{y}_2$, we have that $\mathbb{E}\left[W_{ij}^{2k}\right] = \mathbf{x}_2^\top \mathbf{y}_2$. Therefore, by conditioning on $\mathbf{x}_i[k] = \mathbf{x}_1$ and $\mathbf{x}_i[2k] = \mathbf{x}_2$ we have the following Corollary of Theorem 4.4.2.

Corollary 4.4.3. *Let $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$ be as in Theorem 4.4.1. Then for each k there exists a sequence of orthogonal matrices $\{\mathbf{Q}_{k_N}\}_{N \geq 0}$ such that for all $\mathbf{z} \in \mathbb{R}^d$ and for any fixed row index i,*

$$\lim_{N \to \infty} \mathbb{P}\left[N^{1/2} \left(\hat{\mathbf{X}}[k] \mathbf{Q}_{k_N} - \mathbf{X}[k] \right)_i^\top \leq \mathbf{z} \,\middle|\, \mathbf{x}_i[k] = \mathbf{x}_1,\ \mathbf{x}_i[2k] = \mathbf{x}_2 \right] = \mathbf{\Phi}(\mathbf{z}, \mathbf{\Sigma}_k(\mathbf{x}_1, \mathbf{x}_2)),$$

where $\mathbf{\Sigma}_k : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}^{d \times d}$ is the covariance function

$$\mathbf{\Sigma}_k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{\Delta}_k^{-1} \mathbb{E}\left[\left(\mathbf{x}_2^\top \mathbf{y}_{2k} - (\mathbf{x}_1^\top \mathbf{y}_k)^2 \right) \mathbf{y}_k \mathbf{y}_k^\top \right] \mathbf{\Delta}_k^{-1},$$

and $\{\mathbf{y}_k\}_{k \geq 0} \sim F$.

As seen in a previous example in this chapter, when the random graph follows a weighted stochastic block model (WSBM) with C classes, for each moment index k there are at most C distinct latent positions that a node can assume. We denote

these positions by $\mathbf{y}_m[k]$, for $m = 1, \ldots, C$. Assume that each node belongs to class m with probability π_m, and define the vector $\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_C]^\top$. Let $\mathbf{B} \in \mathbb{R}^{C \times C}$ be the matrix of edge-formation probabilities between classes–that is, the entry b_{lm} in position (l, m) gives the probability that an edge exists between a node in class l and a node in class m. Let d_{lm} denote the probability distribution governing the edge weights between nodes in communities l and m, with $m_{lm}[k]$ denoting its k-th moment.

Under these conventions, each entry of the weighted adjacency matrix $\mathbf{W}$ is either zero with probability $1 - \boldsymbol{\pi}^\top \mathbf{B} \boldsymbol{\pi}$, or drawn from d_{lm} with probability $\pi_l \pi_m b_{lm}$. In this setup, the following corollary of Theorem 4.4.2 holds:

Corollary 4.4.4. *With the above notation, for each moment index k here exists a sequence of orthogonal matrices $\{\mathbf{Q}_{k_N}\}_{N \geq 0}$ such that for all $\mathbf{z} \in \mathbb{R}^d$ and for any fixed row index i,*

$$\lim_{N \to \infty} \mathbb{P}\left[N^{1/2} \left(\hat{\mathbf{X}}[k] \mathbf{Q}_{k_N} - \mathbf{X}[k] \right)_i^\top \leq \mathbf{z} \,\middle|\, \textit{node } i \textit{ belongs to community } l \right] = \boldsymbol{\Phi}(\mathbf{z}, \boldsymbol{\Sigma}_{kl}),$$

where the covariance matrix $\boldsymbol{\Sigma}_{kl}$ is given by $\boldsymbol{\Sigma}_{kl} = \boldsymbol{\Delta}_k^{-1} \tilde{\boldsymbol{\Sigma}}_{kl} \boldsymbol{\Delta}_k^{-1}$, with

$$\tilde{\boldsymbol{\Sigma}}_{kl} = \sum_{m=1}^{C} \pi_m \left(b_{lm} m_{lm}[2k] - b_{lm}^2 m_{lm}^2[k] \right) \mathbf{y}_m[k] \mathbf{y}_m^\top[k]$$

and

$$\boldsymbol{\Delta}_k = \sum_{m=1}^{C} \pi_m \mathbf{y}_m[k] \mathbf{y}_m^\top[k].$$

4.5 Graph generation

So far, we have formally defined the WRDPG model and demonstrated its versatility and discriminative power through examples. In addition, we have provided statistical guarantees for the proposed ASE-based estimation method. In this section, we investigate how to generate graphs from the WRDPG model with latent positions

$$\mathbf{X}[0], \mathbf{X}[1], \ldots, \mathbf{X}[K] \in \mathbb{R}^{N \times d},$$

where $\mathbf{X}[k] = [\mathbf{x}_1[k], \mathbf{x}_2[k], \ldots, \mathbf{x}_N[k]]^\top$. In other words, our aim is to sample the adjacency matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$, such that, for each pair $1 \leq i < j \leq N$, W_{ij} follows a distribution whose first $K + 1$ moments are given by $\mathbf{x}_i^\top[k] \mathbf{x}_j[k]$, for all $k = 0, 1, \ldots, K$. The latent positions could be defined so as to match a prescribed weight distribution, or, be estimated from real networks via the ASE.

In the sequel, we first consider a discrete weight distribution with finite support. We show that the weights' probability mass function (pmf) can be obtained in closed form, by solving a linear system of equations with Vandermonde structure. As we will see, our method is capable of reproducing the original characteristics of the network. Next, we address the problem of generating samples from WRDPG graphs with continuous weight distributions. To this end, we rely on the maximum entropy principle to recover the probability density function (pdf) from its moments. We develop a new primal-dual method, providing better and more robust solutions than prior art (Saad & Ruai, 2019). Finally, we consider the case of a mixture distribution, simultaneously reproducing the connectivity structure of a real network and its weight distribution. Throughout, we provide supporting examples using synthetic and real data.

4.5.1 Discrete weights distribution

We begin by deriving a solution for the case where W_{ij} has a discrete distribution. Suppose that the latent positions for each node are given, and W_{ij} takes on $R+1$ (known) distinct values $v_0, v_1, \ldots, v_R$ with (unknown) probabilities $p_0, p_1, \ldots, p_R$.[1] To generate a WRDPG-adhering graph, we need to estimate these probabilities from the latent position sequence. In this case, the moments for such a distribution can be computed as

$$\mathbb{E}\left[W_{ij}^k\right] = \sum_{r=0}^{R} v_r^k p_r = v_0^k p_0 + v_1^k p_1 + \cdots + v_R^k p_R = \mathbf{x}_i^\top[k]\mathbf{x}_j[k] := m_{ij}[k],$$

where the usual convention $0^0 = 1$ is used in case $v_r = 0$, for some r. We then obtain the following system of $K+1$ linear equations on the pmf $\mathbf{p} = [p_0, p_1, \ldots, p_R]^\top \in [0,1]^{R+1}$:

$$\left\{\begin{array}{rcl} p_0 + p_1 + \cdots + p_R &=& m_{ij}[0] \\ v_0 p_0 + v_1 p_1 + \cdots + v_R p_R &=& m_{ij}[1] \\ v_0^2 p_0 + v_1^2 p_1 + \cdots + v_R^2 p_R &=& m_{ij}[2] \\ \vdots & & \vdots \\ v_0^K p_0 + v_1^K p_1 + \cdots + v_R^K p_R &=& m_{ij}[K] \end{array}\right. \Leftrightarrow \mathbf{V}\mathbf{p} = \mathbf{m}, \tag{4.14}$$

[1] Note that both v_r's and p_r's are dependent on i, j, but that dependence has been omitted to improve readability.

where $\mathbf{m} = [m_{ij}[0], m_{ij}[1], \ldots, m_{ij}[R]]^\top \in \mathbb{R}^{R+1}$ and $\mathbf{V} \in \mathbb{R}^{(K+1)\times(R+1)}$ is the Vandermonde matrix

$$\mathbf{V} = \begin{pmatrix} 1 & 1 & \ldots & 1 \\ v_0 & v_1 & \ldots & v_R \\ v_0^2 & v_1^2 & \ldots & v_R^2 \\ \vdots & \vdots & \ddots & \vdots \\ v_0^K & v_1^K & \ldots & v_R^K \end{pmatrix}.$$

Note that if $K = R$ then the system (4.14) has a unique solution. So, in order to estimate the pmf associated with edge (i, j) we need to prescribe as many moments as there are possible values for W_{ij}. Given the first R moments of the distribution, we readily obtain the probabilities as $\mathbf{p} = \mathbf{V}^{-1}\mathbf{m}$.

While the linear system (4.14) needs to be solved for every pair (i, j), if the distributions associated with different edges share a common support $v_0, v_1, \ldots, v_R$, then $\mathbf{V}$ (and thus $\mathbf{V}^{-1}$) are the same across those edges, making computation of $\mathbf{p}$ for the entire graph less demanding.

Example 4.5.1. To illustrate how the graph generative process works, we simulated a two-class weighted SBM network with $N = 500$ nodes (350 in community 1 and 150 in community 2), and block probability matrix

$$\mathbf{B} = \begin{pmatrix} 0.7 & 0.2 \\ 0.2 & 0.5 \end{pmatrix}, \tag{4.15}$$

where edge weights are sampled from a discrete distribution supported on the values $1, 2, \ldots, 10$, with $p_i = \frac{1}{18}$ for $i \neq 5$ and $p_5 = \frac{1}{2}$. Using (4.4), we compute the analytical latent positions for each node, where $m_d[k]$ denotes the k-th moment of the aforementioned discrete distribution.

For each edge, we compute the finite moment sequence $m_{ij}[k] = \mathbf{x}_i^\top[k]\mathbf{x}_j[k]$, for $k = 0, \ldots, 10$. In this setup, each edge weight can take on values in $\{0, 1, 2, \ldots, 10\}$, with probabilities p_0, $(1 - p_0)p_1$, $(1 - p_0)p_2$, $\ldots$, $(1 - p_0)p_{10}$, where p_0 is the probability that nodes i and j are not connected. Specifically, $p_0 = 1 - 0.7 = 0.3$ if i and j both belong to community 1, $p_0 = 1 - 0.5 = 0.5$ if both are in community 2, and $p_0 = 1 - 0.2 = 0.8$ otherwise. Thus, edge weights are discrete random variables with $R+1 = 11$ possible values, which requires 11 moments to uniquely solve (4.14).

Figure 4.7 shows the distributions of various metrics (degree, betweenness centrality, geodesic distance) for 100 simulated networks generated using the above procedure, along with the same metrics computed on a single reference network drawn from the base model–namely, a two-block SBM with $\mathbf{B}$ as in (4.15), and edge

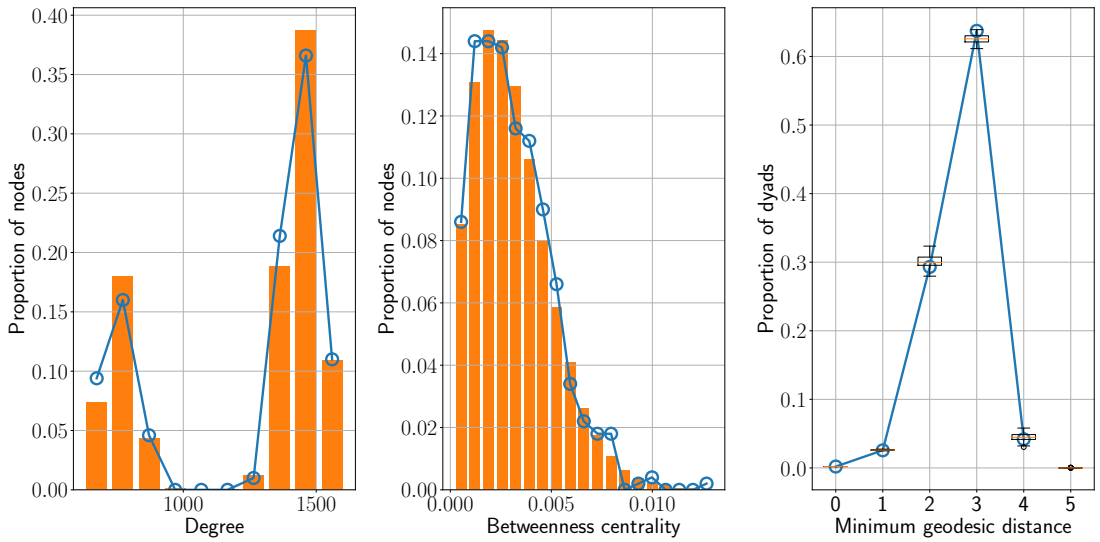

Figure 4.7: Comparisons between two-blocks SBMs generated from the base model (blue line) and from the discrete density estimated from latent positions (histograms and boxplots).

weights sampled from the above discrete distribution. Apparently, the metrics for the network generated from the base model align closely with the distribution of corresponding metrics computed from the sampled graphs.

Remark 4.5.1. Vandermonde matrices–particularly those constructed from monomial bases–are prone to ill-conditioning, especially as the polynomial degree increases, or when the support of the distribution is large. This poses significant numerical challenges when recovering discrete distributions from moment sequences using direct matrix inversion methods, such as solving the system (4.14). To mitigate this issue, we propose an alternative formulation that leverages Chebyshev polynomials of the first kind. These polynomials are orthogonal with respect to the weight function $\frac{1}{\sqrt{1-x^2}}$ on the interval $[-1, 1]$, which endows them with superior numerical stability and conditioning within that domain (Boyd & Gally, 2007).

Our approach is to reformulate the system (4.14) on the basis of Chebyshev polynomials of the first kind. Each expression on the left-hand side of the system can be interpreted as a dot product between the unknown probability vector $\mathbf{p}$ and a polynomial expressed on the monomial basis $\{x^i\}$, evaluated at the support points $\{v_r\}$. Since any polynomial of degree K can be expressed as a linear combination of the first $K+1$ Chebyshev polynomials, we propose to convert each monomial x^k in the system into its Chebyshev expansion. This leads to an equivalent system, with a better conditioned Vandermonde-like matrix constructed using evaluations of Chebyshev polynomials at $\{v_r\}$.

This Chebyshev-based reformulation enhances numerical stability during re-

construction; an attractive feature when dealing with empirically estimated moments, which are inevitably affected by noise due to finite sample size.

4.5.1.1 Chebyshev Polynomial Reformulation

We next provide a detailed description of the Chebyshev-based reformulation of the moment-recovery system in (4.14). By expressing the system on the basis of Chebyshev polynomials of the first kind, we markedly improve the problem's conditioning.

Transformation of monomials into Chebyshev basis

Any monomial x^k of degree $k \leq K$ can be uniquely expanded in the Chebyshev basis $\{T_j(x) : j = 0, 1, \ldots, K\}$:

$$x^k = \sum_{j=0}^{K} c_{kj}\, T_j(x),$$

where the coefficients c_{kj} are computed using recurrence relations or discrete orthogonality integrals. In practice, these coefficients can be precomputed up to degree K using the recurrence:

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_{j+1}(x) &= 2x\, T_j(x) - T_{j-1}(x), \quad j \geq 1, \end{aligned}$$

and the identity relating monomials to Chebyshev polynomials (e.g., via trigonometric definitions or forward recurrence).

Construction of the Chebyshev–Vandermonde matrix

Let $\{v_0, v_1, \ldots, v_R\}$ denote the support points of the discrete distribution. We define the Chebyshev–Vandermonde matrix $\mathbf{V}_{\mathrm{C}} \in \mathbb{R}^{(K+1)\times(R+1)}$ with entries

$$(\mathbf{V}_{\mathrm{C}})_{kr} = T_k\big(v_r^*\big), \quad k = 0, \ldots, K,\ r = 0, \ldots, R,$$

where v_r^* are the support points mapped into the canonical interval $[-1, 1]$ via affine scaling if necessary.

The moment-recovery system then becomes an equivalent system in the Chebyshev basis. Let $\mathbf{C} \in \mathbb{R}^{(K+1)\times(K+1)}$ be the matrix with entries $C_{kj} = c_{kj}$, so that the Chebyshev-moment vector

$$\mathbf{m}_{\mathrm{C}} = \mathbf{C}\mathbf{m}$$

is the projection of the original moment vector $\mathbf{m} = [m_{ij}[0], \ldots, m_{ij}[K]]^\top$ onto the

Chebyshev basis. The system in the Chebyshev basis then reads

$$\mathbf{V}_{\mathrm{C}}\,\mathbf{p} = \mathbf{m}_{\mathrm{C}}.$$

Numerical solution and stability

Because Chebyshev polynomials constitute an orthogonal basis on $[-1, 1]$ with respect to the weight $(1 - x^2)^{-1/2}$, the condition number of $\mathbf{V}_{\mathrm{C}}$ grows only polynomially in K, rather than exponentially as in the monomial basis. Therefore, solving for $\mathbf{p}$ via a standard least-squares or regularized inversion

$$\hat{\mathbf{p}} = \underset{\mathbf{p}\geq 0}{\operatorname{argmin}} \|\mathbf{V}_{\mathrm{C}}\,\mathbf{p} - \mathbf{m}\|_2^2$$

offers improved numerical stability. Additional regularization or non-negativity constraints may be imposed to further enhance stability and ensure valid probability estimates.

4.5.2 Continuous weights distribution

We now move on to the case where the edge weights W_{ij} are continuous random variables. That is, our goal is to determine a pdf g_{ij} such that

$$\int_{\Omega} x^k g_{ij}(x)\,dx = \mathbf{x}_i^{\top}[k]\mathbf{x}_j[k], \quad \text{for all } k = 0, 1, \ldots, K, \tag{4.16}$$

where $\Omega \subset \mathbb{R}$ denotes the (assumed known) support of the random variable W_{ij}.

To identify a unique pdf consistent with this partial information, we turn to the foundational work of Shore and Johnson (1980), who provided an axiomatic justification for using the principle of maximum entropy in such settings. Their key result shows that when one's knowledge about a probability distribution is limited to certain expectation values or constraints given in the form of moments, the only consistent and unbiased method for selecting a distribution is to choose the one that maximizes entropy subject to those constraints. This follows from a set of axioms that any rational updating procedure should satisfy, such as consistency, uniqueness, invariance under coordinate transformations, and system independence.

In this context, entropy refers to the differential entropy of pdf g_{ij}, namely

$$S(g_{ij}) = -\int_{\Omega} g_{ij}(x) \log g_{ij}(x)\,dx.$$

Maximizing $S(g_{ij})$ under the moment constraints (4.16) ensures that no additional structure or assumptions are imposed beyond what is strictly warranted by those constrains. Thus, following Shore and Johnson's derivation, the maximum entropy

principle is not simply a heuristic, but the unique method of inference consistent with the logical requirements of rational belief updating.

Therefore, we formulate the following primal optimization problem:

$$\max_{g\in\mathcal{F}} S(g) \;\text{ s. to }\; \int_\Omega x^k g(x)\,dx - m_k = 0, \quad \text{for all } k = 0, 1, \ldots, K, \tag{4.17}$$

where $m_k = \mathbf{x}_i^\top[k]\mathbf{x}_j[k]$ and $\mathcal{F}$ denotes the space of all probability distributions supported on Ω (as before, for clarity and to avoid notational clutter, we omit the dependence of g and m_k on indices i and j). To solve (4.17), we introduce Lagrange multipliers λ_k and the Lagrangian functional[1]

$$L(g, \boldsymbol{\lambda}) = S(g) - (\lambda_0 - 1)\left(\int_\Omega g(x)\,dx - m_0\right) - \sum_{k=1}^{K} \lambda_k \left(\int_\Omega x^k g(x)\,dx - m_k\right), \tag{4.18}$$

where $\boldsymbol{\lambda} = [\lambda_0, \lambda_1, \ldots, \lambda_K]^\top \in \mathbb{R}^{K+1}$. We will find a solution to (4.17) by solving the dual problem:

$$\min_{\boldsymbol{\lambda}\in\mathbb{R}^{K+1}} d(\boldsymbol{\lambda}), \tag{4.19}$$

where $d : \mathbb{R}^{R+1} \mapsto \mathbb{R}$ is the dual function

$$d(\boldsymbol{\lambda}) = \max_{g\in\mathcal{F}} L(g, \boldsymbol{\lambda}).$$

Using the Euler-Lagrange equation for the calculus of variations we conclude that the function $g_{\boldsymbol{\lambda}}$ that extremizes L is the one that makes the functional derivative $\dfrac{\delta L(g, \boldsymbol{\lambda})}{\delta g(x)}$ equal to zero. By differentiation, we see that such a condition amounts to

$$\frac{\delta L(g, \boldsymbol{\lambda})}{\delta g(x)}\Big|_{g_{\boldsymbol{\lambda}}(x)} = 0 \Leftrightarrow g_{\boldsymbol{\lambda}}(x) = \exp\left(-\sum_{k=0}^{K} \lambda_k x^k\right). \tag{4.20}$$

Looking at the second variation of $L(g, \boldsymbol{\lambda})$ we conclude that this $g_{\boldsymbol{\lambda}}$ maximizes the Lagrangian, so the dual function can be computed as

$$d(\boldsymbol{\lambda}) = L(g_{\boldsymbol{\lambda}}, \boldsymbol{\lambda}) = \sum_{k=0}^{K} \lambda_k m_k + \int_\Omega \exp\left(-\sum_{k=0}^{K} \lambda_k x^k\right) dx - m_0.$$

One can check that $d(\boldsymbol{\lambda})$ is a convex function of $\lambda_0, \lambda_1, \ldots, \lambda_K$ (see, e.g., Kapur

[1] Following Kapur and Kesavan (1992) we have used $\lambda_0 - 1$ as the first Lagrange multiplier for convenience.

and Kesavan (1992, Example 2.3)). Since its partial derivatives are equal to

$$\frac{\partial d(\boldsymbol{\lambda})}{\partial \lambda_k} = m_k - \int_\Omega x^k g_{\boldsymbol{\lambda}}(x)\, dx$$

we have that whenever the gradient of $d(\boldsymbol{\lambda})$ is zero, then $g_{\boldsymbol{\lambda}}$ satisfies the moments constraints (4.16). In other words, if $\boldsymbol{\lambda}^*$ is a solution to the dual problem (4.19), then the function $g_{\boldsymbol{\lambda}^*}$ given by (4.20) both maximizes the Lagrangian (4.18) and satisfies the moments constraints (4.16), i.e., it is a solution to the primal problem (4.17). This implies that strong duality holds for the maximum entropy problem. Therefore, by solving the dual problem, we can recover the maximum entropy pdf we are seeking. Since the dual objective is convex and unconstrained, it can be solved using any standard convex optimization algorithm. In practice, we solve it using the BFGS algorithm (Nocedal & Wright, 2006) available through SciPy's `minimize` function.

Remark 4.5.2. If a function $g_{\boldsymbol{\lambda}}$ such as that in (4.20) satisfies the moments constraints (4.16), then it has maximum entropy among all pdfs g supported in Ω that satisfy such constraints. Indeed, we have that, since $\log g_{\boldsymbol{\lambda}}(x) = -\sum_{k=0}^K \lambda_k x^k$,

$$\begin{aligned} S(g_{\boldsymbol{\lambda}}) &= -\int_\Omega g_{\boldsymbol{\lambda}}(x) \log g_{\boldsymbol{\lambda}}(x)\, dx = \sum_{k=0}^K \lambda_k \int_\Omega g_{\boldsymbol{\lambda}}(x) x^k\, dx = \sum_{k=0}^K \lambda_k \int_\Omega g(x) x^k\, dx \\ &= -\int_\Omega g(x) \log g_{\boldsymbol{\lambda}}(x)\, dx, \end{aligned}$$

where in the third equality we have used that both $g_{\boldsymbol{\lambda}}$ and g satisfy the moments constraints (4.16). Then

$$S(g_{\boldsymbol{\lambda}}) - S(g) = \int_\Omega g(x) \log\left(\frac{g(x)}{g_{\boldsymbol{\lambda}}(x)}\right) dx = D_{KL}(g||g_{\boldsymbol{\lambda}}) \geq 0$$

where D_{KL} is the Kullback-Leibler divergence.

Example 4.5.2. We showcase our method by testing its ability to recover an exponential distribution from its first four moments. Since the pdf of an exponential random variable with parameter α is $\alpha e^{-\alpha x}$, it can be written in the form of (4.20) by setting $\lambda_0 = -\log \alpha$, $\lambda_1 = \alpha$ and $\lambda_k = 0$ for all $k \geq 2$, so we can check whether the Lagrange multipliers λ_k obtained with our gradient descent solution are close to these values. The results for an exponential random variable with parameter $\alpha = 2$ are depicted in Figure 4.8, where we show the λ_k's upon convergence of our GD algorithm for 100 random initializations of $\boldsymbol{\lambda}$. We also show the Lagrange multi-

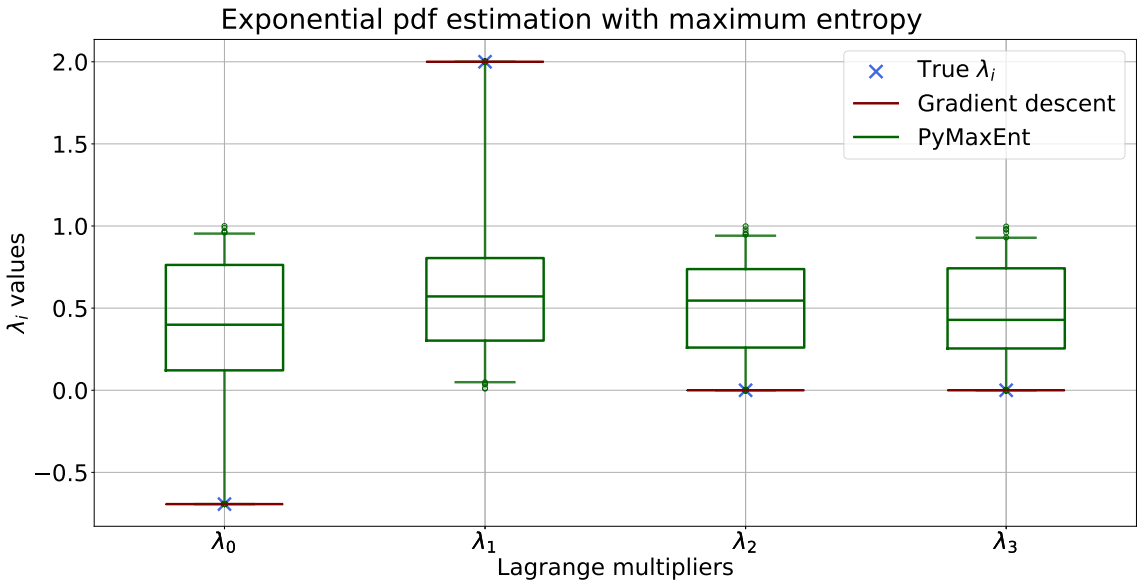

Figure 4.8: Box plots of Lagrange multipliers for maximum entropy estimation of an exponential rv distribution via our dual approach (red) and the method from Saad and Ruai (2019) (PyMaxEnt, green) for 100 random initializations. Our approach always converges to the true value, while PyMaxEnt does not.

pliers obtained with the method proposed in Saad and Ruai (2019). In that work, the authors estimate $\boldsymbol{\lambda}$ by using Newton's method to approximate the solutions of the system of nonlinear equations that arises from imposing that the function $g_{\boldsymbol{\lambda}}$ in (4.20) satisfies the moments constraints (4.16), i.e., the system

$$\int_{\Omega} x^k \exp\left(-\lambda_0 - \lambda_1 x - \lambda_2 x^2 s - \cdots - \lambda_R x^K\right) dx = m_k \quad \forall\, k = 0, \ldots, K.$$

As shown in Figure 4.8 the Lagrange multipliers that we obtain using this approach are, most of the time, quite far from the actual values. This is because Newton's method is sensitive to initialization and heavily relies on having an initial guess that lies on the solution's basin of attraction. Our method, in contrast, always converges to the true solution. That is because our dual function is convex, so as long as the maximum entropy problem has a solution our method will converge to it no matter where it starts.

Remark 4.5.3. This maximum entropy approach can be easily modified to accommodate discrete distributions. This is useful when we have access to fewer moments than the amount of symbols the discrete random variable takes, in which case we can no longer use the procedure described in Section 4.5.1.

Assuming that our random variable takes on the values $v_0, \ldots, v_R$, we wish to

find its corresponding probabilities $p_0, \ldots, p_R$ by maximizing Shannon's entropy

$$S(p_0, \ldots, p_R) = -\sum_{r=0}^{R} p_r \log p_r$$

subject to the moments constrains

$$\sum_{r=0}^{R} v_r^k p_r = m_k \ \ \forall k = 0, \ldots, K.$$

We define the Lagrangian in analogy to (4.18) and by differentiation we find that it has a maximum when $p_r = \exp\left(-\sum_{k=0}^{K} \lambda_k v_r^k\right)$. Thus, the dual function can be computed as:

$$d(\boldsymbol{\lambda}) = \sum_{k=0}^{K} \lambda_k m_k + \sum_{r=0}^{R} \exp\left(-\sum_{k=0}^{K} \lambda_k v_r^k\right) - m_0,$$

which, as before, is a convex function of $\lambda_0, \ldots, \lambda_K$. Therefore, we can find its minimizer using any standard convex optimization algorithm and from it compute the p_r's that maximize Shannon's entropy.

Example 4.5.3. To further illustrate the flexibility of the proposed graph generation method, we consider a two-block weighted SBM network with $N = 500$ nodes, comprising 350 nodes in community 1 and 150 nodes in community 2. The block probability matrix is given by

$$\mathbf{B} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix},$$

which implies that the graph is fully connected. This setup is chosen because assigning probabilities strictly less than one would introduce a point mass at zero in the edge-weight distribution, thereby resulting in a mixture distribution – a setup which we address in the next section. Edge weights are sampled from distinct continuous distributions depending on the community membership of the connected nodes: for intra-community edges within community 1, weights follow a normal distribution $\mathcal{N}(6, 1)$; for intra-community edges within community 2, weights follow an exponential distribution with rate parameter $\lambda = \frac{1}{3}$; and for inter-community edges, weights follow a normal distribution $\mathcal{N}(1, 0.01)$. Using (4.4), we compute the analytical latent positions for each node, where $m_d[k]$ now denotes the k-th moment of the corresponding distribution associated with each block. Edge weights thus arise from a mixture of three distributions, leading to a richer moment structure.

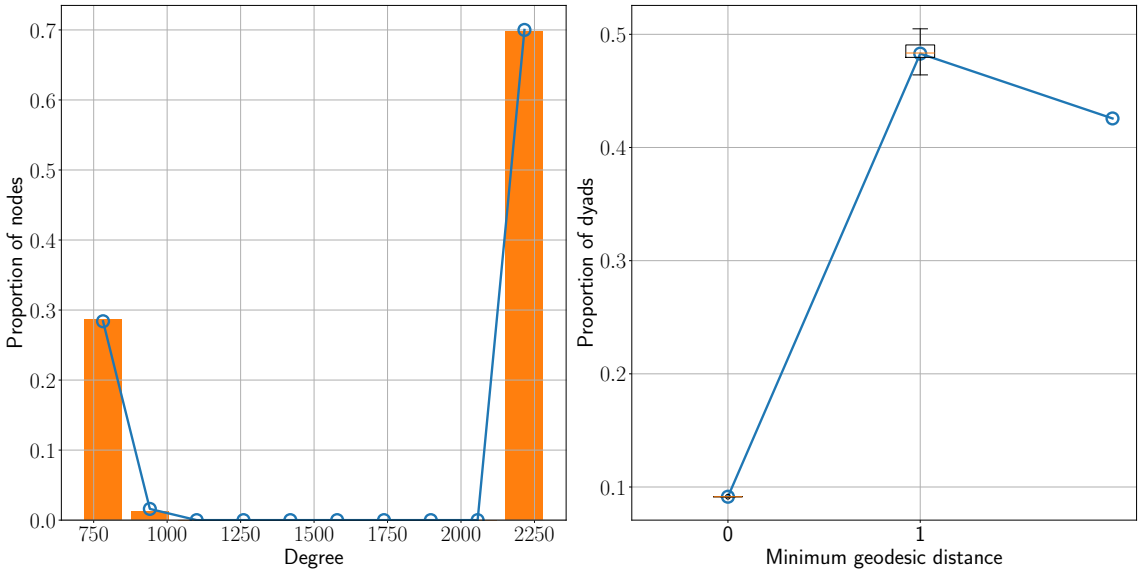

Figure 4.9: Comparisons between two-block SBMs generated from the base model (blue line) and from the pdf estimated with our method for solving the maximum entropy problem from latent positions (histogram and boxplot). Since in this setup graphs are fully connected, we do not report results for betweenness centrality.

As before, for each edge we compute the finite moment sequence $m_{ij}[k] = \mathbf{x}_i^\top[k]\mathbf{x}_j[k]$, for $k = 0, \ldots, 5$. We approximate the underlying edge-weight distributions using those first six moments, estimating them via the maximum entropy principle by solving the previously derived dual problem. Figure 4.9 displays network-wide distributions of summary statistics (degree and geodesic distance) for 100 networks generated using this procedure, alongside the corresponding metrics of a single network sampled directly from the base model, i.e., a two-class, fully connected weighted SBM, and edge weights sampled according to the aforementioned mixture of continuous distributions. The metric distributions computed from the base model are in close agreement with those obtained from the generated graphs, thus validating the accuracy of the moment-based generation process in the presence of heterogeneous weight distributions.

4.5.3 Mixed weights distribution

We now focus on the case where the weights' distribution is a mixture of discrete and continuous components, with its pdf having the form

$$g_{ij}(x) = p_0^{ij}\delta(x) + (1 - p_0^{ij})h_{ij}(x) \tag{4.21}$$

where $\delta(x)$ is Dirac's delta, $h_{ij}(x)$ is an unknown pdf and $p_0^{ij} \in [0, 1]$ is an unknown parameter that controls the edge-formation probability between nodes i and j . If p_0^{ij}

were known, we could estimate h_{ij} from the prescribed moments $m_{ij}[k] = \mathbf{x}_i^\top[k]\mathbf{x}_j[k]$, since

$$m_{ij}[0] = p_0^{ij} + (1 - p_0^{ij}) \int_\Omega h_{ij}(x)\, dx,$$
$$m_{ij}[k] = (1 - p_0^{ij}) \int_\Omega x^k h_{ij}(x)\, dx, \quad \forall\, k \geq 1,$$

where Ω is the support of $h_{ij}(x)$. This implies that we are looking for a pdf h_{ij} with moments

$$\int_\Omega h_{ij}(x)\, dx = \frac{m_{ij}[0] - p_0^{ij}}{1 - p_0^{ij}},$$
$$\int_\Omega x^k h_{ij}(x)\, dx = \frac{m_{ij}[k]}{1 - p_0^{ij}}, \quad \forall\, k \geq 1,$$

so we can use the procedure in Section 4.5.2 to find h_{ij} by maximizing its entropy.

In order to estimate p_0^{ij}, we propose to simultaneously estimate it for every edge via the ASE of the binary matrix $\mathbf{A} := \mathbb{I}\{\mathbf{W} > \mathbf{0}\}$, where $\mathbb{I}\{\cdot\}$ denotes the entry-wise matrix indicator function, i.e., $A_{ij} := \mathbb{I}\{W_{ij} > 0\}$. This estimator can be justified by noting that $\mathbb{E}[A_{ij}] = 1 - p_0^{ij}$. Indeed,

$$\mathbb{E}[A_{ij}] = \int_\Omega \mathbb{I}\{W_{ij} > 0\} g(x)\, dx = \int_\Omega (1 - p_0^{ij}) h_{ij}(x)\, dx = 1 - p_0^{ij},$$

where we have assumed that $\Omega \subset \mathbb{R}^+$. Let $\hat{\mathbf{X}}_A$ denote the ASE of $\mathbf{A}$. Then, in light of our consistency result in Section 4.4.1, matrix $\hat{\mathbf{P}} := \hat{\mathbf{X}}_A \hat{\mathbf{X}}_A^\top$ converges to $\mathbb{E}[\mathbf{A}]$ in the $\|\cdot\|_\infty$ sense, as $N \to \infty$.

Example 4.5.4 (Reproducing real networks.). In this example the latent positions are not given and instead we estimate them from a real graph that is observed. As a result, now the input is a finite sequence of *approximate* moments. We study a dataset that records football matches between national teams (Li & Mateos, 2022). For a given time period, we construct a graph in which nodes represent countries, and an edge exists between two countries if they have played at least one match against each other during that period. The edge weight corresponds to the number of matches played between them. Next, we compute the latent positions for the edge weight distribution by performing the ASE of $\mathbf{W}^{(k)}$, where $\mathbf{W}^{(k)}$ denotes the k-th entry-wise power of the weight matrix $\mathbf{W}$, with k ranging from 0 to a prespecified value K. Using these latent positions, we compute a finite moment sequence for each edge via the corresponding inner product and estimate its density using the procedure described earlier in this section.

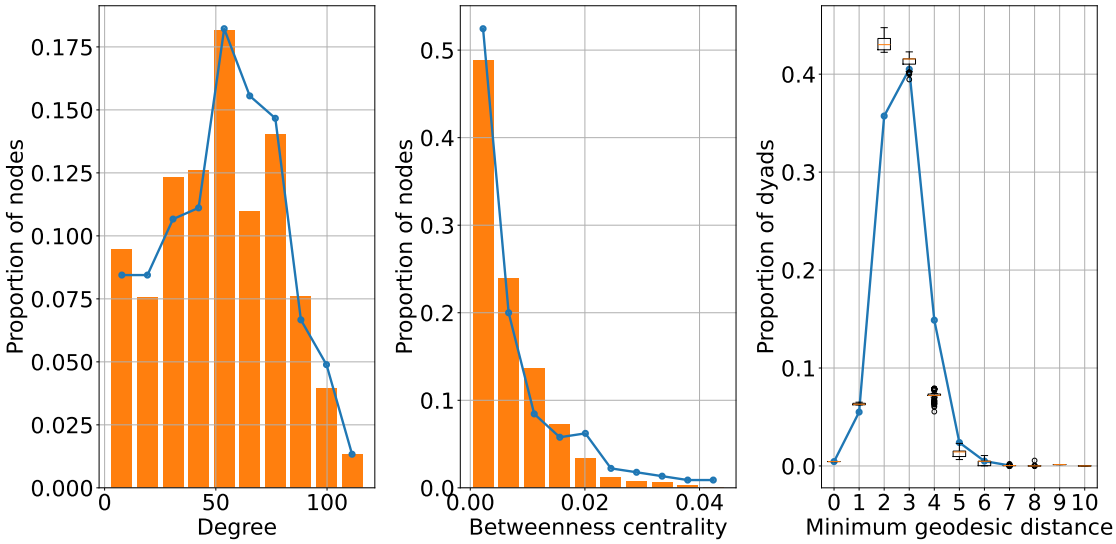

Figure 4.10: Graph generation metrics the football dataset Li and Mateos (2022). Metrics for the true graph are shown with a blue solid line, while a histogram or a boxplot shows the results for the corresponding metric for 100 synthetic graphs generated using the estimated mixed densities.

By assuming a mixture-like density as in (4.21), we explicitly model the sparsity pattern of the football network. This approach enables us to generate synthetic networks that both conform to the observed sparsity pattern and exhibit edge weights that closely mimic those of the actual network. Figure 4.10 shows several metrics for the actual network of football matches during the period 2010–2016, alongside the corresponding metrics for 100 synthetically-generated networks using the aforementioned pdf estimation plus weighted graph sampling procedure. For this example, we used $K = 2$, meaning that the zeroth, first, and second moments were employed for density estimation.

To further demonstrate the quality of the generated graphs, we conducted a community structure analysis. Since matches between countries belonging to the same football confederation are more frequent, one would expect to observe a clear community structure in the real network, with clusters roughly corresponding to confederations. This can be verified by looking at Figure 4.11, where we show the results of applying the classic Louvain clustering algorithm (Blondel et al., 2008) to the real graph. The algorithm detects as many clusters as there are confederations (six), with community membership reflecting, most of the times, the actual confederations teams belong to: for example, Australia is geographically in Oceania, but is a member of the Asian Football Confederation (AFC). The only major discrepancy is in North and Central America: while the countries in those subcontinents belong to the Confederation of North, Central America and Caribbean Association Football (CONCACAF), they are clustered together with southern countries, which

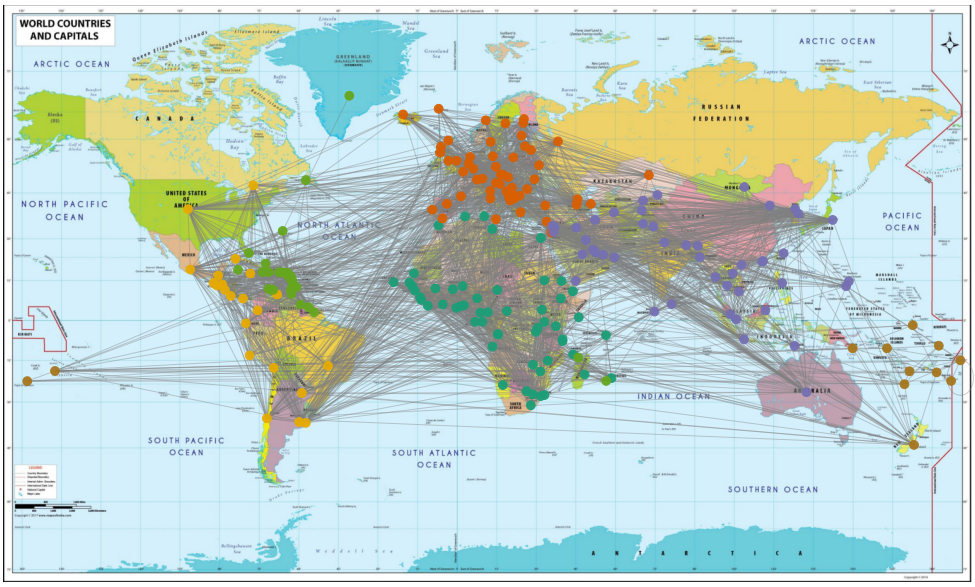

Figure 4.11: Result of applying the Louvain algorithm Blondel et al. (2008) to the network of international football matches. Nodes with the same color belong to the same community.

belong to the South American Football Confederation (CONMEBOL). This might be because the former are often invited to participate in CONMEBOL's flagship tournament, the Copa América, inflating the amount of games they play against CONMEBOL members. The fact that most Caribbean members of CONCACAF belong to a separate cluster further strenthengs this hypothesis, since they rarely play in the Copa América.

We then ran the Louvain algorithm on the simulated networks and compared the results with those of the real network; see the results in Figure 4.12. On the left panel we show a histogram of the amount of communities in the synthetic graph's largest connected component. As we can see, all graphs have either 5 or 6 communities in that component, which is consistent with the structure we discussed perviously. And while most graphs tend to have one community less than those in the real graph, we found that most of the times this is because countries from Oceania tend to be clustered with the AFC.

Besides simply looking at the amount of communities, we compared whether clusters in the simulated graphs matched those of the real network. To that end, we computed three different metrics of cluster agreement between each synthetic graph and the real one. Those metrics were V-measure (Rosenberg & Hirschberg, 2007), the Adjusted Rand Index (Hubert & Arabie, 1985), and the Adjusted Mutual Information (Vinh et al., 2009). On the right panel of Figure 4.12 we show a boxplot

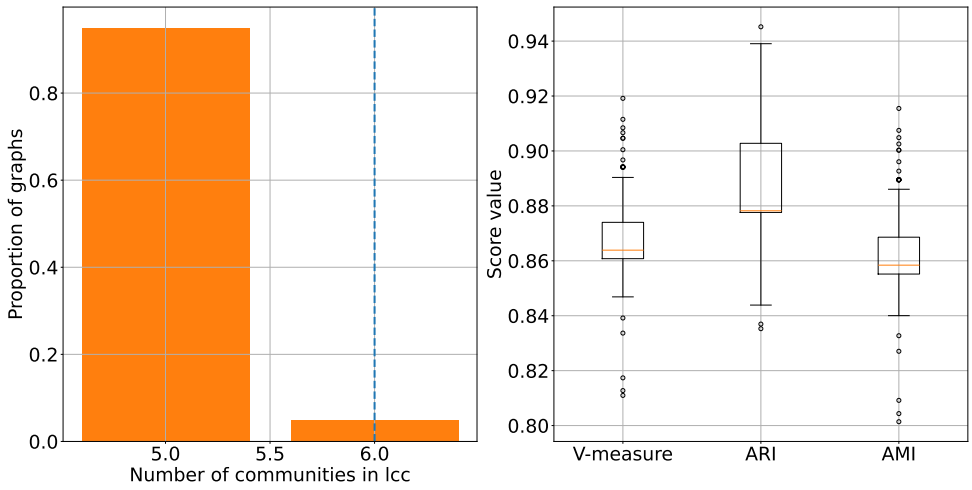

Figure 4.12: Comparisons between community structure of the real football matches network and its synthetic replicates. Left: histogram of number of communities in the largest connected component (lcc) for synthetic graphs. Right: Boxplot for three metrics of clustering agreement between real and synthetic networks: V-measure, Adjusted Rand Index (ARI) and Adjusted Mutual Information (AMI).

for each metric. Apparently, the overall community structure in the synthetic graphs follows closely that of the actual network.

Remark 4.5.4. Since the estimation of p_0^{ij} is based on the consistency result from Section 4.4.1, which guarantees convergence as the number of nodes $N \to \infty$, the finite size of the graph introduces a non-negligible finite-sample deviation. This deviation partially accounts for the (arguably quite minor) discrepancies observed in the metrics shown in Figure 4.10, underscoring the method's robustness.

4.6 Concluding remarks

We developed the Weighted RDPG (WRDPG) model, which extends the vanilla RDPG framework to effectively capture the intricacies of weighted graphs characterized by heterogeneous edge weight distributions. Our approach assigns latent positions to nodes, parameterizing edge weight distributions' moments using moment-generating functions. This innovative modeling strategy enhances the WRDPG's capability to distinguish between weight distributions that may have the same mean but differ in higher-order moments, thus providing a richer representation of network data than previous attempts to weighted RDPG modeling. We derived statistical guarantees (consistency and asymptotic normality) for our estimator of latent positions, leveraging the well-established adjacency spectral embedding method. Furthermore, we developed a generative framework that approximates desired weight

distributions from a given (or estimated) latent position sequence and facilitates sampling synthetic graphs that closely mimic the structure and characteristics of real-world networks. Through a series of illustrative examples, we have demonstrated the effectiveness of the WRDPG model in practical applications. We show how our model can recover latent position structures and replicate key graph metrics, exhibiting a discriminative power essential for community detection tasks. We have also shown that the model accommodates various edge weight distributions, ranging from discrete to continuous, or even mixed random variables. To achieve this versatility, we developed an improved computational method to estimate a continuous density by maximizing the entropy subject to moment constraints. This result is of independent interest and could permeate benefits in other, broader contexts. The findings of this study highlight the importance of incorporating higher-order moments in graph models, contributing to the methods of inference and generation in network data analysis.

Appendix 4.A: Consequences of Assumptions 2 and 3 regarding the largest eigenvalues of $\mathbf{M}_k$ and $\mathbf{W}_k$

In this appendix, we show that Assumptions 2 and 3 for the WRDPG model imply that for all $k \geq 0$, the nonzero eigenvalues of $\mathbf{M}_k := \mathbf{X}[k]\mathbf{X}^\top[k]$ are $\Theta_{\mathbb{P}}(N)$, and that the same holds for the top d eigenvalues of $\mathbf{W}_k := \mathbf{W}^{(k)}$. We begin by proving this for $\mathbf{M}_k$. The proof scheme follows that of (Sussman et al., 2014, Proposition 4.3). In what follows, the notation $\lambda_i(\mathbf{A})$ denotes the i-th largest-magnitude eigenvalue of matrix $\mathbf{A}$.

Lemma 4.A.1. *Let* $(\mathbf{W}, \mathbf{X}_k) \sim \text{WRDPG}(F)$*, where* F *satisfies Assumption 2. Then for all* $k \geq 0$ *it holds that* $\lambda_i(\mathbf{M}_k) = \Theta_{\mathbb{P}}(N)$ *for* $i = 1, \ldots, d$*, while* $\lambda_i(\mathbf{M}_k) = 0$ *for* $i = d+1, \ldots, N$*.*

Proof. That $\lambda_i(\mathbf{M}_k) = 0$ for $i = d+1, \ldots, N$ is immediate since $\mathbf{M}_k = \mathbf{X}[k]\mathbf{X}^\top[k]$, so $\mathbf{M}_k$ is a $N \times N$ matrix with rank at most d. For $i = 1, \ldots, d$, note that

$$\lambda_i(\mathbf{M}_k) = \lambda_i(\mathbf{X}[k]\mathbf{X}^\top[k]) = \lambda_i(\mathbf{X}^\top[k]\mathbf{X}[k]).$$

Now, each entry of $\mathbf{X}^\top[k]\mathbf{X}[k]$ is:

$$(\mathbf{X}^\top[k]\mathbf{X}[k])_{ij} = \sum_{l=1}^{N} (\mathbf{x}_l[k])_i (\mathbf{x}_l[k])_j ,$$

where $(\mathbf{x}_l[k])_i$ denotes the i-th entry of vector $\mathbf{x}_l[k]$. Then $(\mathbf{X}^\top[k]\mathbf{X}[k])_{ij}$ is a sum of

independent random variables, each with expectation $\mathbb{E}\left[(\mathbf{x}_l[k])_i (\mathbf{x}_l[k])_j\right] = (\mathbf{\Delta}_k)_{ij}$, where $(\mathbf{\Delta}_k)_{ij}$ is the (i,j) entry of the second moment matrix $\mathbf{\Delta}_k$ from Assumption 2. Since, for each k, the inner product between the rows of $\mathbf{X}[k]$ equals the k-th moment of some random variable–which, by the definition of our model, we assume to be finite for all k–it follows that all rows of $\mathbf{X}[k]$ (which lie in $\mathbb{R}^d$) must have bounded norm, with a bound that does not depend on N. Therefore, each term in the above sum is bounded by some constant $L_k > 0$, so by Hoeffding's inequality (Vershynin, 2018, Theorem 2.6.2) we have that:

$$\mathbb{P}\left(\left|(\mathbf{X}^\top[k]\mathbf{X}[k])_{ij} - N(\mathbf{\Delta}_k)_{ij}\right| \geq t\right) \leq 2\exp\left(\frac{-2t^2}{N^2 L_k^2}\right).$$

Choosing $t = \frac{L_k}{\sqrt{2}} N^{1/2}\log^{1/2} N$ then shows that

$$\left|(\mathbf{X}^\top[k]\mathbf{X}[k])_{ij} - N(\mathbf{\Delta}_k)_{ij}\right| = \mathcal{O}_{\mathbb{P}}\left(N^{1/2}\log^{1/2} N\right).$$

Taking a union bound then shows that $\left\|\mathbf{X}^\top[k]\mathbf{X}[k] - N\mathbf{\Delta}_k\right\|_{\mathrm{F}}$ is also $\mathcal{O}_{\mathbb{P}}\left(N^{1/2}\log^{1/2} N\right)$, and since the spectral norm is dominated by the Frobenius norm, we have $\left\|\mathbf{X}^\top[k]\mathbf{X}[k] - N\mathbf{\Delta}_k\right\|_2 = \mathcal{O}_{\mathbb{P}}\left(N^{1/2}\log^{1/2} N\right)$. A corollary of Weyl's inequality (Horn & Johnson, 2012, Corollary 7.3.5) then implies that:

$$\left|\lambda_i\left(\mathbf{X}^\top[k]\mathbf{X}[k]\right) - N\lambda_i(\mathbf{\Delta}_k)\right| = \mathcal{O}_{\mathbb{P}}\left(N^{1/2}\log^{1/2} N\right).$$

Since $\lambda_i(\mathbf{\Delta}_k) = \Theta_{\mathbb{P}}(1)$, this in turn implies that $\lambda_i\left(\mathbf{X}^\top[k]\mathbf{X}[k]\right) = \Theta_{\mathbb{P}}(N)$, which completes the proof. □

The next lemma shows that, under Assumptions 2 and 3, the top d eigenvalues of $\mathbf{W}_k$ are $\Theta_{\mathbb{P}}(N)$, while the remaining ones are within $\log^{k\theta} N$ of zero with high probability.

Lemma 4.A.2. *Let $(\mathbf{W}, \mathbf{X}_k) \sim \mathrm{WRDPG}(F)$, where F satisfies Assumption 2 and $\mathbf{W}$ satisfies Assumption 3. Then, for all $k \geq 0$ it holds that $\lambda_i(\mathbf{W}_k) = \Theta_{\mathbb{P}}(N)$ for $i = 1, \ldots, d$, while $\lambda_i(\mathbf{W}_k) = \mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right)$ for $i = d+1, \ldots, N$.*

Proof. Again, by (Horn & Johnson, 2012, Corollary 7.3.5) we have that:

$$|\lambda_i(\mathbf{W}_k) - \lambda_i(\mathbf{M}_k)| \leq \|\mathbf{W}_k - \mathbf{M}_k\|_2.$$

Therefore, using Lemma 4.4.3 we have that $|\lambda_i(\mathbf{W}_k) - \lambda_i(\mathbf{M}_k)| = \mathcal{O}_{\mathbb{P}}\left(\log^{k\theta} N\right)$. Applying Lemma 4.A.1 implies the desired result. □

Chapter 5

Online change point detection for network data

Online (or sequential) change-point detection (CPD) is the problem of deciding whether (and if so when) the generating process underlying an observed data stream has changed–see e.g., (Page, 1954) for seminal work in the context of quality control.

The goal is to flag a problem (in order to take corrective actions) as soon as it happens, while controlling the probability of false alarm. Unlike offline or batch processing–see e.g., (Truong et al., 2020)–, in the online CPD setting we do not have access to the full data sequence which could well be infinitely long.

Given the ubiquity of datasets that are generated in a streaming fashion, online CPD is a timely research area with applications to sensor networks (He et al., 2018), financial markets (Keshavarz et al., 2020), or, social networks (Kaushik et al., 2021; Peel & Clauset, 2014). As these examples suggest, data are increasingly high-dimensional and possibly non-Euclidean. Indeed, here we will consider *network data streams* in the form of graph sequences. In a nutshell, given an incoming sequence of random (possibly weighted and directed) graphs, we want to signal if and when the data generating mechanism changes.

5.1 Relation to prior work on online CPD for network data

Sequential CPD approaches are often parametric, and follow the general premise of minimizing detection delay subject to a constraint on the test's type-I error. For network data existing methods look for changes in the graphs' distribution (He et al., 2018; Keshavarz et al., 2020; Peel & Clauset, 2014), their topology (Kaushik et al., 2021) and community structure (M. Zhang et al., 2020), or else the distribution of signals supported on the nodes (Ferrari et al., 2019). Some of these (Kaushik et al.,

2021; Keshavarz et al., 2020; Peel & Clauset, 2014) are only applicable to undirected graphs. A sequential non-parametric, k-nearest neighbors-based approach was developed in Chen (2019), solely requiring a pairwise distance between samples (e.g., the Frobenius distance between graph adjacency matrices). Unlike methods based on generative models, said distance is prone to overlooking simple changes in network structure; see the comparisons in Section 5.4.1. A computationally-intensive model-based CPD effort advocates the Generalized Hierarchical Random Graph (GHRG) model in Peel and Clauset (2014), which monitors posterior Bayes factors for all partitions of the data over a sliding window. The approach in H. Wang et al. (2014) is more general, as it considers the workhorse Stochastic Block Model (SBM). The distribution of two so-termed scan statistics is derived to signal changes in the input graph sequence.

Going beyond SBMs, the recent work (Yu et al., 2021) considers an inhomogeneous Bernoulli graph; whereby the existence of an edge between a pair of nodes (i, j) is a Bernoulli random variable with probability P_{ij}, independent of all other pairs. Each timestep, two statistics are computed for a logarithmic grid of previous instants to check whether they exceed a certain threshold. Evaluating these statistics requires computing the eigendecomposition of an $N \times N$ matrix–N being the number of graph nodes. In addition to being computationally intensive, the algorithm in Yu et al. (2021) has to store all historical data in memory, which may pose a major hurdle even for moderate-sized networks. The procedure offers solid theoretical guarantees on the detection delay and average run length. Here instead we resort to the RDPG model since its interpretability provides an attractive feature that simplifies the explanation of the detected change-points.

5.2 Contributions and chapter outline

Building on (Kirch & Tadjuidje Kamgaing, 2015), we assume a clean historical dataset with no change-points is available, from which we estimate the latent nodal vectors via the ASE in an offline training phase. As new data arrive in a streaming fashion during the operational phase, the novel online CPD algorithm (Section 5.3) recursively updates a *monitoring function* statistic whose null distribution we characterize analytically via asymptotic arguments. In addition to providing theoretical guarantees on the false alarm rate of the resulting online CPD scheme, an attractive feature is its limited memory footprint –we store a single $N \times N$ matrix in memory (in addition to the estimated latent vectors, naturally). Moreover, the resulting lightweight statistic updates are an order-of-magnitude more efficient than those based on repeated eigendecompositions. Using simplifying approximations we derive conditions under which changes may go undetected.

Numerical tests in Section 5.4 corroborate the effectiveness of the proposed online CPD method, using both simulated and real network datasets that we share in our Github repository. Concluding remarks are outlined in Section 5.5.

The contents of this chapter are based on the paper Marenco et al., 2022. The code to reproduce the experiments reported below is available at https://github.com/git-artes/cpd_rdpg.

5.3 Proposed approach

Our idea to develop an online CPD framework for network data is to endow sequential CPD techniques with a graph representation learning substrate based on RDPGs. For clarity of exposition, we first focus on the unweighted, undirected case before presenting the more general setting.

5.3.1 Problem statement

Suppose we acquire a batch of m independent graphs, with adjacency matrices $\mathbf{A}_1 \ldots, \mathbf{A}_m$, all adhering to an RDPG with latent position matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, in which all matrices stem from the same RDPG model. We will refer to that sequence as the training data set, which is used in an *offline* initialization phase to estimate model parameters from the null model. During the operational phase we observe a (possibly infinite) sequence of streaming adjacency matrices $\mathbf{A}_{m+1}, \mathbf{A}_{m+2}, \ldots$, and would like to detect at what time $t > m$ (if any) the null model described in (2.1) from Chapter 2 is no longer valid (i.e., drifts from the aforementioned RDPG model represent the alternative hypothesis). We tackle this CPD problem in an *online* fashion, meaning graph observations $\{\mathbf{A}_{m+k}\}_{k \geq 1}$ are sequentially and efficiently monitored as they are acquired, without having to store the whole multivariate time series. This way, the algorithm's computational complexity and memory footprint does not grow with k. Another attractive feature is the possibility of detecting the change in (pseudo) real-time, ideally soon after it occurs and with control on the probability of false alarm (i.e., type-I error).

We will also consider generalizations of the aforementioned baseline CPD problem in order to account for weighted and directed graph sequences. This calls for fundamentally re-examining the RDPG model to accommodate said observations –especially in the weighted case–, as well as the associated embedding algorithms and the overall online CPD framework.

5.3.2 General algorithmic framework

We build on the so-called estimating function approach for sequential CPD (Kirch & Tadjuidje Kamgaing, 2015; Kirch & Weber, 2018), which we markedly broaden to accommodate network data. The central notion behind this online CPD method is to consider a *monitoring function* $\mathbf{H}$ of each streaming graph $\mathbf{A}_t$, that should satisfy $\mathbb{E}(\mathbf{H}) = \mathbf{0}$ under the null hypothesis. If one monitors a cumulative sum of $\mathbf{H}$, that quantity should intuitively remain small provided there are no changes in the underlying model. If there is a change however, then $\mathbb{E}(\mathbf{H}) \neq \mathbf{0}$ and we should observe a drift in the trend of the sum.

As proposed in (Kirch & Tadjuidje Kamgaing, 2015) for a network-agnostic setting, we first estimate the parameters of the underlying null RDPG model using the training data set, i.e., we estimate the latent positions matrix $\mathbf{X}$. The estimation should be carried out with an *estimating function* $\mathbf{G}$, where the estimated parameter $\hat{\mathbf{X}}$ is the solution to a system of equations of the form

$$\sum_{t=1}^{m} \mathbf{G}(\mathbf{A}_t, \hat{\mathbf{X}}) = \mathbf{0}. \tag{5.1}$$

To define such a function for our problem, given the training data set we estimate $\mathbf{X}$ as the ASE corresponding to the mean adjacency matrix $\bar{\mathbf{A}} = \frac{1}{m}\sum_{t=1}^{m} \mathbf{A}_t$.

Remark 5.3.1 (ASE variance reduction). Dispersion of ASE estimates can be reduced if one has access to multiple observations from the underlying RDPG. Indeed, let $\mathbf{A}_1 \ldots, \mathbf{A}_m$ be an independent sequence of adjacency matrices, all adhering to an RDPG with latent position matrix $\mathbf{X} \in \mathbb{R}^{N\times d}$. Define the mean adjacency matrix

$$\bar{\mathbf{A}} = \frac{1}{m}\sum_{t=1}^{m} \mathbf{A}_t, \tag{5.2}$$

and henceforth let $\hat{\mathbf{X}}$ be the ASE decomposition of $\bar{\mathbf{A}}$; i.e., the solution of (2.2) using $\bar{\mathbf{A}}$ instead of $\mathbf{A}$. Since $\bar{\mathbf{A}}$ is also an unbiased estimator of $\mathbf{P}$ and $\operatorname{var}\left[\bar{A}_{ij}\right] = \frac{1}{m}P_{ij}(1-P_{ij})$, then as $N \to \infty$ the estimated latent positions $\hat{\mathbf{X}}$ will follow a normal distribution with variance scaled by $\frac{1}{m}$ relative to the variance of the ASE obtained from a single graph as in (2.2) (R. Tang et al., 2018). The alternative of averaging individual ASEs is problematic due to the rotational ambiguity discussed in Remark 2.2.1. Indeed, alignment of the (rotated) ASEs of a graph collection would entail solving several Procrustes distance minimization problems, or else computing the so-termed omnibus embedding (Levin et al., 2017).

Taking the derivative w.r.t. $\mathbf{X}$ of the objective function in (2.2) (with $\mathbf{A} \leftarrow \bar{\mathbf{A}}$) and setting it to zero, we arrive at

$$\sum_{t=1}^{m} \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t \right) \hat{\mathbf{X}} = \mathbf{0},$$

suggesting the use of $\mathbf{G}(\mathbf{A}_t, \hat{\mathbf{X}}) = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t \right) \hat{\mathbf{X}}$ as the estimating function. Accordingly, $\mathbf{G}$ amounts to projecting the residual $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t$ onto $\hat{\mathbf{X}}$.

In order to detect a change on the underlying model during the operational phase, we will track the cumulative sum (CUSUM) of a monitoring function $\mathbf{H}$ as new adjacency matrices arrive for $t \geq m + 1$, namely

$$\mathbf{S}[m, k] = \sum_{t=m+1}^{m+k} \mathbf{H}(\mathbf{A}_t, \hat{\mathbf{X}}).$$

While it is possible (and often natural) to use the same function for both estimation and monitoring (i.e., $\mathbf{H} = \mathbf{G}$), we show in Section 5.4.1 that adopting the residual itself instead of a projection yields in a more powerful detector. Thus, we choose

$$\mathbf{H}(\mathbf{A}_t, \hat{\mathbf{X}}) = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t.$$

We reiterate here that the matrix $\hat{\mathbf{X}}$ is computed during training, via the ASE of the average $\bar{\mathbf{A}}$ of the adjacency matrices in the training set. Once monitoring starts, $\hat{\mathbf{X}}$ is fixed and we do not recompute the ASE for new observations.

Since all involved matrices are hollow and symmetric, we only need to consider entries, say, above the main diagonal. It will also prove useful in the analysis that follows to vectorize the resulting values. We thus define a vector function $\mathbf{h}$ as

$$\mathbf{h}(\mathbf{A}_t, \hat{\mathbf{X}}) = \operatorname{vec}\left[\operatorname{triu}\left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t \right) \right], \tag{5.3}$$

where $\operatorname{vec}(\operatorname{triu}(\mathbf{B}))$ means arranging the entries above the main diagonal of matrix $\mathbf{B}$ in a vector. If $\mathbf{B} \in \mathbb{R}^{N \times N}$, then $\operatorname{vec}(\operatorname{triu}(\mathbf{B})) \in \mathbb{R}^r$, with $r := \frac{N(N-1)}{2}$.

If the norm of the partial sum

$$\mathbf{s}[m, k] = \sum_{t=m+1}^{m+k} \mathbf{h}(\mathbf{A}_t, \hat{\mathbf{X}}) \tag{5.4}$$

exceeds a certain threshold, we will conclude that the model is no longer valid. Let us then denote our CUSUM statistic as

$$\Gamma[m, k] = \|\mathbf{s}[m, k]\|_2^2.$$

Algorithm 4 Online change-point detection for RDPGs

Require: Training graphs $\mathbf{A}_t, t = 1 \dots m$.
Compute the ASE $\hat{\mathbf{X}}$ of $\bar{\mathbf{A}}$ in (5.2) (see Remark 5.3.1)
Compute threshold function c_α (see Section 5.3.5)
Initialize partial sum $\mathbf{s}[m, 0] = \mathbf{0}$
for $k = 1, 2, \dots$ **do**
  Acquire graph $\mathbf{A}_{m+k}$
  Compute monitoring function $\mathbf{h}(\mathbf{A}_{m+k}, \hat{\mathbf{X}})$
  Update CUSUM statistic $\Gamma[m, k]$ (see Remark 5.3.2)
  if $w[k]\Gamma[m, k] > c_\alpha[k]$ **then**
    Change point detected at time $k^* = k$
    break
  end if
end for
return k^*.

In order to control the variance of $\Gamma[m, k]$ as k grows, a weighting function $\omega[k]$ is also introduced. We use $\omega[k] = (rk^{3/2})^{-1}$ and instead monitor $\omega[k]\Gamma[m, k]$; the reason for this choice is explained in the next section when we derive said variance for the null distribution.

All in all, the null hypothesis of no change will be rejected at the first time instant k when

$$\omega[k]\Gamma[m, k] > c_\alpha[k],$$

where $c_\alpha[k]$ is a certain threshold that depends on the distribution of $\omega[k]\Gamma[m, k]$ under the null hypothesis and the prescribed type-I-error level α. In the next section we will discuss how this threshold is chosen after characterizing the running statistic's null distribution. A pseudocode of the online CPD method including the offline (training) and operational (monitoring) phases is tabulated under Algorithm 4.

Remark 5.3.2 (Computational complexity). Efficient *recursive* calculation of the cumulative monitoring function $\mathbf{s}[m, k] = \mathbf{s}[m, k-1] + \mathbf{h}(\mathbf{A}_{m+k}, \hat{\mathbf{X}})$ incurs $O(N^2)$ memory storage and computational complexity. The cost of forming the weighted CUSUM statistic $\omega[k]\Gamma[m, k]$ is of the same order. A single ASE is required in the *offline* training phase to yield fixed edge probabilities estimates $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top$. No embeddings have to be recomputed each time a new graph is observed. To gain discriminative power in the statistical tests we perform, an alternative would be to examine the CUSUM statistic at every time point $t \in [m+1, \dots, m+k]$. This comes at the price of increased computational complexity, since it would entail computing k additional ASEs during the monitoring phase. This computational challenge is compounded

with the need to derive the limiting distribution of the resulting stochastic process.

5.3.3 Statistical analysis of the null distribution

In order to select the weighting and threshold functions, we will study the distribution of our statistic under the null hypothesis. We will first develop theory for the case when the ASE estimate is error-free, i.e., $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top = \mathbf{X}\mathbf{X}^\top = \mathbf{P}$. This way the estimated latent positions allow for a perfect reconstruction of the connection probability matrix. In practice, this will be valid when m and/or N are large enough. Since for some applications this may not be necessarily true, we will then extend the analysis for the imperfect estimation case.

5.3.3.1 Perfect ASE estimation

In this favorable case one has[1] $\mathbf{h} = \mathrm{vec}\left[\mathrm{triu}\left(\mathbf{P} - \mathbf{A}_t\right)\right]$, with $\mathbb{E}\left(\mathbf{h}\right) = \mathbf{0}$. The covariance matrix $\mathbf{\Sigma}_H = \mathbb{E}(\mathbf{h}\mathbf{h}^\top) \in \mathbb{R}^{r\times r}$ has null non-diagonal entries since the random variables A_{ij} are mutually independent. The diagonal entries are $\mathrm{var}\left[A_{ij}\right] = P_{ij}(1-P_{ij})$. In short, $\mathbf{\Sigma}_H$ is a diagonal matrix whose nonzero entries are $p_l(1-p_l)$, $l = 1, \ldots, r$, with p_l denoting the entries of $\mathrm{vec}\left[\mathrm{triu}\left(\mathbf{P}\right)\right]$ (i.e., a reindexing of P_{ij}).

Given this characterization of the first two moments of $\mathbf{h}$, the following proposition gives the asymptotic distribution of the CUSUM statistic $\Gamma[m,k]$ as $k \to \infty$. In practice, we rely on this limiting distribution as an approximation (for finite k) based on which we set the threshold $c_\alpha[k]$.

Proposition 5.3.1. *Suppose the perfect ASE estimation assumption* $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top = \mathbf{X}\mathbf{X}^\top = \mathbf{P}$ *holds. Then, as* $k \to \infty$ *the test statistic sequence converges in distribution, namely*

$$k^{-1}\Gamma[m,k] \xrightarrow{\mathcal{L}} \sum_{l=1}^{r} p_l(1-p_l)y_l^2, \tag{5.5}$$

where $\{y_l\}_{l=1}^r$ *are i.i.d. standard Gaussian random variables.*

Proof. Invoking the Central Limit Theoreom (CLT), as $k \to \infty$ the distribution of $k^{-1/2}\mathbf{s}[m,k]$ in (5.4) converges to a multivariate Gaussian distribution $\mathcal{N}\left(\mathbf{0}, \mathbf{\Sigma}_H\right)$, i.e., $k^{-1/2}\mathbf{s}[m,k] \xrightarrow{\mathcal{L}} (\mathbf{\Sigma}_H)^{1/2}\mathbf{y}$, where $\mathbf{y}$ is a standard Gaussian random vector. Hence, $k^{-1}\Gamma[m,k] = \|k^{-1/2}\mathbf{s}[m,k]\|_2^2$ also converges in distribution because

$$k^{-1}\Gamma[m,k] = (k^{-1/2}\mathbf{s}[m,k])^\top k^{-1/2}\mathbf{s}[m,k]$$

[1] We have omitted the dependence of $\mathbf{h}$ on t and $\hat{\mathbf{X}}$ for clarity.

$$
\begin{aligned}
&\xrightarrow{\mathcal{L}} (\mathbf{\Sigma}_H^{1/2}\mathbf{y})^\top \mathbf{\Sigma}_H^{1/2}\mathbf{y} \\
&= \mathbf{y}^\top \mathbf{\Sigma}_H \mathbf{y} \\
&= \sum_{l=1}^{r} p_l(1-p_l) y_l^2,
\end{aligned}
$$

which is the desired result in (5.5). □

Remark 5.3.3 (Convergence rate and network size). By bringing to bear Berry-Essen type results for the CLT, one can establish that the distribution of $k^{-1}\Gamma[m,k]$ converges to the limit stated in Proposition 5.3.1 at a rate $O(k^{-1/2})$, independent of r and hence the graph size N; see e.g., (Bentkus, 2003, Theorem 1.1).

Since $y_l \sim \mathcal{N}(0,1)$ then $y_l^2 \sim \chi^2(1)$ (chi-squared distribution with one degree of freedom). By virtue of Proposition 5.3.1 and for sufficiently large k, we can approximate the mean and variance of $\Gamma[m,k]$ as

$$
\begin{aligned}
\mathbb{E}\left[\Gamma[m,k]\right] &\approx k \sum_{l=1}^{r} p_l(1-p_l), \\
\operatorname{var}\left[\Gamma[m,k]\right] &\approx 2k^2 \sum_{l=1}^{r} p_l^2(1-p_l)^2,
\end{aligned} \tag{5.6}
$$

where we have used that the $\{y_l\}_{l=1}^r$ are mutually independent.

To control the growing variance of $\Gamma[m,k]$, the weighting function for the perfect ASE case can be chosen as $\omega[k] = (rk)^{-1}$. The threshold $c_\alpha[k]$ is thus selected as the $(1-\alpha)$-quantile of the limiting distribution in (5.5), which provides a type-I error of approximately α. Next, we show that in the presence of estimation errors the weighting function will have to be readjusted accordingly.

5.3.3.2 Imperfect ASE estimation.

In this case, we will write

$$
\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t = \mathbf{X}\mathbf{X}^\top - \mathbf{A}_t + \hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{X}\mathbf{X}^\top,
$$

where $\mathbf{X}$ is the true latent positions matrix (cf. $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$). Defining the estimation error $\mathbf{E} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{X}\mathbf{X}^\top$, then

$$
\mathbf{h}(\mathbf{A}_t, \hat{\mathbf{X}}) = \operatorname{vec}\left[\operatorname{triu}\left(\mathbf{X}\mathbf{X}^\top - \mathbf{A}_t\right)\right] + \mathbf{e}, \tag{5.7}
$$

where $\mathbf{e} = \mathrm{vec}\left[\mathrm{triu}\left(\mathbf{E}\right)\right] = [e_1, \ldots, e_r]^\top$. So the first term in (5.7) corresponds to a perfect ASE, while the second one captures the estimation error stemming from an imperfect reconstruction of $\mathbf{P}$. Note that after training, $\mathbf{e}$ is fixed and it does not depend on t.

Using (5.7) and by virtue of the CLT, it follows that for sufficiently large k the distribution of $\mathbf{s}[m,k]$ can be well approximated by the multivariate Gaussian $\mathcal{N}\left(k\mathbf{e}, k\boldsymbol{\Sigma}_H\right)$. Standard calculations for the norm of a non-centered Gaussian vector suffice to assert that the distribution of $\Gamma[m,k]$ can be in turn approximated by the distribution of the random variable

$$\bar{\Gamma} = k\sum_{l=1}^{r} p_l(1-p_l)\left(y_l + b_l\right)^2, \tag{5.8}$$

where $\{y_l\}_{l=1}^r$ is an independent sequence of standard Gaussian random variables and $\{b_l\}_{l=1}^r$ are the entries of vector $\mathbf{b} = \sqrt{k}\boldsymbol{\Sigma}_H^{-1/2}\mathbf{e}$. Note that if the ASE estimation is perfect, then $\mathbf{e} = \mathbf{0}$ and we recover the distribution in Proposition 5.3.1.

For large k, using (5.8) we can approximate the expected value and variance of $\Gamma[m,k]$ as in the error-free case. The difference here is that each summand $\left(y_l + b_l\right)^2 \sim \chi^2(1, b_l^2)$, i.e., a non-central chi-squared distribution with one degree of freedom and parameter $b_l^2 = \dfrac{k}{p_l(1-p_l)}e_l^2$. Hence, one finds

$$\begin{aligned}
\mathbb{E}\left(\Gamma[m,k]\right) &\approx k^2\|\mathbf{e}\|_2^2 + k\sum_{l=1}^{r} p_l(1-p_l) \\
&= k^2\|\mathbf{e}\|_2^2 + k\|\boldsymbol{\sigma}\|_1, \qquad (5.9) \\
\mathrm{var}\left[\Gamma[m,k]\right] &\approx 4k^3\sum_{l=1}^{r} p_l(1-p_l)e_l^2 + 2k^2\sum_{l=1}^{r} p_l^2(1-p_l)^2 \\
&= 4k^3\boldsymbol{\sigma}^\top\mathbf{e}^2 + 2k^2\|\boldsymbol{\sigma}\|_2^2, \qquad (5.10)
\end{aligned}$$

where for notational convenience we defined the auxiliary vector $\boldsymbol{\sigma}$ with entries $\{p_l(1-p_l)\}_{l=1}^r$, and $\mathbf{e}^2$ denotes the entry-wise square of $\mathbf{e}$. The preceding arguments suffice to establish the following result on the convergence of $\Gamma[m,k]$.

Proposition 5.3.2. *In the general case, as $k \to \infty$ the test statistic sequence converges in distribution, namely*

$$\frac{\Gamma[m,k] - k^2\|\mathbf{e}\|_2^2 - k\|\boldsymbol{\sigma}\|_1}{\sqrt{4k^3\boldsymbol{\sigma}^\top\mathbf{e}^2 + 2k^2\|\boldsymbol{\sigma}\|_2^2}} \xrightarrow{\mathcal{L}} y,$$

where y is a standard Gaussian random variable.

Apparently, we need to choose $\omega[k] = (rk^{3/2})^{-1}$ to control the variance of the weighted statistic. This is because for large k, the term that dominates the variance

expression (5.10) grows like k^3 [cf. k^2 in (5.6)]. The detection threshold $c_\alpha[k]$ is thus set as the $(1-\alpha)$-quantile of the generalized chi-squared distribution defined in (5.8), after weighting. We note that the resulting cumulative distribution function has a complex form which requires numerical integration to compute the desired quantiles; see also (Imhof, 1961; Munuswamy, 1963) for classic formulae to approximate said distribution function. As the next example shows, for particular cases the resulting distribution simplifies.

Example 5.3.1. For an ER model with connection probability p we have $p_l = p$ for all $l = 1, \ldots, r$ and (5.8) simplifies to

$$\bar{\Gamma}_{\text{ER}} = kp(1-p)u, \quad \text{with } u \sim \chi^2\left(r, \frac{k}{p(1-p)}\|\mathbf{e}\|_2^2\right). \tag{5.11}$$

Alternatively, for threshold selection we will often resort to the mean plus three standard deviations

$$\text{th}[k] := \omega[k]\mathbb{E}_{bc}[k] + 3\sqrt{\omega^2[k]\text{var}_{bc}[k]}, \tag{5.12}$$

where $\mathbb{E}_{bc}$ is the expectation of the statistic before the change and var_{bc} is its variance; given by (5.9) and (5.10), respectively, using a suitable estimate of $\mathbf{e}$ described in Section 5.3.5 Numerical tests in Section 5.4.1 corroborate that this rule of thumb works well for all practical CPD purposes and it comes close to the true 0.99-quantile. Moreover, having an analytic threshold expression facilitates studying the detection resolution of the online CPD procedure, the subject of the next section.

5.3.4 Change detectability analysis

Let us examine what changes are detectable by the proposed online CPD algorithm, when using the simple thresholding rule $\text{th}[k]$ based on the derived mean and variance of the statistics's null distribution. To this end, we will assume that from a certain change-point $k = k_c$ onward, the sequence of graphs is generated by an RDPG with latent vectors $\mathbf{Y}$ so that $\mathbf{\Delta} := \mathbf{X}\mathbf{X}^\top - \mathbf{Y}\mathbf{Y}^\top$ (i.e., the change is manifested through a perturbation on the resulting probability matrix). Given the expressive power of RDPGs, the modeling assumption for $k \geq k_c$ comes with limited loss of generality. Henceforth, let $\boldsymbol{\delta} := \text{vec}\left[\text{triu}\left(\mathbf{\Delta}\right)\right]$.

If we are at a certain time $k > k_c$, the partial sum of the monitoring function is

then (recall $\mathbf{E} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{X}\mathbf{X}^\top$)

$$\begin{aligned}\mathbf{s}[m,k] &= \sum_{t=m+1}^{m+k} \mathbf{h}\left(\mathbf{A}_t, \hat{\mathbf{X}}\right) \\ &= \sum_{t=m+1}^{m+k_c-1} \mathbf{h}\left(\mathbf{A}_t, \mathbf{X}\right) + \sum_{t=m+k_c}^{m+k} \mathbf{h}\left(\mathbf{A}_t, \mathbf{Y}\right) + k\mathbf{e} + (k-k_c)\boldsymbol{\delta}.\end{aligned}$$

Similar to the previous section, for large k_c and k we obtain a Gaussian vector with independent entries; mean $k\mathbf{e} + (k-k_c)\boldsymbol{\delta}$ and covariance matrix $k_c\text{diag}[\boldsymbol{\sigma}_X] + (k-k_c)\text{diag}[\boldsymbol{\sigma}_Y]$, where $\boldsymbol{\sigma}_X$ and $\boldsymbol{\sigma}_Y$ are the auxiliary vectors defined in (5.10) corresponding to $\mathbf{X}$ and $\mathbf{Y}$, respectively. This results in a CUSUM statistic with mean approximately equal to

$$\begin{aligned}\mathbb{E}\left(\Gamma[m,k]\right) \approx\ & \|k\mathbf{e} + (k-k_c)\boldsymbol{\delta}\|_2^2 \\ & + k_c\|\boldsymbol{\sigma}_X\|_1 + (k-k_c)\|\boldsymbol{\sigma}_Y\|_1. \end{aligned} \tag{5.13}$$

In the long run as $k \to \infty$, the dominant term will be the first one, which when weighted by $\omega[k] = (rk^{3/2})^{-1}$ amounts to $\omega[k]\mathbb{E}\left(\Gamma[m,k]\right) \approx k^{1/2}\left\|\mathbf{e}+\boldsymbol{\delta}\right\|_2^2/r$. Given that $\omega[k]\Gamma[m,k]$ has finite variance and that on this asymptotic regime $\text{th}[k] \approx k^{1/2}\|\mathbf{e}\|_2^2/r$ plus a constant, we have established that changes are detectable as long as

$$\|\mathbf{e}+\boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2 \ \Rightarrow 2\|\mathbf{e}\|_2\cos\theta + \|\boldsymbol{\delta}\|_2 > 0, \tag{5.14}$$

where θ is the angle between $\mathbf{e}$ and $\boldsymbol{\delta}$. It thus follows that a large value of $\|\boldsymbol{\delta}\|_2$ aids detectability, as expected. The same happens for small values of the estimation error magnitude $\|\mathbf{e}\|_2$, and in the idealized perfect estimation scenario we find all changes will be detected in the long run. Naturally, condition (5.14) is sufficient for changes to be detected, but not necessary. On the imperfect scenario, the resulting model estimation error will result in small changes likely going undetected provided $\theta \in (\frac{\pi}{2}, \frac{3\pi}{2})$. On top of this angular requirement, a change may be missed when the "perturbation-to-imperfection" ratio is small, i.e., $\frac{\|\boldsymbol{\delta}\|_2}{\|\mathbf{e}\|_2} < 2|\cos\theta|$.

The following simple example offers additional insights on the feasibility of the condition (5.14).

Example 5.3.2. Consider a sequence of ER graphs with connection probability p, which at a certain time-step k_c changes to $q = p - \Delta$. Then the bound

$$\mathbb{P}\left(\|\mathbf{e}+\boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2\right) \geq 1 - \frac{8(1-p)}{\Delta^2 N^2(N-1)m}\left[\frac{1-p}{Nm} + 2(N-1)p\right] \tag{5.15}$$

on the probability of satisifying the detectability condition (5.14) holds asymptotically in N. This means that if $\Delta^2 N^2 m$ goes to infinity as N grows, then the change will be detected with high probability. In other words, the method detects changes Δ up to an order of $N^{-1}m^{-1/2}$. This example further illustrates that Algorithm 4's performance improves with growing m (the size of the training set) as well as N (the number of nodes).

In order to prove the bound in (5.15) holds, first note that the equation $\|\mathbf{e}+\boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2$ in this case may be written as

$$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} E_{ij} > -\Delta\frac{N(N-1)}{2}, \tag{5.16}$$

where we have assumed that $\Delta > 0$ (the analysis that follows is readily extended to $\Delta < 0$). Recalling that in this case $\mathbf{E} = \hat{\mathbf{x}}\hat{\mathbf{x}}^\top - p\mathbf{1}_{N\times N}$ (with $\hat{\mathbf{x}} \in \mathbb{R}^{N\times 1}$), we rewrite (5.16) as

$$\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}} > \left(p - \frac{\Delta}{2}\right) N(N-1). \tag{5.17}$$

Since asymptotically (in N) $\hat{\mathbf{x}}$ is a normal vector with mean $\boldsymbol{\mu} = \sqrt{p}\mathbf{1}_{N\times 1}$ and covariance matrix $\boldsymbol{\Sigma} = \frac{(1-p)}{Nm}\mathbf{I}$ (Athreya et al., 2016; Bourgade et al., 2017; R. Tang et al., 2018), we consider this asymptotic regime and use results about the statistics of quadratic forms of Gaussian vectors (Rencher & Schaalje, 2008, Ch. 5). For instance, the resulting mean is

$$\begin{aligned}\mathbb{E}[\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}}] &= \operatorname{tr}\left[(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma}\right] + \boldsymbol{\mu}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\mu} \\ &= pN(N-1).\end{aligned}$$

Comparing the equation above to (5.17), it follows we have to bound the probability that $\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\mathbf{x}$ exceeds its mean minus $\Delta N(N-1)/2$. To this end we compute the variance of the quadratic form, which is (let $\sigma^2 := (1-p)/(Nm)$)

$$\begin{aligned}\operatorname{var}[\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}}] &= 2\operatorname{tr}\left[((\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma})^2\right] \\ &\quad + 4\boldsymbol{\mu}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma}(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\mu} \\ &= 2\sigma^2 N(N-1)(\sigma^2 + 2(N-1)p).\end{aligned}$$

Applying Chebyshev's inequality, the result follows.

5.3.5 Implementation details

We close this section with some necessary implementation details for Algorithm 4. These pertain to the calculation of the threshold and the possibility of utilizing windowed statistics as alternatives to the the cumulative sum (5.4).

5.3.5.1 Threshold calculation

The procedure outlined in Section 5.3.3 requires prior knowledge on the values of $\mathbf{P}$ and $\mathbf{e}$ in order to set the threshold $c_\alpha[k]$. This will be the case if one uses the exact $(1-\alpha)$-quantile of the null distribution, approximate formulae, or, simply $\mathrm{th}[k]$ in (5.12). In most applications the values of $\mathbf{P}$ and $\mathbf{e}$ are unknown, so it is necessary to estimate them from the observations in the training set.

For $\mathbf{P}$ we simply use the plugin estimator $\hat{\mathbf{P}} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top$, i.e., we estimate $\mathbf{P}$ using the ASE of $\bar{\mathbf{A}}$ in (5.2), computed over the training set. Characterization of the statistical properties of $\mathbf{E}$ (and subsequently $\mathbf{e}$) is challenging in general. Even for the simple ER model, the study of $\mathbf{E}$ is non-trivial as shown in Example 5.3.2. Therefore, we opted for a data-driven approach to form point estimates of $\mathbf{E}$ by performing "leave-one-out" passes over the training set: we randomly select an index j in $1, \ldots, m$ and compute the ASE of $\mathbf{A}[j]$ and of

$$\bar{\mathbf{A}}_{(-j)} = \frac{1}{m-1} \sum_{\substack{t=1 \\ t \neq j}}^{m} \mathbf{A}_t,$$

the mean adjacency matrix over the left-out samples. We denote these ASEs as $\hat{\mathbf{X}}_j$ and $\overline{\mathbf{X}}_{(-j)}$, respectively. Because $\mathrm{var}\left[\overline{\mathbf{X}}_{(-j)}\overline{\mathbf{X}}_{(-j)}^\top - \mathbf{P}\right] = \mathrm{var}\left[\hat{\mathbf{X}}_j\hat{\mathbf{X}}_j^\top - \mathbf{P}\right]/(m-1)$ as discussed in Remark 5.3.1 and (R. Tang et al., 2018), we compute

$$\mathbf{E}_j = \frac{\hat{\mathbf{X}}_j\hat{\mathbf{X}}_j^\top - \overline{\mathbf{X}}_{(-j)}\overline{\mathbf{X}}_{(-j)}^\top}{\sqrt{m-1}},$$

a fixed number of times, obtain a set of values $\mathbf{E}_j$, and estimate a "worst-case" $\hat{\mathbf{E}}$ via the 0.99-quantile of this set.

Note that the change detectability of the algorithm depends on the value of $\hat{\mathbf{e}}$ and how close it is to $\mathbf{e}$. In particular, the relevant condition (5.14) in practice becomes $\|\hat{\mathbf{e}}\|^2 < \|\mathbf{e} + \boldsymbol{\delta}\|_2^2$.

5.3.5.2 Finite memory statistics

The CUSUM statistic $\Gamma[m,k] = \|\mathbf{s}[m,k]\|_2^2$ we have dealt with so far is based on the partial sum $\mathbf{s}[m,k] = \sum_{t=m+1}^{m+k} \mathbf{h}(\mathbf{A}_t, \hat{\mathbf{X}})$. As discussed in Remark 5.3.2, it can

be computed in a recursive and memory-efficient fashion that is ideal for online operation. Moreover, such an infinite-memory statistic accrues information from the entire data stream $\{\mathbf{A}_{m+k}\}_{k\geq 1}$, which is beneficial when it comes to invoking asymptotic approximations to the null distribution as in Section 5.3.3. However, if the change point k_c occurs rather late during the monitoring horizon, then the inertia effect induced by a lengthy history of nominal graph observations will translate to longer detection delays.

To attain faster reaction times one can resort to alternative finite memory statistics, which tend to rely on a judicious subset of the most recent observations. One natural variant is to adopt a fixed-length sliding window statistic, where the partial sum is $\mathbf{s}[k-L,k]$ for given window length L. At time k, this moving sum (MOSUM) statistic discards past data in the interval $(m, k-L)$, and its computation requires storing the last L graphs in the sequence; see also (5.21) and (Kirch & Weber, 2018) for a modified version where the window length grows proportionally with k. Another useful procedure stems from the exponentially-weighted sum (EWSUM) statistic, namely

$$\mathbf{s}_\beta[m,k] = \sum_{t=m+1}^{m+k} \beta^{m+k-t} \mathbf{h}\left(\mathbf{A}_t, \hat{\mathbf{X}}\right), \tag{5.18}$$

where $\beta \in (0,1]$ is a so-termed forgetting factor. EWSUM coincides with CUSUM for $\beta = 1$, whereas for $\beta < 1$ past samples are exponentially down-weighted and thus it offers a faster response to changes. Similar to CUSUM, (5.18) can be recursively updated as $\mathbf{s}_\beta[m,k] = \beta \mathbf{s}_\beta[m,k-1] + \mathbf{h}(\mathbf{A}[k], \hat{\mathbf{X}})$ and does not require storing any of the past measurements. Notice that as long as the window length is long enough we may still use the results derived in Section 5.3.3, and the only algorithmic difference is that the weight $\omega[k]$ and the threshold $c_\alpha[k]$ should be changed accordingly (e.g., $\omega[k] = (r \min\{k, L\}^{3/2})^{-1}$ in the MOSUM case). The effect of choosing different windowed statistics is studied in the numerical tests of Section 5.4.

5.3.6 Handling weighted and directed networks

Let us briefly discuss how to perform online CPD for the general weighted and/or directed case. Extending the results presented in Section 5.3 to digraphs is straightforward. The only noteworthy difference is that, since the adjacency matrices are no longer symmetric, we need to consider entries from the entire residual matrix $\mathbf{H}$ (except the diagonal) during online monitoring, instead of the upper triangular half in (5.3).

The path forward in the weighted case is also clear. The important difference is that the variance of each A_{ij} is no longer of the form $p_{ij}(1-p_{ij})$, because we are naturally allowing for non-Bernoulli edge weight distributions. Following the WRDPG

model we introduced in Chapter 4, we have $\text{var}\,[A_{ij}] = \mathbf{x}_i^\top[2]\mathbf{x}_j[2] - (\mathbf{x}_i^\top[1]\mathbf{x}_j[1])^2$. We rely on plugin variance estimates using the corresponding ASEs to compute the thresholds for the numerical test cases that follow [cf. vector $\boldsymbol{\sigma}$ in (5.9) and (5.10)]. One can seamlessly blend the ideas in Chapter 4 to perform online CPD for weighted digraphs. The provided code offers this functionality.

In closing, note that the aforementioned discussion is pertinent only when the goal is to detect changes in the mean adjacency matrix (i.e., $l = 1$). This is the scope of the ensuing numerical experiments. Considering larger values of l could be prudent when interested in more fine-grained changes on the weights' distribution.

5.4 Numerical Experiments

Here we carry out numerical experiments to evaluate the performance of the proposed online CPD algorithm for weighted and (un)directed graph sequences. We start with a controlled synthetic data setting, where the goal is to identify emergent network community structure (Section 5.4.1). We carefully examine: (i) the choice of the detection threshold and monitoring function; (ii) the choice of the running statistic and its effect on the detection delay; (iii) robustness to the prescribed false alarm rate α; and (iv) comparisons with relevant batch and online CPD methods. Test cases with real wireless and social network data are presented in Section 5.4.2. For the implementations we used the Python libraries `NumPy` (Harris et al., 2020), `NetworkX` (Hagberg et al., 2008), `pandas` (McKinney, 2010), `graspologic` (Chung et al., 2019), as well as our own code which we share in https://github.com/git-artes/cpd_rdpg. For the comparison with the online CPD method in Chen (2019), we used the official `R` implementation in the `gStream` package with the default parameters settings. Furthermore, as a baseline we have implemented the offline CPD algorithm described in Madrid Padilla et al. (2022). This implementation is also available in our GitHub repository.

5.4.1 Simulated data

A timely problem is to detect when communities arise in networks. So, we first test the proposed online CPD method by generating a sequence of 150 ER graphs with $N = 100$ nodes and connection probability $p = 0.5$. After $t_c = 150$, the model shifts to a two-block SBM with $N/2 = 50$ nodes in each community and connection probability $q_1 = 0.6$ for nodes in the same community and $q_2 = 0.4$ for nodes in different blocks. We use the first $m = 50$ graphs as the training set, and the value of d is automatically chosen via the algorithm by M. Zhu and Ghodsi (2006). Because the index k in $\Gamma[m, k]$ measures how much time has elapsed since monitoring started,

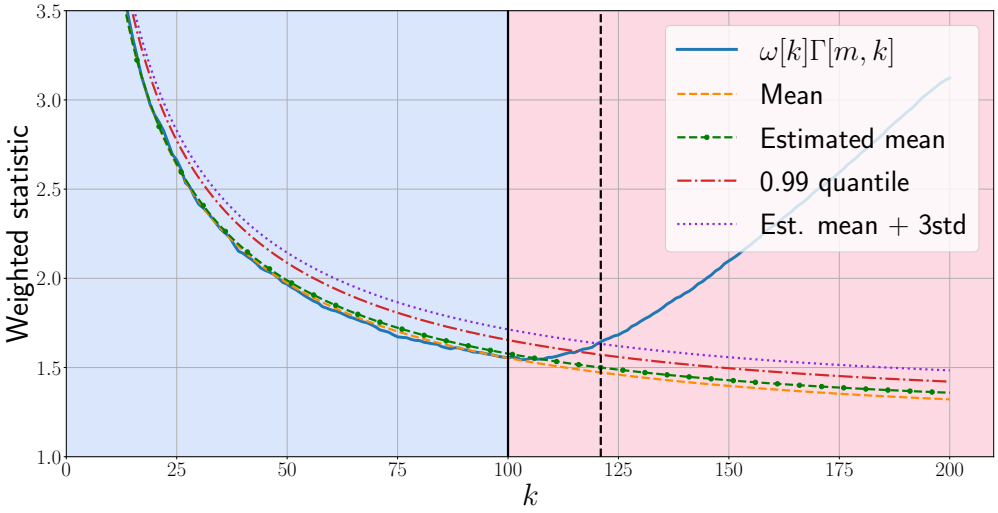

Figure 5.1: Evolution of $\omega[k]\Gamma[m,k]$, its mean and the estimated mean, for simulated data. Two thresholds are shown: the 0.99-quantile of the distribution in (5.11) and three standard deviations away from the mean; those thresholds are very close and the latter is preferred due to its reduced complexity. The solid vertical line indicates the actual change-point, while the dashed one is the detection. A change in background color indicates a change-point detected by the offline algorithm (Madrid Padilla et al., 2022). Our approach is able to detect the change with a relatively small delay, while operating in an online fashion.

the change-point is at $k_c = 100$.

Figure 5.1 depicts the results for this test case. We show two thresholds: the 0.99 quantile of the estimated distribution [i.e., the distribution given by (5.11) but with $\hat{\mathbf{e}}$ instead of $\mathbf{e}$] and $\text{th}[k]$, the estimated mean plus three standard deviations. Apparently, the difference between those two thresholds is small, so $\text{th}[k]$ is preferred due to its reduced complexity. Using that threshold a change-point is declared at $k^* = 121$, so our algorithm is successfully identifying the change in the model. The detection delay can be explained if we look at the estimated mean of the weighted CUSUM statistic. Since we are estimating the error $\mathbf{E}$ as the 0.99-quantile over the training set, we always overestimate the true value. Also, since we are monitoring the cumulative sum (5.4), if a change occurs after a long period of time then the drift in $\Gamma[m,k]$ will not be noticed immediately; see also the discussion in Section 5.3.5. As a way to compare the performance of Algorithm 4 with other approaches, Figure 5.1 also shows the detection result for the offline baseline proposed in Madrid Padilla et al. (2022). That algorithm detects the change with no delay, but it has a markedly greater computational complexity than ours and examines the entire data sequence as a batch.

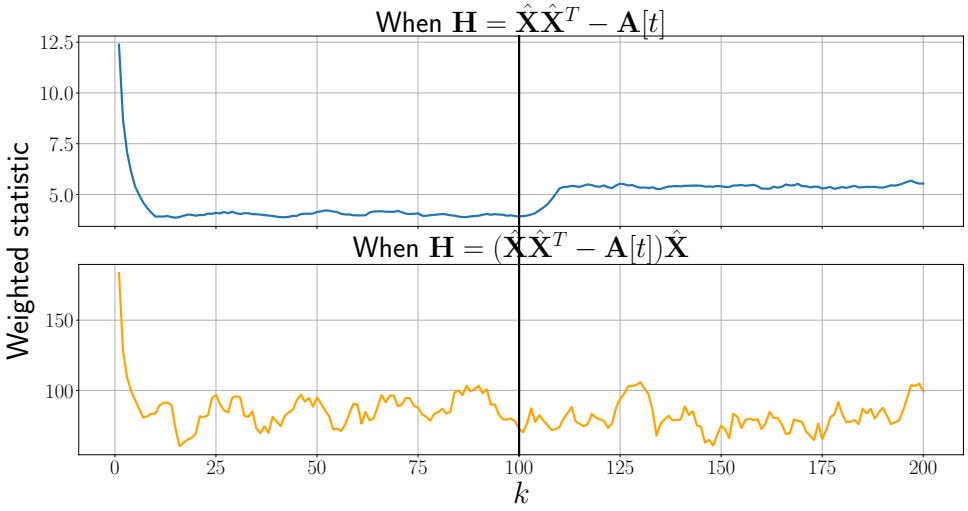

Figure 5.2: Evolution of $\omega[k]\Gamma[m,k]$ for residual (top) and projection (bottom) monitoring functions, using the MOSUM sliding window statistic. After the change-point there is a discernible change in trend for the residual; the projection does not exhibit such desirable behavior.

5.4.1.1 On the choice of the monitoring function

In this running example, had we used the monitoring function $\mathbf{H}' = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}_t\right)\hat{\mathbf{X}}$ (i.e., use the projection instead of the residual) we would have missed the change altogether. Indeed, for perfect ASE estimation, if our training data adheres to an ER model with parameter p then $\hat{\mathbf{X}} = \sqrt{p}\mathbf{1}_{N\times d}$ and $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top = p\mathbf{1}_N$. Now suppose there is a change in the nominal model and we shift to a two-block SBM, where each community has $N/2$ nodes and the connection probabilities are q_1 for nodes in the same block and q_2 for nodes in different communities. The connection probability matrix for said SBM is

$$\mathbf{P}_{\text{SBM}} = \left(\begin{array}{c|c} \mathbf{Q}_1 & \mathbf{Q}_2 \\ \hline \mathbf{Q}_2 & \mathbf{Q}_1 \end{array}\right), \tag{5.19}$$

where $\mathbf{Q}_1 = q_1(\mathbf{1}_{N/2} - \mathbf{I}_{N/2})$ and $\mathbf{Q}_2 = q_2\mathbf{1}_{N/2}$. After the change we thus have $\mathbb{E}(\mathbf{H}') = (p\mathbf{1}_N - \mathbf{P}_{\text{SBM}})\sqrt{p}\mathbf{1}_{N\times d}$. Since each row of $\mathbf{P}_{\text{SBM}}$ has $N/2-1$ entries with value q_1 and $N/2$ entries with value q_2, each entry of $\mathbb{E}(\mathbf{H}')$ is given by

$$\begin{aligned}(\mathbb{E}(\mathbf{H}'))_{ij} &= \left(\frac{N}{2} - 1\right)\sqrt{p}(p - q_1) + \frac{N}{2}\sqrt{p}(p - q_2) \\ &\approx N\sqrt{p}\left(p - \frac{q_1 + q_2}{2}\right),\end{aligned}$$

for large N. Accordingly, choosing p, q_1 and q_2 such that $q_1 + q_2 = 2p$ (as was the case for our simulation), we find that $\mathbb{E}(\mathbf{H}') = \mathbf{0}$, i.e., we do not expect to see a drift in the monitoring function after the change.

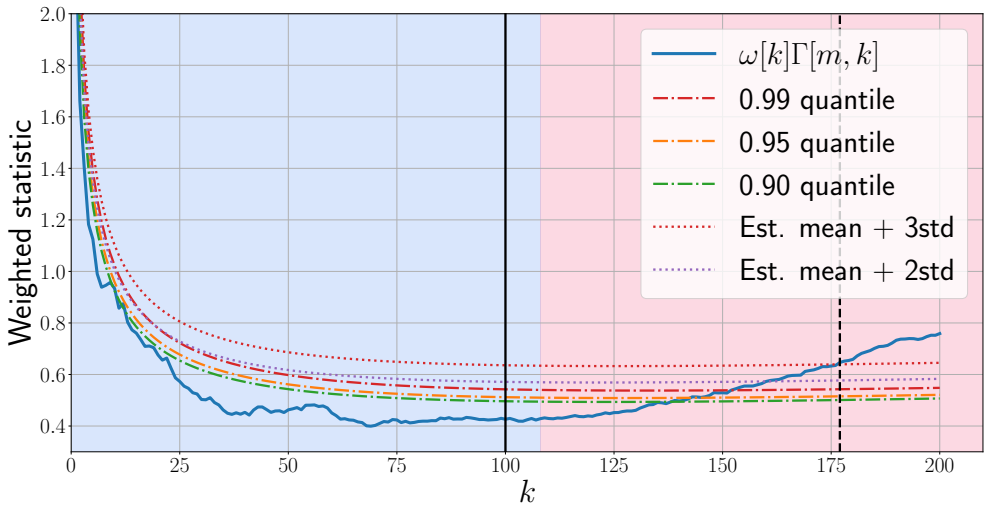

Figure 5.3: Evolution of $\omega[k]\Gamma[m,k]$ and five possible thresholds: $c_\alpha[k]$ (for $1-\alpha \in \{0.9, 0.95, 0.99\}$) and th$[k]$ equal to the mean plus two and three standard deviations. The setting is the same as in Fig. 5.1 except that $N = 20$ to increase the variance of $\omega[k]\Gamma[m,k]$. Using $1-\alpha = 0.99$ is preferred as it provides more robustness to false positives. Both choices of th$[k]$ are reasonable, although using three standard deviations is consistently above $c_{0.01}[k]$ (see the first time-steps).

Figure 5.2 shows the evolution of the weighted statistic $\omega[k]\Gamma[m,k]$ for both choices of the monitoring function. The setup is the same as in the previous test case, with a change-point located at $k_c = 100$. The MOSUM statistic is adopted here, using a sliding window of length $L = 10$. When the residual $\mathbf{H}$ is chosen as the monitoring function, a sudden shift in trend is observed after the change-point. However, when the projection $\mathbf{H}'$ is used the statistic does not exhibit such desirable behaviour and misses the model change.

5.4.1.2 On the sensitivity to α

Here we examine the robustness of Algorithm 4 to the choice of the false alarm rate α. We simulate the same scenario as before, except that $N = 20$ in order to increase the variance of $\omega[k]\Gamma[m,k]$ and the error $\mathbf{E}$. As thresholds we test $c_\alpha[k]$ for $1-\alpha \in \{0.99, 0.95, 0.9\}$, along with two versions of th$[k]$: the mean plus two and three times the standard deviations as in (5.12).

The results are depicted in Figure 5.3. The example illustrates how using $1-\alpha = 0.95$ or $1-\alpha = 0.9$ may prove too conservative. In this particular instance, $1-\alpha = 0.9$ would result in a (false) change-point detected at $k \approx 10$. Furthermore, both versions of th$[k]$ provide reasonable results, although the one that uses three standard deviations is consistently above $c_{0.01}[k]$ and is thus preferred. We will re-examine this choice in Section 5.4.2, when we present real-world examples.

5.4.1.3 Comparison with (Chen, 2019)

An online CPD algorithm based on a k-nearest neighbor approach was proposed in Chen (2019). Observations are viewed as points in a normed space and the distance induced by such norm is used to define a neighborhood for each observation. Changes are detected by performing two-sample testing on the neighborhood graph. The proposed approach is computationally intensive because it requires that, if the current observation index is n, a two-sample hypothesis test is performed for each time $t \in \{1, \dots, n-1\}$ (or for a subset of these time instants). Also, it is memory-inefficient since one has to store the pairwise distances between all past observations. Even if these aspects are not a concern, this approach is ill-suited to detect changes in some sequences of networks, as we will argue shortly.

An example in Chen (2019) illustrates the performance of the CPD algorithm on network sequences. Observations are the adjacency matrices of the graphs and neighborhoods are defined using the distance induced by the Frobenius norm over such matrices. We will see that this distance does not allow for capturing some changes in the network connectivity, such as the formation of two communities discussed so far. Indeed, if $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{N \times N}$ are adjacency matrices of two ER graphs with connection probability p, then

$$\mathbb{E}\left(\|\mathbf{A} - \mathbf{B}\|_F^2\right) = N(N-1)2p(1-p),$$

since all entries $A_{ij}, B_{ij} \sim \text{Bernoulli}(p)$. Thus $\|\mathbf{A}-\mathbf{B}\|_F^2 \approx 2p(1-p)N^2$ for sufficiently large N. Suppose now that $\mathbf{C}$ and $\mathbf{D}$ are two adjacency matrices from a two-block SBM, where each community has $N/2$ nodes and the connection probabilities are q_1 for nodes in the same cluster and q_2 for nodes in different communities. Then the connection probability matrix for $\mathbf{C}$ and $\mathbf{D}$ is given by (5.19), so those matrices have $N^2/2$ entries whose expected value is q_2 and $(N/2-1)N \approx N^2$ entries whose expected value is q_1. All in all, similarly to the ER case we have

$$\begin{aligned}
\|\mathbf{C} - \mathbf{D}\|_F^2 &\approx N^2\left(q_1 - q_1^2 + q_2 - q_2^2\right), \\
\|\mathbf{A} - \mathbf{C}\|_F^2 &\approx N^2\left(p - p(q_1+q_2) + \frac{q_1+q_2}{2}\right).
\end{aligned}$$

Again, if we choose q_1 and q_2 such that $q_1 + q_2 = 2p$, then we obtain $\|\mathbf{A} - \mathbf{B}\|_F^2 \approx \|\mathbf{A} - \mathbf{C}\|_F^2$. In other words, the distance between an observation before the change ($\mathbf{A}$) and an observation after the change ($\mathbf{C}$) will be very similar to the distance between two observed matrices before the change ($\mathbf{A}$ and $\mathbf{B}$). For matrices after the change, we have that when $q_1 + q_2 = 2p$ then

$$\|\mathbf{C} - \mathbf{D}\|_F^2 \approx 2N^2\left(p - p^2 - (p - q_1)^2\right),$$

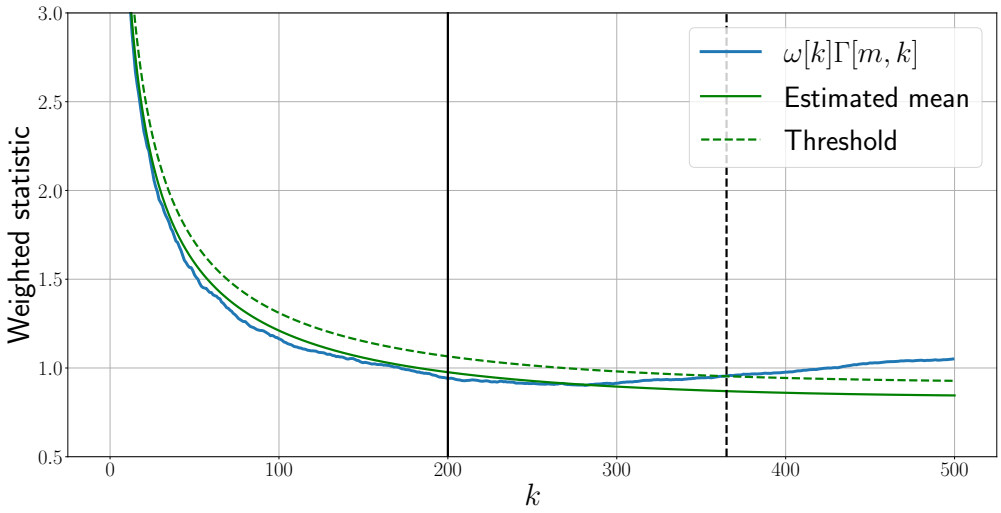

Figure 5.4: Detection result for a network transitioning from an ER model with $p = 0.3$ to a two-block SBM with $q_1 = 0.275$ and $q_2 = 0.325$. Algorithm 4 is able to detect the change in this setup, while the approach proposed in Chen (2019) fails to do so.

so choosing p and q_1 to be very similar (but not equal, so there is effectively a change), for large N these two models will be indistinguishable under the Frobenius distance criterion.

We simulated such a setup, with a network of $N = 100$ nodes switching from an ER model with $p = 0.3$ to a two-block SBM with $q_1 = 0.275$ and $q_2 = 0.325$. The change-point was located at $k_c = 200$. We ran the algorithm proposed in Chen (2019) using the implementation in the `R` package `gStream`. Selecting between 3 and 10 nearest neighbors and an average run length of 1000, it found no change-points in the data. Results for our CUSUM detector are depicted in Figure 5.4. Apparently, there is a noticable change in trend in the weighted statistic after $k = 300$, with a change-point being detected at $k^* = 365$. This arguably large detection delay can be shortened using a finite memory statistic such as MOSUM.

5.4.1.4 Detection delay

Characterizing the distribution of the detection delay τ (i.e., the time interval between the occurrence of a change and it actually being detected) is in general challenging. Instead, we will settle with a point estimate obtained via identification of the first instant the weighted statistic $\omega[k]\Gamma[m,k]$ crosses the threshold function $c_\alpha[k]$. Recall that this is the condition that defines the rejection region of our test. Since that statistic has finite variance, it is possible to predict at which time point k^* the change will be detected by studying when the expectation of the weighted test statistic after the change [cf. (5.13)] first exceeds the threshold. Once more, for simplicity and analytical tractability we will henceforth assume the threshold is set as $\text{th}[k]$ in (5.12). This choice (approximately) corresponds to $\alpha = 0.01$; see

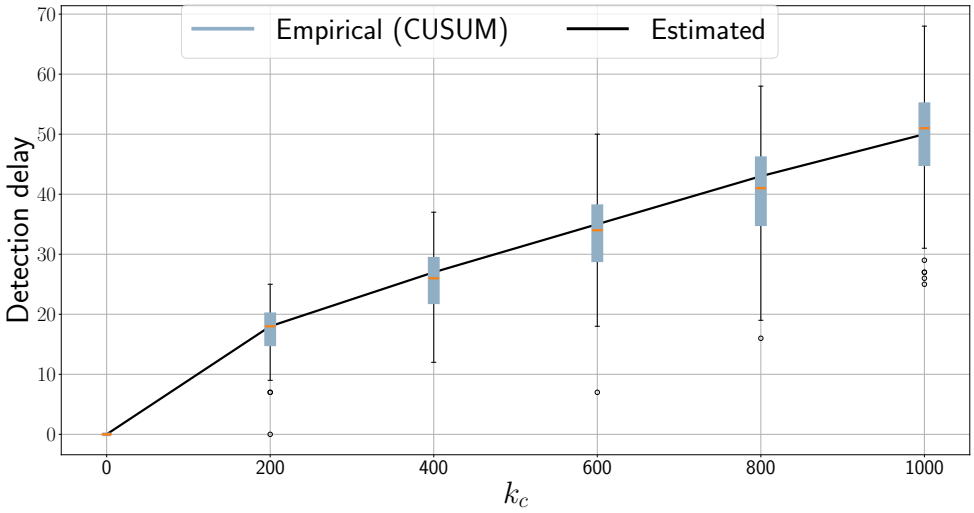

Figure 5.5: Estimated detection delay and empirical delays for different change-point locations k_c. Empirical delay is well predicted by the estimated curve. For the adopted CUSUM statistic, as expected the delay grows with k_c.

Figures 5.1 and 5.3 for further discussion on this point. To estimate the delay, we find the first instant $k^* \geq k_c$ for which $\omega[k^*]\mathbb{E}_{ac}[k^*] \geq \text{th}[k^*]$, where $\mathbb{E}_{ac}$ denotes the expectation of $\Gamma[m, k]$ after the change that is approximately given by (5.13). This amounts to solving the equation

$$\begin{aligned}(k^* - k_c)^2 \left(\|\boldsymbol{\sigma}_Y\|_1 - \|\boldsymbol{\sigma}_X\|_1 + 2k^*(\mathbf{e}^\top \boldsymbol{\delta}) + (k^* - k_c)||\boldsymbol{\delta}||_2^2\right)^2 \\ = 9\left(2(k^*)^2\|\boldsymbol{\sigma}_X\|_2^2 + 4(k^*)^3(\boldsymbol{\sigma}_X^\top \mathbf{e}^2)\right),\end{aligned} \tag{5.20}$$

which entails finding the roots of a fourth-order polynomial. The solution k^* can be obtained numerically, and the estimated delay becomes $\tau = k^* - k_c$.

To test said method, we simulated a sequence of ER networks with $N = 100$ nodes and connection probability $p = 0.5$. We use the first $m = 100$ graphs for training. The first k_c graphs after training follow that same model, but then observations shift to an ER with $p = 0.6$. The solution to (5.20) allows us to estimate the detection delay for different values of k_c. This can be done after training, since once that phase ends the error $\mathbf{e}$ is fixed, and vectors $\boldsymbol{\sigma}_X$, $\boldsymbol{\sigma}_Y$ and $\boldsymbol{\delta}$ are defined by the change in the underlying model. Figure 5.5 shows the estimated delay for $k_c \in \{0, 200, 400, 600, 800, 1000\}$. For each k_c a box plot of Algorithm 4's empirical delays is also shown, computed for 100 simulated runs using the CUSUM statistic. Our estimation is consistent with the experimental delays in Algorithm 4, which tend to show a linear growth with k_c. Figure 5.6 depicts the empirical delays in this setup for three different statistics: CUSUM, MOSUM and mMOSUM. This last

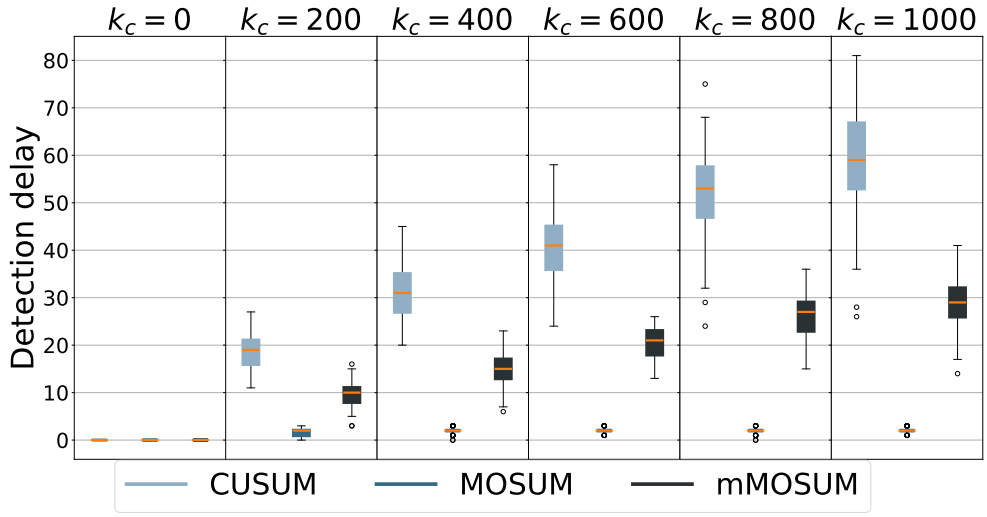

Figure 5.6: Empirical delays for different change-point locations k_c, using the CUSUM, MOSUM (with $L = 10$), and mMOSUM (with $h = 0.4$) statistics. Delays behave as expected given the different effective observation intervals: roughly constant delay for MOSUM, growing delays with k_c for both CUSUM and mMOSUM, but at a slower rate for the latter.

running statistic is defined in Kirch and Weber (2018) as

$$\mathbf{s}[m, k] = \sum_{t=m+\lfloor kh \rfloor + 1}^{m+k} \mathbf{h}\left(\mathbf{A}_t, \hat{\mathbf{X}}\right), \tag{5.21}$$

where $h \in (0, 1)$ and $\lfloor x \rfloor$ is the floor function, i.e., the largest integer that is smaller or equal to x. The mMOSUM is defined in a way such that early observations are discarded and the window length grows proportionally with k. Hence, the algorithm's response time should be faster than when using the CUSUM statistic. That is consistent with Figure 5.6, which shows that the detection delay for the mMOSUM statistic grows with k_c, but at a slower rate than that of CUSUM. For this simulation we set $h = 0.4$. The MOSUM statistic, with a window length of $L = 10$ observations, attains the shortest delay among the three and it is roughly constant with k_c. This is expected given that the window size remains constant for MOSUM, so there is no inertia associated with the change-point occurring long after monitoring started.

Finally, Figure 5.7 shows the empirical and estimated delays for the CUSUM statistic for various training set sizes m. The setup is similar to that of the previous test case, with an ER model switching from $p = 0.5$ to $p = 0.6$ at $k_c = 100$. As expected, the delay decreases with m, since more training samples lead to more accurate ASE estimates. Also, it is important to note that Algorithm 4 performs well with a relatively small training set size. In this setting, we observe there is no significant improvement beyond $m = 25$ (with the expected delay going from $\tau = 13$

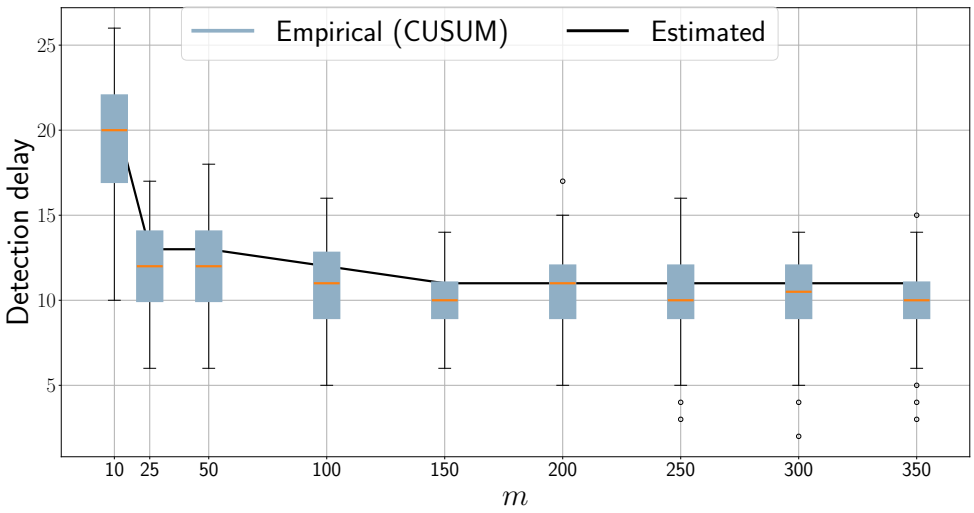

Figure 5.7: Estimated detection delay and empirical delays for different training set sizes m, for the CUSUM statistic. The delay is lower as m increases, but there is no significant improvement after $m = 25$.

to $\tau = 11$ for $m = 300$).

5.4.2 Real data experiments

5.4.2.1 Wireless network data

Received Signal Strength Indicator (RSSI) measurements between Wi-Fi access points (APs) in a Uruguayan school are obtained from the dataset described in Capdehourat et al. (2020). In this particular example we considered a network consisting of $N = 6$ APs, with measurements collected hourly during almost four weeks, spanning from 10/17/2018 to 11/13/2018 (corresponding to $T = 655$ graphs). The AP corresponding to node 4 was moved on 10/30/2018. As RSSI is measured in dBm (and are negative), we have first added an offset of 91 to all weights so that they become positive (as -90 dBm is the smallest RSSI measurement in this case) and that larger values still mean "stronger" edges. We thus have a directed (as power measurements between APs are not necessarily symmetric) and weighted graph sequence.

We used two days worth of measurements for training ($m = 48$) beginning on 10/12/2018. The resulting MOSUM statistic, the estimated mean and the resulting threshold th$[k]$ are shown in Figure 5.8 (top). Note how Algorithm 4 rapidly detects the AP movement. The offline CPD baseline in Madrid Padilla et al. (2022) is also able to detect the change, at around the same date. Furthermore, we complement the threshold studies carried out in Section 5.4.1 and compare the same two versions of th$[k]$, namely the estimated mean plus two or three standard deviations. Note how the change-point is detected around the same instant regardless of the specific

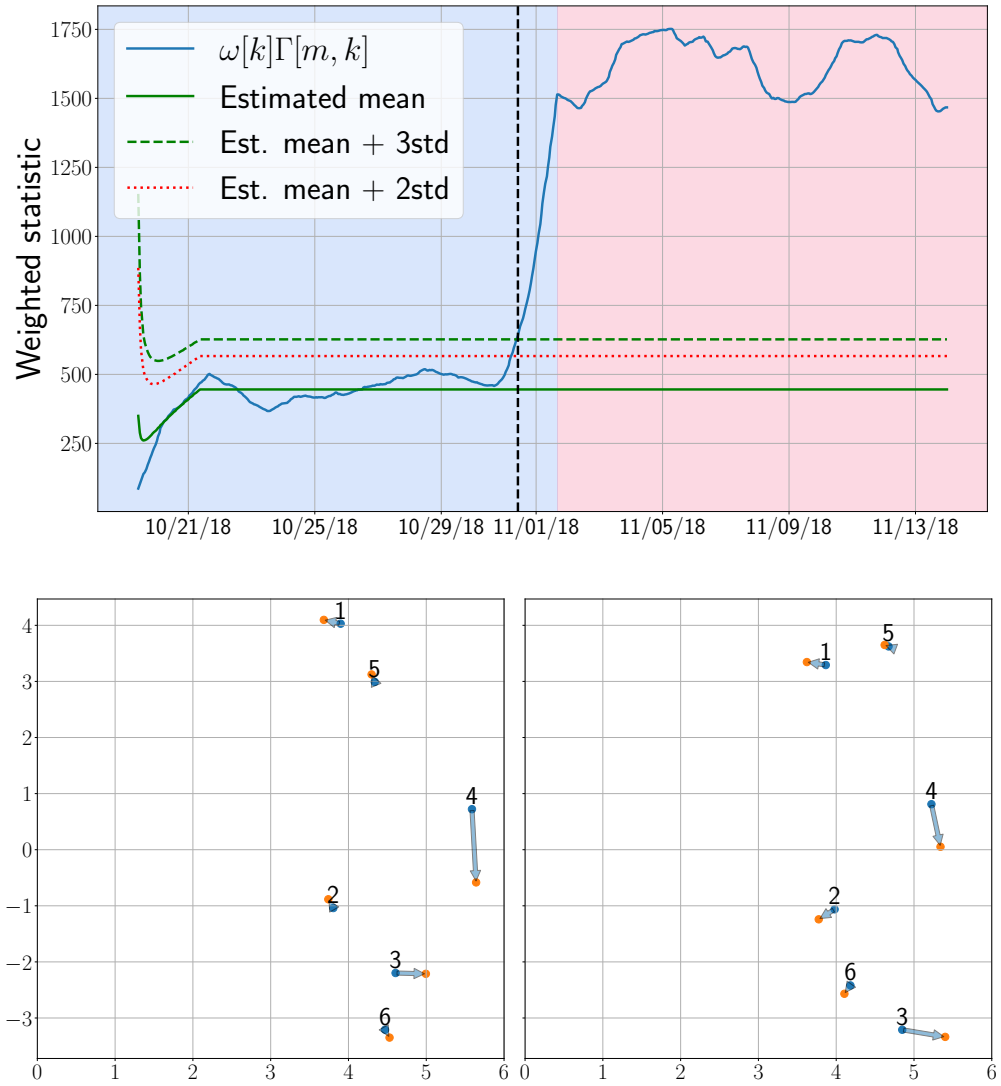

Figure 5.8: Online CPD for the RSSI dataset. Top: MOSUM statistic. A change in background color indicates a change-point detected by the offline algorithm Madrid Padilla et al. (2022). The dashed vertical line shows the detected change-point for the online algorithm. Algorithm 4 successfully detects that an AP was moved. Bottom, left: $\hat{\mathbf{X}}_1^l$ (blue) and $\hat{\mathbf{X}}_2^l$ (orange) latent vectors for $d = 2$ corresponding to $\bar{\mathbf{A}}_1$ and $\bar{\mathbf{A}}_2$ respectively. Vectors corresponding to the same node are joined by an arrow. Bottom, right: Id. but with $\hat{\mathbf{X}}_1^r$ (blue) and $\hat{\mathbf{X}}_2^r$ (orange). Node 4 corresponds to the AP that was moved, which together with node 3 are the ones whose embeddings change more prominently.

choice.

In addition to CPD, a valuable feature of RDPGs and its variants is their interpretability. To illustrate this attribute, let us consider two averaged adjacency matrices: those corresponding to the historic dataset and the last two days of the observation period. Let us denote the resulting matrices as $\bar{\mathbf{A}}_1$ and $\bar{\mathbf{A}}_2$, respectively, and analyze the resulting latent positions. In order to avoid the rotation ambiguities, we have used the so-called omnibus embedding (Levin et al., 2017), which in

this case amounts to performing ASE to

$$\mathbf{M} = \begin{pmatrix} \bar{\mathbf{A}}_1 & (\bar{\mathbf{A}}_1 + \bar{\mathbf{A}}_2)/2 \\ (\bar{\mathbf{A}}_1 + \bar{\mathbf{A}}_2)/2 & \bar{\mathbf{A}}_2 \end{pmatrix}.$$

This approach is only practical when jointly embedding a few adjacency matrices (two here), as the size of $\mathbf{M}$ increases rapidly with the number of matrices considered.

Nodal vectors ($d = 2$) are depicted in Figure 5.8 (bottom), where an arrow shows the changes between the embeddings of $\bar{\mathbf{A}}_1$ and $\bar{\mathbf{A}}_2$. Notice how the largest changes correspond to nodes 3 and 4. The scaling ambiguity we discussed in Remark 2.2.1 obscures which of the two APs was actually moved. Still, this monitoring tool would be valuable to network administrations as it identifies changes in a timely fashion and it provides a curated list of potentially problematic APs.

5.4.2.2 South American football matches

Consider a dynamic football network, whose $N = 10$ nodes are the national teams affiliated to CONMEBOL (which associates all South American countries except Guyana and Suriname). This is the oldest continental confederation under FIFA, and its teams have a long history going back to 1901. We consider yearly matches since 1940, when all national associations were founded and most have joined CONMEBOL (Venezuela joined in 1952).

The resulting undirected graphs have edge weights indicating the number of matches played between the two incident national teams during a particular year (Li & Mateos, 2022). We used the first $m = 20$ years for training and the evolution of the resulting weighted CUSUM statistic is shown in Figure 5.9 (top).

A change-point is detected around 1990 both by the online and offline CPD algorithms. Indeed, CONMEBOL's flagship tournament (*Copa América*) went through a period of intermittency that would last until 1987, when it started being organized regularly every two years with a nation hosting the event. This is apparent from the resulting embeddings in Figure 5.9 (bottom), where northern countries increase their corresponding magnitudes (indicating more frequent matches) and form a relatively tight community. On the other hand, southern countries form another (more loose) community, which approached the northern's one in recent years. Furthermore, this community's structure changed, where e.g., the historic Argentina-Uruguay match is now not as significant. We also examine the robustness of the results with respect to the choice of the threshold. Notice that both versions of th$[h]$ we implemented again detect a change-point roughly around the same time (one year difference in Figure 5.9). But as mentioned in Section 5.4.1, using the mean plus three standard deviations clearly provides more robustness to false positives, particularly in high

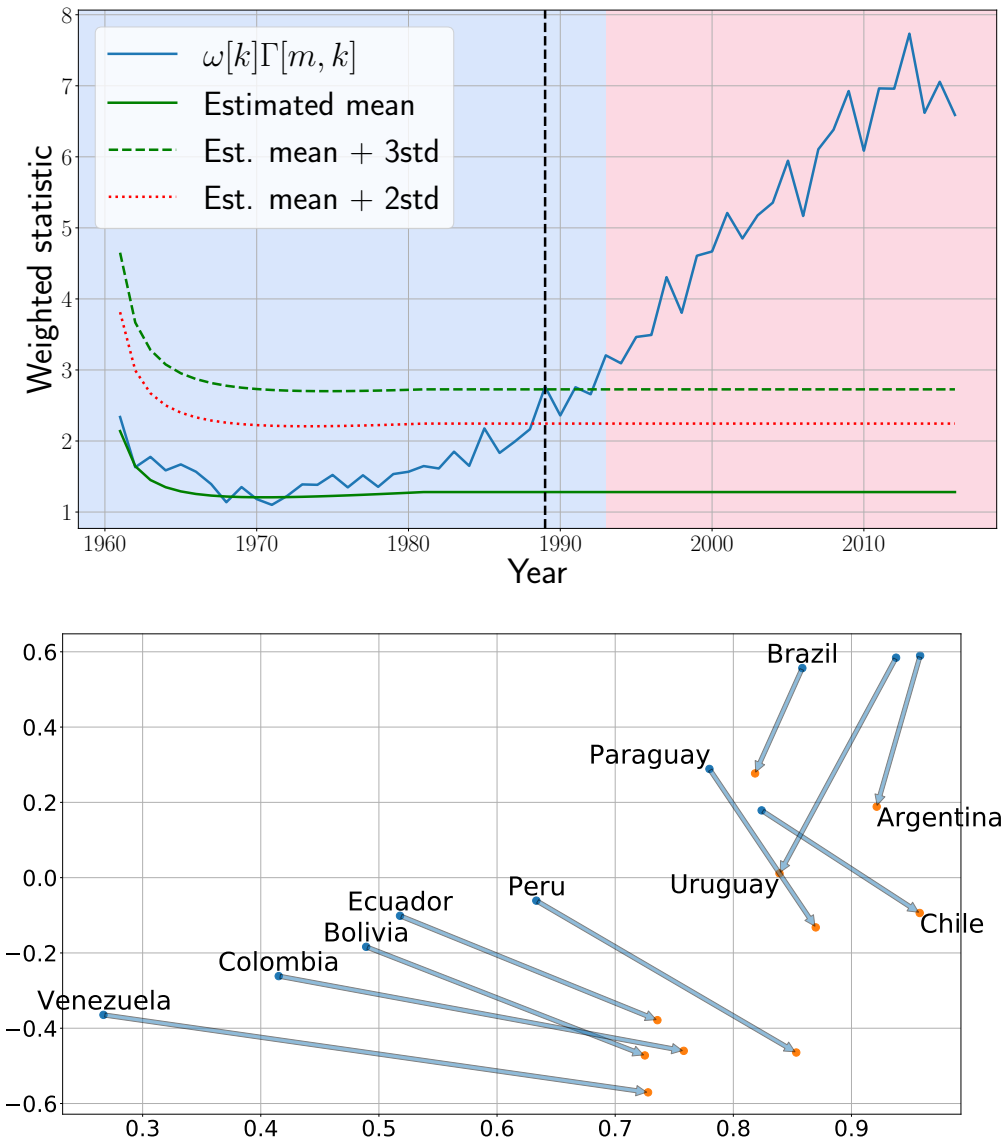

Figure 5.9: Online CPD for the South American football matches. Top: evolution of MOSUM statistic. The dashed vertical line shows the detected change-point, that can be traced to a change in the *Copa América* organization format. A change in background color indicates a change-point detected by the offline algorithm (Madrid Padilla et al., 2022). Bottom: embeddings corresponding to the averaged historic set (blue) and the last 10 graphs of the observation period (orange). There are two distinct communities (northern and southern countries), and an increase of the number of matches played by the northern countries (with relatively less football tradition at the time) is clear by the changes in its embeddings.

noise settings as in this test case.

5.4.2.3 MIT proximity network

Lastly, let us consider the stream of social graphs introduced in Eagle and Pentland (2006). The dataset includes the cell tower to which the mobile phone of a group of MIT faculty and graduate students connected between July 2004 and June 2005.

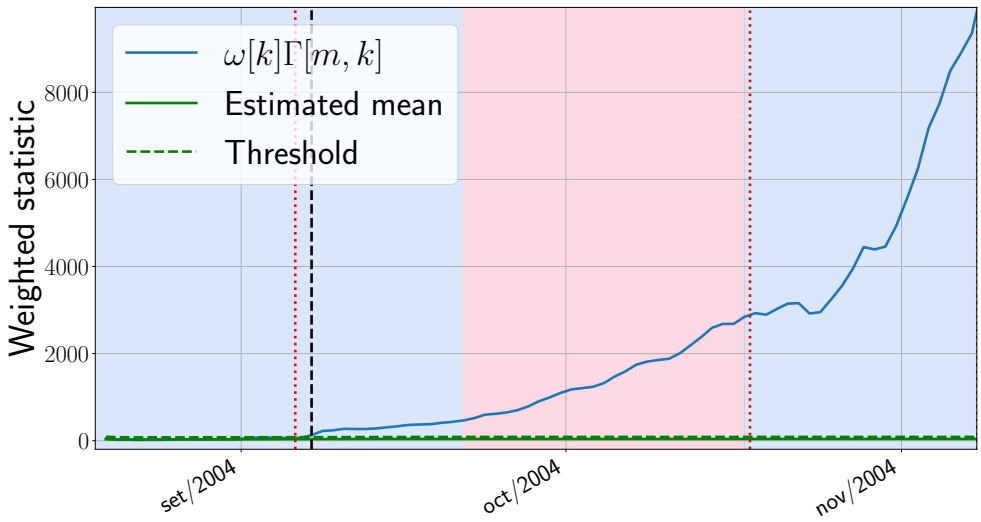

Figure 5.10: Online CPD for the MIT proximity dataset (using the MOSUM window). A change in background color indicates a change-point detected by the offline algorithm of Madrid Padilla et al. (2022). The dashed vertical line shows the detected change-point for the online algorithm. Dotted vertical lines indicate the beginning of the semester and the "sponsor week". The offline algorithm misses the first change-point.

We have processed the dataset and constructed a daily graph where nodes are people and the weight of each edge is how many minutes two people share the same tower on that given day[1]. A collection of labeled events are described in Peel and Clauset (2014, Fig. 8), such as the beginning of the semester in early September and the "sponsor week" during mid-October.

We have considered a full month worth of undirected graphs starting on mid-July as training set and all the $N = 84$ people that were registered during the study. The evolution of the MOSUM statistic until early November is shown in Figure 5.10. Dotted vertical red lines indicate the two events we mentioned before, which fall within the observation period. First of all, it is important to note that the online CPD algorithm detects a change during early September, very near to the beginning of the semester. This change-point is missed by the offline algorithm in Madrid Padilla et al. (2022) (see the changes on the background color), which indicates a change-point almost two weeks later. Furthermore, the example illustrates an interesting advantage of a finite-memory statistic such as MOSUM: the second change-point (this time correctly flagged by the offline algorithm) is also clearly discernible. Notice how the statistic is starting to stabilize around mid-October and then presents a large change of slope. Indeed, changes on the statistic after plateauing are indicative of further change-points.

[1]We used Jeremy Kun's scripts in https://github.com/j2kun/reality-mining.

5.5 Concluding remarks

We developed a computationally-efficient online CPD algorithm for monitoring applications involving streaming network data. The goal is to declare in (pseudo) real time when a sequence of observed graphs changes its underlying distribution. Leveraging the RDPG modeling framework and assuming historical "clean" data are available, the novel algorithm computes (offline) the ASE of the historical graphs (i.e., a training set) and then efficiently updates the cumulative sum of a monitoring function as data arrive sequentially-in-time. Statistical analysis of the monitored random sequence facilitates deriving meaningful detection thresholds to control type-I error rates, as well as to study the algorithm's detectability limits and to numerically predict delay behavior. By resorting to generalizations of the RDPG model to directed and weighted graphs we markedly broaden the applicability of the novel online CPD framework, as illustrated through various real-data case studies.

Chapter 6

Conclusions and future work

The study of network data through latent position models, and in particular the RDPG model, has yielded powerful insights into the structure, dynamics, and inference procedures associated with complex graphs. This thesis focused on the RDPG model as a unifying framework to address key challenges in statistical network analysis, from embedding and estimation to model extensions and applications. In this final chapter, we reflect on the core findings and contributions developed throughout the thesis, organizing them by thematic emphasis and technical progression.

First, we revisited spectral embedding methods under the RDPG model and proposed a new formulation that more faithfully represents the underlying model assumptions. This alternative formulation not only better captures the probabilistic structure of the model but also enables a principled analysis of the optimization landscape. Our results demonstrate that the embedding problem exhibits a benign landscape under appropriate regularity conditions, laying a foundation for provably correct non-convex optimization approaches.

Building upon this formulation, we then turned to directed networks, which pose additional challenges due to inherent asymmetries in their structure. We introduced a Riemannian optimization algorithm for embedding directed RDPG networks that directly addresses the non-identifiability induced by the model's invariances. Our approach provides a practical and theoretically grounded solution to a problem that is often overlooked in the literature, especially in the context of real-world applications where directed edges are prevalent.

We next proposed an extension of the RDPG model to the setting of weighted graphs. While the original formulation is limited to binary adjacency matrices, many applications involve edge weights reflecting intensity, frequency, or capacity of interactions. We introduced the Weighted RDPG (WRDPG) model and provided an inference procedure for estimating latent positions from weighted observations. Our results include consistency guarantees and a method for generating graphs under the WRDPG model, enabling synthetic generation of networks that mirror real-world

statistical properties.

Finally, we explored the use of the RDPG model in change-point detection (CPD), illustrating how embeddings can reveal temporal shifts in network structure. By modeling each graph in a sequence as an RDPG, we developed a lightweight framework for detecting statistically significant changes in latent structure in an online fashion. This approach combines theoretical robustness with practical applicability, highlighting the utility of latent position models beyond static settings.

Collectively, these contributions advance the understanding of the RDPG model along several axes: formulation, algorithmic development, model generalization, and practical application. They underscore the versatility of spectral methods for statistical network analysis and open several directions for future work, which we outline next.

6.1 Future work

Building upon the results of this thesis, there are several promising directions for future research, which we organize into four broad categories: theoretical directions, algorithmic and computational enhancements, application-oriented extensions, and modeling directions.

6.1.1 Theoretical directions

A key open theoretical problem pertains to the online CPD algorithm presented in Chapter 5. While this method is shown to be effective in practice, providing rigorous statistical guarantees for its detection delay remains an important challenge. Establishing such results would help formalize its performance in both finite-sample and asymptotic regimes. For the WRDPG model introduced in Chapter 4, future work could investigate robustness under model misspecification, as well as extend the current consistency guarantees to more general settings involving dependent edge weights or partially observed graphs. Additionally, the reformulated embedding problem introduced in Chapter 3 raises interesting questions regarding global optimality guarantees. Since, when accounting for unobserved or missing data, the optimization landscape remains non-convex, understanding when and how local minima can be avoided would offer valuable insights into the algorithm's theoretical properties.

6.1.2 Algorithmic and computational enhancements

From a computational standpoint, improving the scalability and efficiency of our methods is a natural next step. One avenue is to port the current Python imple-

mentations to lower-level languages such as C or Rust to gain significant performance improvements, particularly in large-scale or real-time applications. Another promising direction is to explore the parallelization of the block coordinate descent algorithm developed in Chapter 3, which could make it feasible for even larger graphs. In streaming scenarios, developing online rules for adaptively selecting the embedding dimension remains a practical challenge, and could help improve performance in dynamic environments. Furthermore, while the online gradient descent techniques proposed in Chapter 3 have shown good empirical results, a dynamic regret analysis in non-convex settings would represent a valuable contribution to the theoretical literature on streaming optimization.

6.1.3 Application-oriented extensions

Chapter 5 already presents an efficient and lightweight online CPD pipeline, which accounts for partially observed networks and dynamic node sets. Nonetheless, several additional application-focused extensions are possible. For instance, it would be interesting to develop CPD techniques that rely solely on graph signal observations, rather than explicit adjacency matrices. This would be particularly valuable in settings where only indirect measurements of network activity are available, such as through diffusion or communication processes. Moreover, one could extend the CPD framework to detect more complex structural changes, such as shifts in latent dimension or edge variability, particularly under the WRDPG model where edge weights carry additional information. Such extensions would further broaden the practical utility of our methodology in real-world systems.

6.1.4 Modeling directions

Finally, several avenues remain open for enriching the modeling framework introduced in this thesis. While Chapter 3 already provides a method for embedding partially observed networks, and Chapter 5 leverages this to handle streaming graphs with varying node sets, further exploration is warranted. In particular, it would be useful to formalize inference procedures for networks with temporally varying availability or sampling schemes. The WRDPG model proposed in Chapter 4 also opens the door to more general formulations of weighted latent position models, especially those capable of incorporating temporal dynamics, edge attributes, or more complex dependence structures. Another promising direction involves adapting the WRDPG methodology to heterophilous graphs, as motivated by the generalized RDPG model of Rubin-Delanchy et al. (2022). Our framework, being based on spectral techniques such as those in Gallagher et al. (2023), is well-suited for this extension.

Bibliography

Absil, P.-A., Mahony, R., & Sepulchre, R. (2009). *Optimization algorithms on matrix manifolds*. Princeton University Press.

Athreya, A., Fishkind, D. E., Tang, M., Priebe, C. E., Park, Y., Vogelstein, J. T., Levin, K., Lyzinski, V., & Qin, Y. (2017). Statistical inference on random dot product graphs: A survey. *J. Mach. Learn. Res.*, *18*(1), 8393–8484.

Athreya, A., Priebe, C. E., Tang, M., Lyzinski, V., Marchette, D. J., & Sussman, D. L. (2016). A limit theorem for scaled eigenvectors of random dot product graphs. *Sankhya A*, *78*(1), 1–18.

Belkin, M., & Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for mmbedding and clustering. *Proc. Adv. Neural. Inf. Process. Syst.*, *14*, 1–7.

Bentkus, V. (2003). On the dependence of the Berry–Esseen bound on dimension. *Journal of Statistical Planning and Inference*, *113*(2), 385–402.

Bertsekas, D. (1999). *Nonlinear programming*. Athena Scientific.

Bhojanapalli, S., Kyrillidis, A., & Sanghavi, S. (2016). Dropping convexity for faster semi-definite optimization. *Proc. Conf. Learn. Theory*, 530–582.

Bickel, P. J., & Doksum, K. A. (2015). *Mathematical statistics: Basic ideas and selected topics*. Chapman; Hall/CRC.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiments*, *2008*(10), P10008.

Borgs, C., Chayes, J. T., Lovász, L., Sós, V. T., & Vesztergombi, K. (2008). Convergent sequences of dense graphs I: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, *219*(6), 1801–1851.

Boumal, N. (2023). *An introduction to optimization on smooth manifolds*. Cambridge University Press.

Boumal, N., Absil, P.-A., & Cartis, C. (2018). Global rates of convergence for nonconvex optimization on manifolds. *IMA Journal of Numerical Analysis*, *39*(1), 1–33.

Bourgade, P., Huang, J., & Yau, H.-T. (2017). Eigenvector statistics of sparse random matrices. *Electronic Journal of Probability*, *22*, 1–38.

Bourin, C., & Bondon, P. (1998). Efficiency of high-order moment estimates. *IEEE Transactions on Signal Processing*, *46*(1), 255–258.

Boyd, J. P., & Gally, D. H. (2007). Numerical experiments on the accuracy of the Chebyshev–Frobenius companion matrix method for finding the zeros of a truncated series of Chebyshev polynomials. *Journal of Computational and Applied Mathematics*, *205*(1), 281–295.

Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition [Special Issue on Large Scale Linear and Nonlinear Eigenvalue Problems]. *Linear Algebra Appl.*, *415*(1), 20–30.

Burer, S., & Monteiro, R. D. (2005). Local minima and convergence in low-rank semidefinite programming. *Mathematical programming*, *103*(3), 427–444.

Campos, P., Díez, F., & Cantador, I. (2014). Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols. *User Model. User-adapt. Interact.*, *24*, 67–119.

Capdehourat, G., Larroca, F., & Morales, G. (2020). A nation-wide Wi-Fi RSSI dataset: Statistical analysis and resulting insights. *Proc. IFIP Networking Conf.*, 370–378.

Cape, J., Tang, M., & Priebe, C. E. (2019). The two-to-infinity norm and singular subspace geometry with applications to high-dimensional statistics. *The Annals of Statistics*, *47*(5), pp. 2405–2439.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., & Murphy, K. (2022). Machine learning on graphs: A model and comprehensive taxonomy. *J. Mach. Learn. Res.*, *23*(89), 1–64.

Chen, H. (2019). Sequential change-point detection based on nearest neighbors. *Ann. Stat*, *47*(3), 1381–1407.

Chi, Y., Lu, Y., & Chen, Y. (2019). Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Trans. Signal Process.*, *67*(20), 5239–5269.

Chung, J., Pedigo, B. D., Bridgeford, E. W., Varjavand, B. K., Helm, H. S., & Vogelstein, J. T. (2019). GraSPy: Graph statistics in Python. *J. Mach. Learn. Res.*, *20*(158), 1–7.

Davenport, M. A., & Romberg, J. (2016). An overview of low-rank matrix recovery from incomplete observations. *IEEE J. Sel. Topics Signal Process.*, *10*(4), 608–622.

De Bruijn, N. G. (2014). *Asymptotic methods in analysis*. Courier Corporation.

DeFord, D. R., & Rockmore, D. N. (2016). A random dot product model for weighted networks. *arXiv:1611.02530 [stat.AP]*.

Dokmanic, I., Parhizkar, R., Ranieri, J., & Vetterli, M. (2015). Euclidean distance matrices: Essential theory, algorithms, and applications. *IEEE Signal Process. Mag.*, *32*(6), 12–30.

Eagle, N., & Pentland, A. (2006). Reality mining: Sensing complex social systems. *Personal Ubiquitous Comput.*, *10*(4), 255–268.

Eckart, C., & Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, *1*(3), 211–218.

Ferrari, A., Richard, C., & Verduci, L. (2019). Distributed change detection in streaming graph signals. *Proc. IEEE Workshop on Computational Advances in Multi-Sensor Adaptive Process.*, 166–170.

Fiori, M., Marenco, B., Larroca, F., Bermolen, P., & Mateos, G. (2023). Gradient-Based Spectral Embeddings of Random Dot Product Graphs. *IEEE Transactions on Signal and Information Processing over Networks*, *10*, 1–16.

Gallagher, I., Jones, A., & Rubin-Delanchy, P. (2021). Spectral embedding for dynamic networks with stability guarantees. *Proc. Adv. Neural. Inf. Process. Syst.*, 1–13.

Gallagher, I., Jones, A., Bertiger, A., Priebe, C. E., & Rubin-Delanchy, P. (2023). Spectral embedding of weighted graphs. *J. Am. Stat. Assoc.*, *0*(0), 1–10.

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 855–864.

Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught, & J. Millman (Eds.), *Proceedings of the 7th python in science conference* (pp. 11–15).

Halko, N., Martinsson, P. G., & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, *53*(2), 217–288.

Hamilton, W. L. (2020). Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *14*, 1–159.

Harris, C. R., Millman, K. J., van der Walt, S. J., & et al. (2020). Array programming with NumPy. *Nature*, *585*, 357–362.

He, X., Xie, Y., Wu, S.-M., & Lin, F.-C. (2018). Sequential graph scanning statistic for change-point detection. *52nd Asilomar Conference on Signals, Systems, and Computers*, 1317–1321.

Hoff, P. D., Raftery, A. E., & Handcock, M. S. (2002). Latent space approaches to social network analysis. *J. Am. Stat. Assoc.*, *97*(460), 1090–1098.

Holland, P., Laskey, K. B., & Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social Networks*, *5*(2), 109–137.

Horn, R. A., & Johnson, C. R. (2012). *Matrix Analysis*. Cambridge University Press.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*, 193–218.

Imhof, J. P. (1961). Computing the distribution of quadratic forms in normal variables. *Biometrika*, *48*(3-4), 419–426.

Izenman, A. J. (2023). *Network models for data science: Theory, algorithms, and applications*. Cambridge University Press.

Jones, A., & Rubin-Delanchy, P. (2020). The multilayer random dot product graph. *arXiv:2007.10455 [stat.ML]*.

Kalantzis, V., & Traganitis, P. (2023). Matrix resolvent eigenembeddings for dynamic graphs. *Proc. Int. Conf. Acoustics, Speech, Signal Process.*

Kapur, J., & Kesavan, H. (1992). *Entropy optimization principles with applications*. Academic Press.

Kaushik, C., Roddenberry, T., & Segarra, S. (2021). Network topology change-point detection from graph signals with prior spectral signatures. *Proc. Int. Conf. Acoustics, Speech, Signal Process.*, 5395–5399.

Keshavarz, H., Michaildiis, G., & Atchade, Y. (2020). Sequential change-point detection in high-dimensional Gaussian graphical models. *J. Mach. Learn. Res.*, *21*(82), 1–57.

Kirch, C., & Tadjuidje Kamgaing, J. (2015). On the use of estimating functions in monitoring time series for change points. *J. Stat. Plan. Inference*, *161*, 25–49.

Kirch, C., & Weber, S. (2018). Modified sequential change point procedures based on estimating functions. *Electron. J. Statist.*, *12*(1), 1579–1613.

Kolaczyk, E. D. (2009). *Statistical analysis of network data: Methods and models*. Springer.

Kolaczyk, E. D. (2017). *Topics at the frontier of statistics and network analysis: (re)visiting the foundations*. Cambridge University Press.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37.

Lancichinetti, A., Fortunato, S., & Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Phys. Rev. E*, *78*, 046110.

Levin, K., Athreya, A., Tang, M., Lyzinski, V., & Priebe, C. E. (2017). A central limit theorem for an omnibus embedding of multiple random dot product graphs. *Int. Conf. on Data Mining Workshops*, 964–967.

Levin, K., Roosta, F., Mahoney, M., & Priebe, C. (2018). Out-of-sample extension of graph adjacency spectral embedding. *Proc. Int. Conf. Mach. Learn.*, *80*, 2975–2984.

Li, Y., & Mateos, G. (2022). Networks of international football: Communities, evolution and globalization of the game. *Applied Network Science*, *7*.

Lovász, L., & Szegedy, B. (2006). Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, *96*(6), 933–957.

Lovász, L., & Szegedy, B. (2007). Szemerédi's lemma for the analyst. *GAFA Geometric And Functional Analysis*, *17*(1), 252–270.

Luo, Y., & Garcia Trillos, N. (2022). Nonconvex matrix factorization is geodesically convex: Global landscape analysis for fixed-rank matrix optimization from a Riemannian perspective. *arXiv preprint arXiv:2209.15130*.

Lyzinski, V., Tang, M., Athreya, A., Park, Y., & Priebe, C. E. (2017). Community detection and classification in hierarchical stochastic blockmodels. *IEEE Trans. Netw. Sci. Eng.*, *4*(1), 13–26.

Madrid Padilla, O. H., Yu, Y., & Priebe, C. E. (2022). Change point localization in dependent dynamic nonparametric random dot product graphs. *Journal of Machine Learning Research*, *23*(234), 1–59.

Marchette, D., Priebe, C., & Coppersmith, G. (2011). Vertex nomination via attributed random dot product graphs. *Proceedings of the 57th ISI World Statistics Congress*, *6*, 16.

Marenco, B., Bermolen, P., Fiori, M., Larroca, F., & Mateos, G. (2022). Online Change Point Detection for Weighted and Directed Random Dot Product Graphs. *IEEE Trans. Signal Inf. Process. Netw.*, *8*, 144–159.

Marenco, B., Bermolen, P., Fiori, M., Larroca, F., & Mateos, G. (2025). Weighted random dot product graphs. *arXiv preprint arXiv:2505.03649*.

Mateos, G., & Rajawat, K. (2013). Dynamic network cartography: Advances in network health monitoring. *IEEE Signal Process. Mag.*, *30*(3), 129–143.

McKinney, W. (2010). Data structures for statistical computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61).

Munuswamy, S. (1963). Approximations to the non-central chi-square distribution. *Biometrika*, *50*(1-2), 199–204.

Newman, M. (2018). *Networks*. Oxford University Press.

Nocedal, J., & Wright, S. J. (2006). *Numerical Optimization* (2nd). Springer.

Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, *41*(1-2), 100–115.

Peel, L., & Clauset, A. (2014). Detecting change points in the large-scale structure of evolving networks. *arXiv:1403.0989 [cs.SI]*.

Priebe, C. E., Park, Y., Tang, M., Athreya, A., Lyzinski, V., Vogelstein, J. T., Qin, Y., Cocanougher, B., Eichler, K., Zlatic, M., & Cardona, A. (2017). Semiparametric spectral modeling of the drosophila connectome. *arXiv:1705.03297 [stat.ML]*.

Rencher, A. C., & Schaalje, G. B. (2008). *Linear models in statistics*. John Wiley & Sons.

Rosenberg, A., & Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. *Proc. of the 2007 Joint Conference on*

Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 410–420.

Rubin-Delanchy, P., Cape, J., Tang, M., & Priebe, C. E. (2022). A statistical interpretation of spectral embedding: The generalised random dot product graph. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *84*(4), 1446–1473.

Saad, T., & Ruai, G. (2019). Pymaxent: A python software for maximum entropy moment reconstruction. *SoftwareX*, *10*, 100353.

Scheinerman, E., & Tucker, K. (2010). Modeling graphs using dot product representations. *Comput. Stat*, *25*, 1–16.

Shore, J., & Johnson, R. (1980). Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *IEEE Transactions on Information Theory*, *26*(1), 26–37.

Sussman, D. L., Tang, M., Fishkind, D. E., & Priebe, C. E. (2012). A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, *107*(499), 1119–1128.

Sussman, D. L., Tang, M., & Priebe, C. E. (2014). Consistent latent position estimation and vertex classification for random dot product graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(1), 48–57.

Tang, M., Athreya, A., Sussman, D., Lyzinski, V., Park, Y., & Priebe, C. (2017). A semiparametric two-sample hypothesis testing problem for random graphs. *J. Comput. Graph. Stat.*, *26*(2).

Tang, R., Ketcha, M., Badea, A., Calabrese, E. D., Margulies, D. S., Vogelstein, J. T., Priebe, C. E., & Sussman, D. L. (2018). Connectome smoothing via low-rank approximations. *IEEE Trans. Med. Imaging*, *38*(6), 1446–1456.

Tang, R., Tang, M., Vogelstein, J. T., & Priebe, C. E. (2017). Robust estimation from multiple graphs under gross error contamination. *arXiv:1707.03487 [stat.ME]*.

Townsend, J., Koep, N., & Weichwald, S. (2016). Pymanopt: A Python toolbox for optimization on manifolds using automatic differentiation. *J. Mach. Learn. Res.*, *17*(137), 1–5.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, *12*, 389–434.

Truong, C., Oudre, L., & Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Process.*, *167*, 107299.

Vershynin, R. (2018). *High-dimensional probability: An introduction with applications in data science*. Cambridge University Press.

Vinh, N. X., Epps, J., & Bailey, J. (2009). Information theoretic measures for clusterings comparison: Is a correction for chance necessary? *Proc. International Conference on Machine Learning (ICML)*, 1073–1080.

Vladimirova, M., Girard, S., Nguyen, H., & Arbel, J. (2020). Sub-Weibull distributions: Generalizing sub-Gaussian and sub-Exponential properties to heavier tailed distributions. *Stat*, *9*(1), e318.

Voeten, E., Strezhnev, A., & Bailey, M. (2009). *United Nations General Assembly Voting Data*.

Vu, T., & Raich, R. (2021). Exact linear convergence rate analysis for low-rank symmetric matrix completion via gradient descent. *Proc. Int. Conf. Acoustics, Speech, Signal Process.*, 3240–3244.

Wang, H., Tang, M., Park, Y., & Priebe, C. E. (2014). Locality statistics for anomaly detection in time series of graphs. *IEEE Trans. Signal Process.*, *62*(3), 703–717.

Wang, L., Zhang, X., & Gu, Q. (2017). A unified computational and statistical framework for nonconvex low-rank matrix estimation. *Artificial intelligence and statistics*, 981–990.

Xie, F., & Xu, Y. (2023). Efficient estimation for random dot product graphs via a one-step procedure. *J. Am. Stat. Assoc.*, *118*(541), 651–664.

Yu, Y., Padilla, O. H. M., Wang, D., & Rinaldo, A. (2021). Optimal network online change point localisation. *arXiv:2101.05477 [math.ST]*.

Yu, Y., Wang, T., & Samworth, R. J. (2015). A useful variant of the Davis–Kahan theorem for statisticians. *Biometrika*, *102*(2), 315–323.

Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of anthropological research*, *33*(4), 452–473.

Zhang, M., Xie, L., & Xie, Y. (2020). Online community detection by spectral CUSUM. *Proc. Int. Conf. Acoustics, Speech, Signal Process.*

Zhang, Z., Cui, P., Pei, J., Wang, X., & Zhu, W. (2018). TIMERS: Error-bounded SVD restart on dynamic networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1).

Zhou, D., Cao, Y., & Gu, Q. (2020). Accelerated factored gradient descent for low-rank matrix factorization. *Proc. Int. Conf. Artif. Intell. Statist.*, 4430–4440.

Zhu, M., & Ghodsi, A. (2006). Automatic dimensionality selection from the scree plot via the use of profile likelihood. *Comput Stat Data Anal*, *51*(2), 918–930.

Zhu, Z., Li, Q., Tang, G., & Wakin, M. B. (2021). The global optimization geometry of low-rank matrix optimization. *IEEE Transactions on Information Theory*, *67*(2), 1308–1331.