Online Change Point Detection for Weighted and Directed Random Dot Product Graphs

Bernardo Marenco, Paola Bermolen, Marcelo Fiori, Federico Larroca, and Gonzalo Mateos, *Senior Member, IEEE*

***Abstract*—Given a sequence of random (directed and weighted) graphs, we address the problem of online monitoring and detection of changes in the underlying data distribution. Our idea is to endow sequential change-point detection (CPD) techniques with a graph representation learning substrate based on the versatile Random Dot Product Graph (RDPG) model. We consider efficient, online updates of a judicious monitoring function, which quantifies the discrepancy between the streaming graph observations and the nominal RDPG. This reference distribution is inferred via spectral embeddings of the first few graphs in the sequence. We characterize the distribution of this running statistic to select thresholds that guarantee error-rate control, and under simplifying approximations we offer insights on the algorithm's detection resolution and delay. The end result is a lightweight online CPD algorithm, that is also explainable by virtue of the well-appreciated interpretability of RDPG embeddings. This is in stark contrast with most existing graph CPD approaches, which either rely on extensive computation, or they store and process the entire observed time series. An apparent limitation of the RDPG model is its suitability for undirected and unweighted graphs only, a gap we aim to close here to broaden the scope of the CPD framework. Unlike previous proposals, our non-parametric RDPG model for weighted graphs does not require a priori specification of the weights' distribution to perform inference and estimation. This network modeling contribution is of independent interest beyond CPD. We offer an open-source implementation of the novel online CPD algorithm for weighted and direct graphs, whose effectiveness and efficiency are demonstrated via (reproducible) synthetic and real network data experiments.**

***Index Terms*—Online change-point detection, graph representation learning, node embeddings, random dot product graphs.**

I. Introduction

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., [3] 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., [4]), 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 [5], financial markets [6], or, social networks [7], [8]. 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.

A. 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 [5]–[7], their topology [8] and community structure [9], or else the distribution of signals supported on the nodes [10]. Some of these [6]–[8] are only applicable to undirected graphs. A sequential non-parametric, k-nearest neighbors-based approach was developed in [11], 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 V-A. A computationally-intensive model-based CPD effort advocates the Generalized Hierarchical Random Graph (GHRG) model in [7], which monitors posterior Bayes factors for all partitions of the data over a sliding window. The approach in [12] 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 [13] 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

Manuscript received September 3, 2021; revised December 27, 2021 and January 24, 2022; accepted January 25, 2022. Date of publication February 7, 2022; date of current version February 28, 2022. This work was supported in part by ANII under Grant FMV_3_2018_1_148149 and in part by NSF under Awards CCF-1750428, CCF-1934962, and ECCS-1809356. Part of the results in this paper were submitted to the 2021 EUSIPCO and Asilomar Conferences [1], [2]. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Paolo Di Lorenzo. *(Corresponding author: Gonzalo Mateos.)*

Bernardo Marenco, Paola Bermolen, Marcelo Fiori, and Federico Larroca are with the Facultad de Ingeniería, Universidad de la República, Montevideo 11000, Uruguay (e-mail: bmarenco@fing.edu.uy; paola@fing.edu.uy; mfiori@fing.edu.uy; flarroca@fing.edu.uy).

Gonzalo Mateos is with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627 USA (e-mail: gmateosb@ece.rochester.edu).

Digital Object Identifier 10.1109/TSIPN.2022.3149098

threshold. Evaluating these statistics necessitates computing the eigendecomposition of an $N \times N$ matrix (N is the number of graph nodes). In addition to being computationally intensive, the algorithm in [13] 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 Random Dot Product Graph (RDPG) model, a particular but very versatile case of the inhomogeneous Bernoulli graph [14], [15]. In RDPGs each node has an associated latent position in $\mathbb{R}^d$ with $d \leq N$, and P_{ij} is given by the inner product between the corresponding vectors. As we discuss in Section II, RDPGs capture phenomena commonly encountered with real-world graphs (e.g., statistical dependencies among edges) and subsume the SBM as a special case, while still being amenable to analysis [15]. Moreover, RDPGs offer interpretability, an attractive feature that simplifies the explanation of the detected change-points.

B. Paper Outline and Contributions

Building on [16], we assume a clean historical dataset with no change-points is available, from which we estimate the latent nodal vectors via the adjacency spectral embedding (ASE) in an offline training phase. As new data arrive in a streaming fashion during the operational phase, the novel online CPD algorithm (Section III) 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.

An additional contribution is to extend the vanilla RDPG model [14], [17] to accommodate weighted and directed graphs (digraphs), which we seamlessly adopt to perform online CPD for these general network models (Section IV). Extensions to digraphs are straightforward [18], but we carefully study those ambiguities inherent to the model (not discussed in previous work) which may challenge downstream CPD methods. Unlike previous RDPG proposals for weigthed graphs [19], [20], our new non-parametric model in Section IV-B does not require *a priori* specification of the weights' distribution to perform provably consistent inference and estimation. We believe this contribution is significant in its own right, and beyond CPD it can e.g., impact node classification and visualization of network data. Numerical tests in Section V 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 and future directions are outlined in Section VI.

Relative to its conference precursors [1], [2], here we consider online CPD for weighted and directed graphs through a unified presentation along with full-blown technical details (including extended discussions, examples and unpublished proofs for all the theoretical results). Noteworthy novel pieces include: (i) examination of delay and change-detectability conditions; (ii) adoption of finite-memory (windowed) statistics; (iii) integrating the directed and weighted RDPG models for *online* CPD; (iv) a consistency result for the weighted RDPG embeddings; and (v) a comprehensive and reproducible performance evaluation protocol. The latter offers comparisons with batch and online CPD baselines; an study of detection delay; the choice of monitoring function and thresholds; as well as applications to wireless and social networks.

II. Preliminaries and Problem Statement

Here we introduce the necessary background on RDPG modeling and inference. The interested reader is referred to the comprehensive survey [15] for additional details about batch statistical network analysis. We then state the online CPD problem where the streaming graphs are modeled as RDPGs.

A. Random Dot Product Graphs

Consider an unweighted and undirected graph $G = (\mathcal{V}, \mathcal{E})$, with nodes $\mathcal{V} = \{1, \ldots, N\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. If nodes i and j are connected in G, then the unordered pair $(i,j) \in \mathcal{E}$. More general models involving directed and weighted graphs will be dealt with in Section IV. To start, we restrict ourselves to the simplest possible case for ease of exposition.

In the RDPG model of G each node $i \in \mathcal{V}$ has an associated latent position vector $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$, and edge (i,j) exists with probability $P_{ij} = \mathbf{x}_i^\top \mathbf{x}_j$, independent of all other edges. We do not allow for self loops, hence $P_{ii} = 0$ for all $i \in \mathcal{V}$. The geometric interpretation is that nodes with large $\|\mathbf{x}_i\|_2$ tend to exhibit higher connectivity, whereas a small angle between $\mathbf{x}_i$ and $\mathbf{x}_j$ indicates higher "affinity" among i and j. Note that the set $\mathcal{X}$ of possible $\mathbf{x}_i$ is such that $\mathbf{x}^\top \mathbf{y} \in [0,1]$, for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. Just like with blockmodels and SBMs [21], in general vectors $\mathbf{x}_i$ may be random, drawn from a (so-termed inner product) distribution in $\mathcal{X}$. The dimensionality d of the latent space is a model parameter, often much smaller than N.

Thus, letting $\mathbf{A} \in \{0,1\}^{N \times N}$ be the random symmetric adjacency matrix of G and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times d}$ the matrix of latent vertex positions, the RDPG model specifies

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

That is, *given* $\mathbf{X}$, edges are conditionally independent with $A_{ij} \sim \text{Bernoulli}(\mathbf{x}_i^\top \mathbf{x}_j)$.

Example 1: The RDPG model is a tractable yet expressive family of random graphs that subsume Erds-Rényi (ER) and SBM ensembles as particular cases. Indeed, if $\mathbf{x}_i = \sqrt{p}$ for all i, we obtain an ER graph with edge probability p. An SBM with M communities may be generated by restricting $\mathbf{X}$ to having only (at most) M different columns (i.e. $|\mathcal{X}| = M$); see also [15] for additional examples. On the other hand, the RDPG is a particular case of the latent space model [22], in which edge probabilities $P_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$ are specified by means of a symmetric link function κ.

B. Inference on RDPG via the Adjacency Spectral Embedding

Given the matrix $\mathbf{X}$ of latent vertex positions, the joint distribution in (1) specifies the generative process to sample graphs from the RDPG model. We now discuss the associated inference (a.k.a. node embedding) problem. That is, how to estimate $\mathbf{X}$ having observed a graph stemming from an RDPG with adjacency matrix $\mathbf{A}$.

In lieu of a maximum-likelihood estimator that is intractable beyond toy graphs [23], the key intuition is that $\mathbf{A}$ is a noisy observation of

$$\mathbf{P} = \mathbf{X}\mathbf{X}^\top, \tag{2}$$

the rank-d matrix of edge probabilities P_{ij}, since $\mathbb{E}[\mathbf{A} \mid \mathbf{X}] = \mathbf{P}$. It is thus natural to adopt the estimator

$$\hat{\mathbf{X}} = \underset{\mathbf{X}}{\text{argmin}} \, \|\mathbf{A} - \mathbf{X}\mathbf{X}^\top\|_F^2, \text{ s. to rank}(\mathbf{X}) = d. \tag{3}$$

The solution to (3) is readily given by

$$\hat{\mathbf{X}} = \hat{\mathbf{V}}\hat{\mathbf{\Lambda}}^{1/2}, \tag{4}$$

where $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$ is the eigendecomposition of $\mathbf{A}$, $\hat{\mathbf{\Lambda}} \in \mathbb{R}^{d\times d}$ is a diagonal matrix with the d largest eigenvalues of $\mathbf{A}$, and $\hat{\mathbf{V}} \in \mathbb{R}^{N\times d}$ are the corresponding d dominant eigenvectors. We are assuming that $\hat{\mathbf{\Lambda}}$ has only non-negative values, an apparent limitation that may be easily circumvented [24]. The bias introduced by the implicit constraint diag$(\mathbf{X}\mathbf{X}^\top) \approx \mathbf{0}$ can be alleviated as well [23]. In practice, d is likely unknown but can be estimated by looking for "elbows" on the so-termed eigenvalue scree plot [25]. We find it is safer to overestimate d (which will add some noise) than underestimate it, that will oversimplify the model and may e.g., hide change-points [1]. Estimator (4) is known as the Adjacency Spectral Embedding (ASE), which is asymptotically normal and approaches $\mathbf{X}$ as $N \to \infty$ provided the true d is chosen [15]. It is also possible to define an analogous normalized Laplacian spectral embedding for undirected G, which can be shown to enjoy similar desirable asymptotic properties to those of the ASE [15].

Before moving on and stating the formal online CPD problem addressed in this paper, a couple of remarks on model identifiability and ASE variance reduction are in order.

Remark 1 (Identifiability of Latent Positions): The RDPG model is identifable up to rotations of $\mathbf{X}$. To see this, consider an orthogonal matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$, and note that the rotated vectors $\mathbf{X}\mathbf{W}$ will produce the same probability matrix as in (2). Hence, the estimator (4) is unbiased up to an unknown rotation matrix $\mathbf{W}$, and the ambiguity should be accounted for when detecting changes on G's distribution.

Remark 2 (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

$$\bar{\mathbf{A}} = \frac{1}{m}\sum_{t=1}^{m} \mathbf{A}[t], \tag{5}$$

and henceforth let $\hat{\mathbf{X}}$ be the ASE decomposition of $\bar{\mathbf{A}}$; i.e., the solution of (3) using $\bar{\mathbf{A}}$ instead of $\mathbf{A}$. Since $\bar{\mathbf{A}}$ is also an unbiased estimator of $\mathbf{P}$ and var$[\bar{A}_{ij}] = \frac{1}{m}P_{ij}(1-P_{ij})$, 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 (3) [26]. The alternative of averaging individual ASEs is problematic due to the rotational ambiguity discussed in Remark 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 [27].

C. Problem Statement

Suppose we acquire a batch of m graphs as in Remark 2, 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 (1) 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.

III. Online Change-Point Detection

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.

A. General Algorithmic Framework

We build on the so-called estimating function approach for sequential CPD [16], [28], 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 [16] 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

$$\sum_{t=1}^{m} \mathbf{G}(\mathbf{A}[t], \hat{\mathbf{X}}) = \mathbf{0}. \tag{6}$$

To define such a function for our problem, given the training data set we estimate $\mathbf{X}$ as the ASE corresponding to $\bar{\mathbf{A}}$ [cf. (5) and the discussion in Remark 2]. Taking the derivative w.r.t. $\mathbf{X}$ of the objective function in (3) (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 $\mathbf{G}(\mathbf{A}[t], \hat{\mathbf{X}}) = (\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}[t])\hat{\mathbf{X}}$ 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 V-A 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

$$\mathbf{h}(\mathbf{A}[t], \hat{\mathbf{X}}) = \text{vec}\left[\text{triu}\left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}[t]\right)\right], \tag{7}$$

where vec(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 vec(triu($\mathbf{B}$)) $\in \mathbb{R}^r$, with $r := \frac{N(N-1)}{2}$.

If the norm of the partial sum

$$\mathbf{s}[m,k] = \sum_{t=m+1}^{m+k} \mathbf{h}(\mathbf{A}[t], \hat{\mathbf{X}}) \tag{8}$$

exceeds a certain threshold, we will conclude that the model is no longer valid. Let us then denote our CUSUM statistic as

$$\Gamma[m,k] = \|\mathbf{s}[m,k]\|_2^2.$$

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.

Algorithm 1: Online change-point detection for RDPGs.

```
Require: Training graphs $\mathbf{A}[t], t = 1 \ldots m$.
 1: Compute the ASE $\hat{\mathbf{X}}$ of $\bar{\mathbf{A}}$ in (5) (see Remark 2)
 2: Compute threshold function $c_\alpha$ (see Section III-D)
 3: Initialize partial sum $\mathbf{s}[m,0] = \mathbf{0}$
 4: for $k = 1, 2, \ldots$ do
 5:   Acquire graph $\mathbf{A}[m+k]$
 6:   Compute monitoring function $\mathbf{h}(\mathbf{A}[m+k], \hat{\mathbf{X}})$
 7:   Update CUSUM statistic $\Gamma[m,k]$ (see Remark 3)
 8:   if $w[k]\Gamma[m,k] > c_\alpha[k]$ then
 9:     Change point detected at time $k^* = k$
10:     break
11:   end if
12: end for
13: return $k^*$.
```

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 1.

Remark 3 (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 $t \in [m+1, \ldots, m+k]$. 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.

B. 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.

1) Perfect ASE Estimation: In this favorable case one has[1] $\mathbf{h} = \text{vec}[\text{triu}(\mathbf{P} - \mathbf{A}[t])]$, with $\mathbb{E}[\mathbf{h}] = \mathbf{0}$. 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 $\text{var}[A_{ij}] = P_{ij}(1 - P_{ij})$. 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 $\text{vec}[\text{triu}(\mathbf{P})]$ (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 treshold $c_\alpha[k]$.

Proposition 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

$$k^{-1}\Gamma[m, k] \xrightarrow{D} \sum_{l=1}^{r} p_l(1 - p_l)y_l^2, \tag{9}$$

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 (8) converges to a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_H)$, i.e., $k^{-1/2}\mathbf{s}[m, k] \xrightarrow{D} (\mathbf{\Sigma}_H)^{1/2}\mathbf{y}$, 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

$$\begin{aligned} k^{-1}\Gamma[m, k] &= (k^{-1/2}\mathbf{s}[m, k])^\top k^{-1/2}\mathbf{s}[m, k] \\ &\xrightarrow{D} (\mathbf{\Sigma}_H^{1/2}\mathbf{y})^\top \mathbf{\Sigma}_H^{1/2}\mathbf{y} \\ &= \mathbf{y}^\top \mathbf{\Sigma}_H \mathbf{y} \\ &= \sum_{l=1}^{r} p_l(1 - p_l)y_l^2, \end{aligned}$$

which is the desired result in (9).

Remark 4 (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 1 at a rate $O(k^{-1/2})$, independent of r and hence the graph size N; see e.g., [29, 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 1 and for sufficiently large k, we can approximate the mean and variance of $\Gamma[m, k]$ as

$$\begin{aligned} \mathbb{E}\left[\Gamma[m, k]\right] &\approx k \sum_{l=1}^{r} p_l(1 - p_l), \\ \text{var}\left[\Gamma[m, k]\right] &\approx 2\,k^2 \sum_{l=1}^{r} p_l^2(1 - p_l)^2, \end{aligned} \tag{10}$$

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 (9), 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.

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

$$\mathbf{h}(\mathbf{A}[t], \hat{\mathbf{X}}) = \text{vec}\left[\text{triu}\left(\mathbf{X}\mathbf{X}^\top - \mathbf{A}[t]\right)\right] + \mathbf{e}, \tag{11}$$

where $\mathbf{e} = \text{vec}[\text{triu}(\mathbf{E})] = [e_1, \ldots, e_r]^\top$. So the first term in (11) 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 (11) 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 $\mathcal{N}(k\mathbf{e}, k\mathbf{\Sigma}_H)$. 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

$$\bar{\Gamma} = k \sum_{l=1}^{r} p_l(1 - p_l)\left(y_l + b_l\right)^2, \tag{12}$$

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 $\mathbf{b} = \sqrt{k}\mathbf{\Sigma}_H^{-1/2}\mathbf{e}$. Note that if the ASE estimation is perfect, then $\mathbf{e} = \mathbf{0}$ and we recover the distribution in Proposition 1.

For large k, using (12) 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 $(y_l + b_l)^2 \sim \chi^2(1, b_l^2)$, 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

$$\begin{aligned} \mathbb{E}\left[\Gamma[m, k]\right] &\approx k^2\|\mathbf{e}\|_2^2 + k \sum_{l=1}^{r} p_l(1 - p_l) \\ &= k^2\|\mathbf{e}\|_2^2 + k\|\boldsymbol{\sigma}\|_1, \end{aligned} \tag{13}$$

$$\begin{aligned} \text{var}\left[\Gamma[m, k]\right] &\approx 4\,k^3 \sum_{l=1}^{r} p_l(1 - p_l)e_l^2 + 2\,k^2 \sum_{l=1}^{r} p_l^2(1 - p_l)^2 \\ &= 4\,k^3\boldsymbol{\sigma}^\top\mathbf{e}^2 + 2\,k^2\|\boldsymbol{\sigma}\|_2^2, \end{aligned} \tag{14}$$

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 2: In the general case, as $k \to \infty$ the test statistic sequence converges in distribution, namely

$$\frac{\Gamma[m, k] - k^2\|\mathbf{e}\|_2^2 - k\|\boldsymbol{\sigma}\|_1}{\sqrt{4\,k^3\boldsymbol{\sigma}^\top\mathbf{e}^2 + 2\,k^2\|\boldsymbol{\sigma}\|_2^2}} \xrightarrow{D} y,$$

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 (14) grows like k^3 [cf. k^2 in (10)]. The detection threshold $c_\alpha[k]$ is thus set as the $(1 - \alpha)$-quantile of the generalized chi-squared distribution defined in (12), after weighting. We note that the resulting

[1]We have omitted the dependence of $\mathbf{h}$ on t and $\hat{\mathbf{X}}$ for clarity.

cumulative distribution function has a complex form which requires numerical integration to compute the desired quantiles; see also [30], [31] for classic formulae to approximate said distribution function. As the next example shows, for particular cases the resulting distribution simplifies.

Example 2: For an ER model with connection probability p we have $p_l = p$ for all $l = 1, \dots, r$ and (12) simplifies to

$$\bar{\Gamma}_{\text{ER}} = kp(1-p)u, \text{ with } u \sim \chi^2\left(r, \frac{k}{p(1-p)}\|\mathbf{e}\|_2^2\right). \quad (15)$$

Alternatively, for threshold selection we will often resort to the mean plus three standard deviations

$$\text{th}[k] := \omega[k]\mathbb{E}_{bc}[k] + 3\sqrt{\omega^2[k]\text{var}_{bc}[k]}, \quad (16)$$

where $\mathbb{E}_{bc}$ is the expectation of the statistic before the change and var_{bc} is its variance; given by (13) and (14), respectively, using a suitable estimate of $\mathbf{e}$ described in Section III-D. Numerical tests in Section V-A 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.

C. 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 $\boldsymbol{\Delta} := \mathbf{X}\mathbf{X}^\top - \mathbf{Y}\mathbf{Y}^\top$ (i.e., the change is manifested through a perturbation on the resulting probability matrix). Given the expressive power of RDPGs [15], the modeling assumption for $k \geq k_c$ comes with limited loss of generality. Henceforth, let $\boldsymbol{\delta} := \text{vec}[\text{triu}(\boldsymbol{\Delta})]$.

If we are at a certain time $k > k_c$, the partial sum of the monitoring function is then (recall $\mathbf{E} = \hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{X}\mathbf{X}^\top$)

$$\begin{aligned}\mathbf{s}[m,k] &= \sum_{t=m+1}^{m+k} \mathbf{h}\left(\mathbf{A}[t], \hat{\mathbf{X}}\right) \\ &= \sum_{t=m+1}^{m+k_c-1} \mathbf{h}\left(\mathbf{A}[t], \mathbf{X}\right) + \sum_{t=m+k_c}^{m+k} \mathbf{h}\left(\mathbf{A}[t], \mathbf{Y}\right) \\ &\quad + k\mathbf{e} + (k-k_c)\boldsymbol{\delta}.\end{aligned}$$

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 (14) corresponding to $\mathbf{X}$ and $\mathbf{Y}$, respectively. This results in a CUSUM statistic with mean approximately equal to

$$\begin{aligned}\mathbb{E}\left[\Gamma[m,k]\right] \approx\ & \|k\mathbf{e} + (k-k_c)\boldsymbol{\delta}\|_2^2 \\ & + k_c\|\boldsymbol{\sigma}_X\|_1 + (k-k_c)\|\boldsymbol{\sigma}_Y\|_1. \quad (17)\end{aligned}$$

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 $\omega[k]\mathbb{E}[\Gamma[m,k]] \approx k^{1/2}\|\mathbf{e}+\boldsymbol{\delta}\|_2^2/r$. 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

$$\|\mathbf{e}+\boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2 \Rightarrow 2\|\mathbf{e}\|_2\cos\theta + \|\boldsymbol{\delta}\|_2 > 0, \quad (18)$$

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 (18) 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 (18).

Example 3: Consider a sequence of ER graphs with connection probability p, which at a certain time-step k_c changes to $q = p - \Delta$. In Appendix A we show that the following bound

$$\begin{aligned}&\text{P}\left(\|\mathbf{e}+\boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2\right) \\ &\qquad \geq 1 - \frac{8(1-p)}{\Delta^2 N^2(N-1)m}\left[\frac{1-p}{Nm} + 2(N-1)p\right]\end{aligned}$$

on the probability of satisifying the detectability condition (18) 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 1's performance improves with growing m (the size of the training set) as well as N (the number of nodes).

D. Further Implementation Details

We close this section with some necessary implementation details for Algorithm 1. These pertain to the calculation of the threshold and the possibility of utilizing windowed statistics as alternatives to the the cumulative sum (8).

1) Threshold Calculation: The procedure outlined in Section III-B 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 $\text{th}[k]$ in (16). 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), 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 Appendix A. 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 $1, \dots, m$ 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 $\text{var}[\overline{\mathbf{X}}_{(-j)}\overline{\mathbf{X}}_{(-j)}^\top - \mathbf{P}] = \text{var}[\hat{\mathbf{X}}_j\hat{\mathbf{X}}_j^\top - \mathbf{P}]/(m-1)$ as discussed in Remark 2 and [26], 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 (18) in practice becomes $\|\hat{\mathbf{e}}\|^2 < \|\mathbf{e}+\boldsymbol{\delta}\|_2^2$.

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 3, 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 III-B. 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 (26) and [28] for a modified version where the window length grows proportionally with k. Another useful procedure stems from the exponentially-weighted sum (EWSUM) statistic, namely

$$\mathbf{s}_\beta[m,k] = \sum_{t=m+1}^{m+k}\beta^{m+k-t}\mathbf{h}\left(\mathbf{A}[t],\hat{\mathbf{X}}\right), \tag{19}$$

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, (19) 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 III-B, 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 V.

IV. Directed and Weighted Graphs

A. Directed RDPG

As introduced in Section II, the RDPG model is only suitable for undirected graphs. Indeed, $\mathbf{X}\mathbf{X}^\top = \mathbf{P}$ is always symmetric. For digraphs, edges (or arcs) are defined as *ordered* pairs (i,j), with $i,j\in\mathcal{V}$. 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.

1) Model Specification: Digraphs require an adaptation to the RDPG model, where each node $i\in\mathcal{V}$ has an associated column vector $\mathbf{x}_i$ – now in $\mathbb{R}^{2d}$ [18]. 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 (D-RDPG) model as

$$\mathrm{P}\left(\mathbf{A}\mid\mathbf{X}\right) = \prod_{i\neq j}[(\mathbf{x}_i^l)^\top\mathbf{x}_j^r]^{A_{ij}}[1-(\mathbf{x}_i^l)^\top\mathbf{x}_j^r]^{1-A_{ij}} \tag{20}$$

[cf. the product over all $i\neq j$ here versus $i<j$ in (1)], and the *asymmetric* matrix of connection probabilities now becomes

$$\mathbf{P} = \mathbf{X}^l(\mathbf{X}^r)^\top. \tag{21}$$

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 arc (i,j) is given by $(\mathbf{x}_i^l)^\top\mathbf{x}_j^r$.

Note that the rotational ambiguity is still present. Actually, the ambiguity is exacerbated in this case because *any invertible* matrix $\mathbf{W}\in\mathbb{R}^{d\times d}$ will result in the same $\mathbf{P}$. Indeed, consider $\mathbf{X}^l\mathbf{W}$ and $\mathbf{X}^r\mathbf{W}^{-\top}$ and note that $\mathbf{X}^l\mathbf{W}(\mathbf{X}^r\mathbf{W}^{-\top})^\top = \mathbf{X}^l\mathbf{W}\mathbf{W}^{-1}(\mathbf{X}^r)^\top = \mathbf{X}^l(\mathbf{X}^r)^\top = \mathbf{P}$. Thus, as introduced the D-RDPG model (21) will be challenging to interpret, particularly when it comes to comparing two digraphs via their corresponding embeddings. This last task is critical when it comes to CPD. In order to have roughly the same level of ambiguity as in the undirected RDPG case, we will henceforth require that the d columns of both $\mathbf{X}^l$ and $\mathbf{X}^r$ are orthogonal vectors (i.e., $(\mathbf{X}^l)^\top\mathbf{X}^l$ and $(\mathbf{X}^r)^\top\mathbf{X}^r$ are $d\times d$ diagonal matrices). This extra constraint does not fundamentally limit the expressiveness of the model because $\mathbf{P}$ is still of rank d.

All in all, we are left with the same rotational ambiguity as in the vanilla RDPG model, in addition to a scaling one. Indeed, consider a diagonal matrix $\text{diag}(\boldsymbol{\alpha})$ with non-zero entries and let $\mathbf{W}$ be an orthogonal matrix. Then it follows that $\mathbf{X}^l\mathbf{W}\text{diag}(\boldsymbol{\alpha})$ and $\mathbf{X}^r\mathbf{W}\text{diag}(\boldsymbol{\alpha})^{-1}$ (which still have orthogonal columns) will produce the same $\mathbf{P}$ as (21). Consequently, comparing the magnitude of $\mathbf{x}_i^l$ with that of $\mathbf{x}_i^r$ is meaningless. This scaling ambiguity, which to the best of our knowledge was overlooked before, will challenge CPD if one is interested in the behavior in a single direction (either incoming or outgoing). This is an interesting extension we will leave for future work.

2) D-RDPG Inference: 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 $\{\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r\}$ with orthogonal columns such that $\hat{\mathbf{X}}^l(\hat{\mathbf{X}}^r)^\top$ is the best rank-d approximant of $\mathbf{A}$. Letting $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$ be the singular-value decomposition (SVD) of $\mathbf{A}$, we set

$$\hat{\mathbf{X}}^l = \hat{\mathbf{U}}\hat{\mathbf{D}}^{1/2} \text{ and } \hat{\mathbf{X}}^r = \hat{\mathbf{V}}\hat{\mathbf{D}}^{1/2}. \tag{22}$$

Note that (22) satisfies the required orthogonality constraint. The choice in terms of scaling and counterscaling of columns is arbitrary. Choosing $\hat{\mathbf{D}}^{1/2}$ assumes an even contribution of the incoming and outgoing connectivity of incident nodes to edge generation, which seems reasonable in lieu of any additional prior information. The scaling ambiguity is inconsequential to digraph CPD if we adopt the monitoring function $\mathbf{H}(\mathbf{A}[t], \{\hat{\mathbf{X}}^l, \hat{\mathbf{X}}^r\}) = \hat{\mathbf{X}}^l(\hat{\mathbf{X}}^r)^\top - \mathbf{A}[t]$.

B. Weighted RDPG

We now shift our focus to weighted, undirected graphs. Let us then define a positive weight for each edge through a map $w : \mathcal{E} \mapsto \mathbb{R}_+$ such that $A_{ij} = A_{ji} = w_{ij}$ for $(i, j) \in \mathcal{E}$. The absence of an edge is encoded as $A_{ij} = A_{ji} = 0$. Naturally, an unweighted graph is a particular case of a weighted graph where the edge weights are 0 or 1 (i.e., $w \equiv 1$).

A couple of works have proposed similar adaptations of the vanilla RDPG model to the weighted case; see [19], [20]. The basic ideas therein are outlined next. Suppose that the (possibly weighted) adjacency entries are generated from a given parametric distribution $F_{\boldsymbol{\theta}}(A_{ij})$ with $\boldsymbol{\theta} \in \mathbb{R}^L$, for instance $\theta = \lambda$ for a Poisson(λ) distribution. Each node $i \in \mathcal{V}$ now has L latent vectors $\mathbf{x}_i[l] \in \mathbb{R}^{d_l}$ $(l = 1, \ldots, L)$, such that the weight A_{ij} between nodes i and j is random with parametric distribution $F_{(\mathbf{x}_i^\top[1]\mathbf{x}_j[1],\ldots,\mathbf{x}_i^\top[L]\mathbf{x}_j[L])}(A_{ij})$, independently of all other edges. The distribution may have mass at $A_{ij} = 0$ to capture sparse graphs where some pairs of nodes will be not be joined by edges. One recovers the vanilla RDPG by letting $F_{\boldsymbol{\theta}}(A_{ij})$ be a Bernoulli(θ) distribution.

This approach has several drawbacks. For starters, all edges are required to have the same weight distribution, albeit with different parameters. This limitation may be partially overcome by considering a mixture distribution. However, and limiting even more its applicability, $F_{\boldsymbol{\theta}}(A_{ij})$ has to be chosen *a priori*. So if edges have different weight distributions, we would have to know which of them adhere to each distribution (and what these distributions are) prior to inference.

1) Model Specification: We propose instead that the sequence of vectors associated with each node is related to the moment generating function (MGF) of the weight distribution. In particular, each node has a sequence of latent positions $\mathbf{x}_i[l] \in \mathbb{R}^{d_l}$ that determine the l-th moments of the weighted adjacency matrix as $\mathbb{E}[A_{ij}^l] = \mathbf{x}_i^\top[l]\mathbf{x}_j[l]$, for $l \in \mathbb{N}_+$.Given the sequence $\mathbf{X} := \{\mathbf{X}[l]\}_l$, with $\mathbf{X}[l] = [\mathbf{x}_1[l], \ldots, \mathbf{x}_N[l]]^\top \in \mathbb{R}^{N\times d_l}$, our weighted RDPG (W-RDPG) model specifies the MGF of the adjacency matrix as

$$\mathbb{E}\left[e^{tA_{ij}}|\mathbf{X}\right] = \sum_{l=0}^{\infty} \frac{t^l\mathbb{E}\left[A_{ij}^l\right]}{l!} = 1 + \sum_{l=1}^{\infty} \frac{t^l\mathbf{x}_i^\top[l]\mathbf{x}_j[l]}{l!} \quad (23)$$

and the entries A_{ij} are independent, i.e., edge independent. One can recover the vanilla RDPG by setting $\mathbf{x}_i[l] = \mathbf{x}_i$ for all l, where $\mathbf{x}_i$ is the vector associated to node i in (1).

2) W-RDPG Inference: Vectors $\mathbf{x}_i[l]$ are estimable via an ASE of matrix $\mathbf{A}^{(l)}$, where $\mathbf{A}^{(l)}$ denotes the entry-wise l-th power of adjacency matrix $\mathbf{A}$. The following theorem establishes the consistency (up to an unknown rotation) of this estimator, under some minor eigengap assumptions for the inner product matrices $\mathbf{X}[l]\mathbf{X}^\top[l]$.

Theorem 1: Let $\mathbf{B} \in \mathbb{R}^{N\times N}$ be a random, symmetric, and hollow matrix. Suppose that $0 \leq B_{ij} < M$ for some $M > 0$ and that $\{B_{ij}\}_{i<j}$ are independent with $\mathbb{E}[B_{ij}] = P_{ij}$, where $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$ for some fixed $\mathbf{X} \in \mathbb{R}^{N\times d}$. Suppose that rank$(\mathbf{P}) = d$ and that $\mathbf{P}$ has d distinct positive eigenvalues $\lambda_1 > \lambda_2 > \cdots > \lambda_d > 0$ that satisfy

$$\min_{i\neq j} |\lambda_i - \lambda_j| > \delta N \text{ and } \lambda_d > \delta N$$

for some $\delta > 0$.

Let $\hat{\mathbf{X}} \in \mathbb{R}^{N\times d}$ denote the ASE of $\mathbf{B}$, where it is assumed that the latent space dimension d is known. Then almost surely there exists an orthogonal matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ such that, for each $i \in \{1, \ldots, N\}$ and all $\gamma < 1$,

$$\mathbb{P}\left[||(\hat{\mathbf{X}}\mathbf{W})_{i\cdot} - (\mathbf{X})_{i\cdot}||_2^2 > N^{-\gamma}\right] = o\left(N^{\gamma-1}\log N\right)$$

where $(\mathbf{C})_{i\cdot}$ denotes the i-th row of matrix $\mathbf{C}$.

The relevance to W-DRPG inference is that Theorem 1 can be applied, for each fixed l, to $\mathbf{B} = \mathbf{A}^{(l)}$ to ensure that the latent position matrix $\mathbf{X}[l]$ can be consistently recovered (*modulo* an orthogonal transformation) via the ASE of $\mathbf{A}^{(l)}$.

Theorem 1 is an extension of [32, Theorem 4.1], so in Appendix B we sketch how the proof therein can be adapted to our setting. The main differences with the result in [32] are that in our setting, *a*) latent positions are not random, and *b*) entries B_{ij} of matrix $\mathbf{B}$ are not necessarily Bernoulli random variables; we only assume that they are bounded and their expectation is given by the dot product of the corresponding latent positions. We remark that *b*) is a more general setup than that of [32]. Extending the W-RDPG model to accommodate random latent positions remains an open direction we will pursue as part of our future work.

Example 4: To illustrate the discriminative power of this novel embedding, we consider a two-block weighted SBM graph with $N = 2000$ nodes. Edges are formed with fixed probability $p = 0.5$, but weights are Gaussian with mean $\mu = 5$ and standard deviation $\sigma = 0.1$ for all edges except between a group of 1000 nodes indexed as $i = 1001, \ldots, 2000$, where the weights' distribution is Poisson with parameter $\lambda = 5$. As discussed in Example 1, in this case matrix $\mathbf{X}[l]$ will have at most 2 different columns for all l. The vectors $\hat{\mathbf{x}}_i[l]$ corresponding to the ASE for $l = 1, 2, 3$ and $d_l = 2$ are shown in Fig. 1, where each community is colored differently.

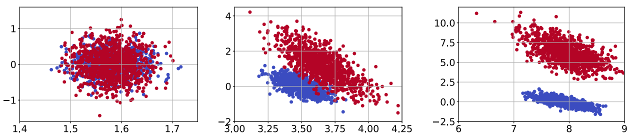


Fig. 1. ASE embedding of $\mathbf{A}^{(l)}$ for Gaussian ($\mu = 5$ and $\sigma = 0.1$; in blue) and Poisson ($\lambda = 5$; in red) distributed weights for $d_l = 2$ and $l = 1$ (left), $l = 2$ (center), and $l = 3$ (right). Nodes with different weight distributions are clearly revealed for $l = 3$, but they overlap for $l = 1$.

Note how the nodes are indistinguishable for $l = 1$. Indeed, the $\hat{\mathbf{x}}_i[1]$ vectors are, as expected, centered around $(\sqrt{\mu p}, 0) = (\sqrt{\lambda p}, 0) \approx (1.58, 0)$ corresponding to the mean weight. For $l = 2$, Fig. 1 (center) shows the vectors start to separate into the corresponding communities. Expressions for higher-order embeddings are easily obtained for this toy example. For instance, arbitrarily assuming that the $\mathbf{x}_i[l]$ lie on the abscissa for

$i = 1, \ldots, 1000$ (recall that any rotation of $\mathbf{X}[l]$ will result in the same expected weights), it thus follows

$$\mathbf{x}_i[2] = \begin{cases} (\sqrt{p(\mu^2+\sigma^2)}, 0) & i \leq 1000, \\ (\sqrt{p(\mu^2+\sigma^2)}, \sqrt{p}(\lambda^2+\lambda-(\mu^2+\sigma^2)) & i > 1000. \end{cases}$$

Indeed, the estimates are around (3.55,0) and (3.55,1.58) respectively, although the noise corrupting the estimates hinders the ability to distinguish both distributions. For $l = 3$, where the skewness of the distribution comes into play, vectors are clearly separated into the two groups; see Fig. 1 (right).

C. Online Change-Point Detection

Let us briefly discuss how to perform online CPD for the general weighted and/or directed case. Extending the results presented in Section III 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 (7).

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 W-RDPG model we introduced in the previous section, 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 (13) and (14)]. One can seamlessly blend the ideas in Sections IV-A and IV-B 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 prundent when interested in more fine-grained changes on the weights' distribution, as illustrated in Example 4.

V. 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 V-A). 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 V-B. For the implementations we used the Python libraries `NumPy` [33], `NetworkX` [34], `pandas` [35], `graspologic` [36], 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 [11], 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 [17]. This implementation is also available in our GitHub repository.

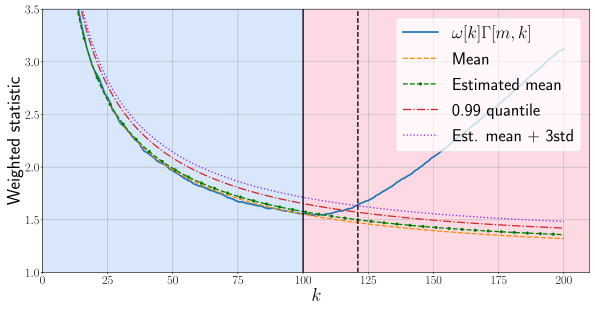


Fig. 2. Evolution of $\omega[k]\Gamma[m,k]$, its mean and the estimated mean, for simulated data. Two thresholds are shown: the 0.99-quantile of the distribution in (15) and three standard deviations away from the mean; those thresholds are very close and the latter is preferred due to its reduced complexity. 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 [17]. Our approach is able to detect the change with a relatively small delay, while operating in an online fashion.

A. 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 scree plot) by the `graspologic` library used to obtain the ASE. Because the index k in $\Gamma[m,k]$ measures how much time has elapsed since monitoring started, the change-point is at $k_c = 100$.

Fig. 2 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 (15) but with $\hat{\mathbf{e}}$ instead of $\mathbf{e}$] and th$[k]$, the estimated mean plus three standard deviations. Apparently, the difference between those two thresholds is small, so 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 (8), 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 III-D. As a way to compare the performance of Algorithm 1 with other approaches, Fig. 2 also shows the detection result for the offline baseline proposed in [17]. 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.

1) On the choice of the monitoring function: In this running example, had we used the monitoring function $\mathbf{H}' = (\hat{\mathbf{X}}\hat{\mathbf{X}}^\top -$

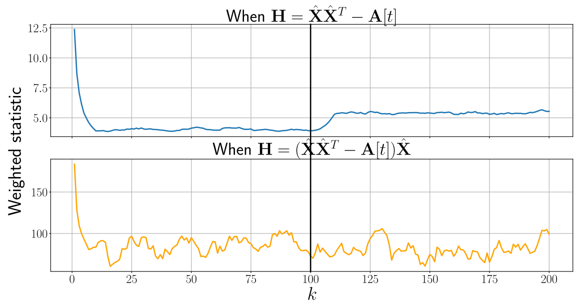


Fig. 3. Evolution of $\omega[k]\Gamma[m,k]$ for residual (top) and projection (bottom) monitoring functions, using the MOSUM sliding window statistic. After the change-point there is a discernible change in trend for the residual; the projection does not exhibit such desirable behavior.

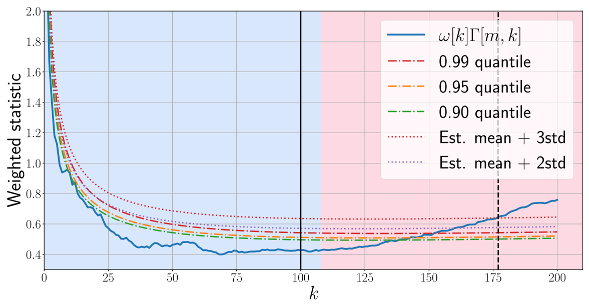


Fig. 4. Evolution of $\omega[k]\Gamma[m,k]$ and five possible thresholds: $c_\alpha[k]$ (for $1-\alpha \in \{0.9, 0.95, 0.99\}$) and $\text{th}[k]$ equal to the mean plus two and three standard deviations. The setting is the same as in Fig. 2 except that $N = 20$ to increase the variance of $\omega[k]\Gamma[m,k]$. Using $1-\alpha = 0.99$ is preferred as it provides more robustness to false positives. Both choices of $\text{th}[k]$ are reasonable, although using three standard deviations is consistently above $c_{0.01}[k]$ (see the first time-steps).

$\mathbf{A}[t])\hat{\mathbf{X}}$ (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 $\hat{\mathbf{X}} = \sqrt{p}\mathbf{J}_{N\times d}$ and $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top = p\mathbf{J}_N$, with $\mathbf{J}_{N\times d}$ denoting the $N \times d$ all-ones matrix. 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

$$\mathbf{P}_{\text{SBM}} = \left(\begin{array}{c|c} \mathbf{Q}_1 & \mathbf{Q}_2 \\ \hline \mathbf{Q}_2 & \mathbf{Q}_1 \end{array}\right), \tag{24}$$

where $\mathbf{Q}_1 = q_1(\mathbf{J}_{N/2} - \mathbf{I}_{N/2})$ and $\mathbf{Q}_2 = q_2\mathbf{J}_{N/2}$, with $\mathbf{I}_m$ denoting the identity matrix of size m. After the change we thus have $\mathbb{E}[\mathbf{H}'] = (p\mathbf{J}_N - \mathbf{P}_{\text{SBM}})\sqrt{p}\mathbf{J}_{N\times d}$. 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

$$\begin{aligned}(\mathbb{E}\left[\mathbf{H}'\right])_{ij} &= \left(\frac{N}{2} - 1\right)\sqrt{p}(p - q_1) + \frac{N}{2}\sqrt{p}(p - q_2) \\ &\approx N\sqrt{p}\left(p - \frac{q_1 + q_2}{2}\right),\end{aligned}$$

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.

Fig. 3 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.

2) On the sensitivity to α: Here we examine the robustness of Algorithm 1 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 $\text{th}[k]$: the mean plus two and three times the standard deviations as in (16).

The results are depicted in Fig. 4. 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 $\text{th}[k]$ 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 V-B, when we present real-world examples.

3) Comparison with [11]: An online CPD algorithm based on a k-nearest neighbor approach was proposed in [11]. 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 $t \in \{1, \ldots, n-1\}$ (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 [11] 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

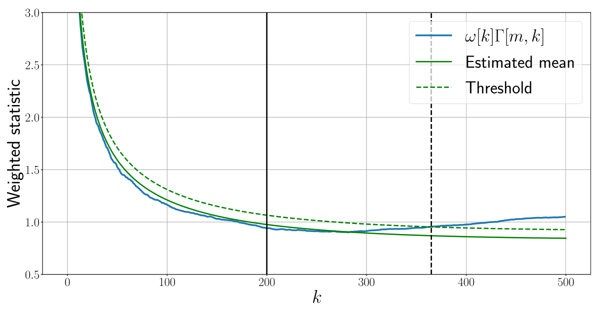


Fig. 5. Detection result for a network transitioning from an ER model with $p = 0.3$ to a two-block SBM with $q_1 = 0.275$ and $q_2 = 0.325$. Algorithm 1 is able to detect the change in this setup, while the approach proposed in [11] fails to do so.

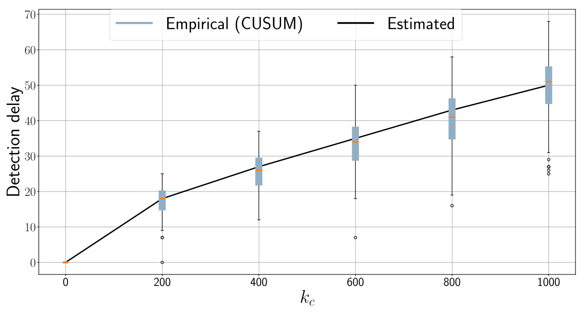


Fig. 6. Estimated detection delay and empirical delays for different change-point locations k_c. Empirical delay is well predicted by the estimated curve. For the adopted CUSUM statistic, as expected the delay grows with k_c.

$\mathbf{D}$ is given by (24), 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

$$\|\mathbf{C}-\mathbf{D}\|_F^2 \approx N^2\left(q_1 - q_1^2 + q_2 - q_2^2\right),$$

$$\|\mathbf{A}-\mathbf{C}\|_F^2 \approx N^2\left(p - p(q_1+q_2) + \frac{q_1+q_2}{2}\right).$$

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

$$\|\mathbf{C}-\mathbf{D}\|_F^2 \approx 2\,N^2\left(p - p^2 - (p-q_1)^2\right),$$

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 [11] 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 Fig. 5. 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.

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. (17)] first exceeds the threshold. Once more, for simplicity and analytical tractability we will henceforth assume the threshold is set as $\text{th}[k]$ in (16). This choice (approximately) corresponds to $\alpha = 0.01$; see Figs. 2 and 4 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 (17). This amounts to solving the equation

$$(k^*-k_c)^2\left(\|\boldsymbol{\sigma}_Y\|_1 - \|\boldsymbol{\sigma}_X\|_1 + 2\,k^*(\mathbf{e}^\top\boldsymbol{\delta}) + (k^*-k_c)\|\boldsymbol{\delta}\|_2^2\right)^2$$
$$= 9\left(2(k^*)^2\|\boldsymbol{\sigma}_X\|_2^2 + 4(k^*)^3(\boldsymbol{\sigma}_X^\top\mathbf{e}^2)\right), \tag{25}$$

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 (25) 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. Fig. 6 shows the estimated delay for $k_c \in \{0, 200, 400, 600, 800, 1000\}$. For each k_c a box plot of Algorithm 1'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 1, which tend to show a linear growth with k_c.

Fig. 7 depicts the empirical delays in this setup for three different statistics: CUSUM, MOSUM and mMOSUM. This last running statistic is defined in [28] as

$$\mathbf{s}[m,k] = \sum_{t=m+\lfloor kh\rfloor+1}^{m+k} \mathbf{h}\left(\mathbf{A}[t], \hat{\mathbf{X}}\right), \tag{26}$$

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 Fig. 7, 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

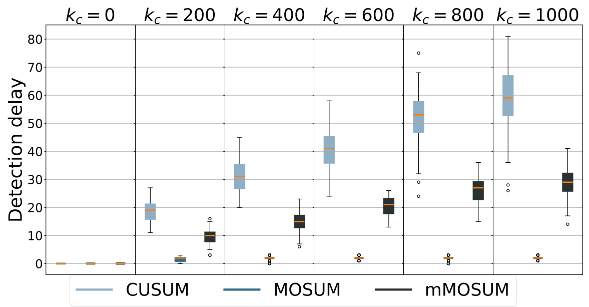


Fig. 7. Empirical delays for different change-point locations k_c, using the CUSUM, MOSUM (with $L = 10$), and mMOSUM (with $h = 0.4$) statistics. Delays behave as expected given the different effective observation intervals: roughly constant delay for MOSUM, growing delays with k_c for both CUSUM and mMOSUM, but at a slower rate for the latter.

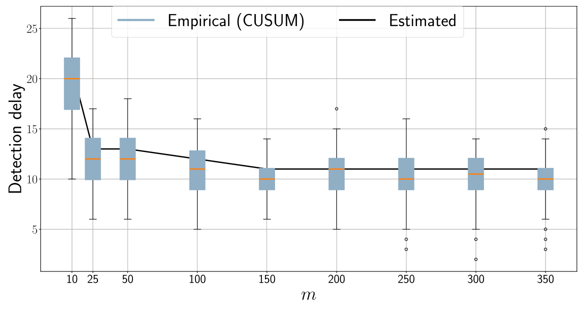


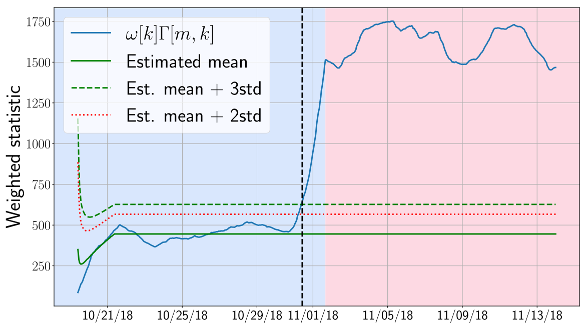
Fig. 8. Estimated detection delay and empirical delays for different training set sizes m, for the CUSUM statistic. The delay is lower as m increases, but there is no significant improvement after $m = 25$.

$L = 10$ 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, Fig. 8 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 1 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 $\tau = 13$ to $\tau = 11$ for $m = 300$).

B. Real Data Experiments

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 [37]. 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



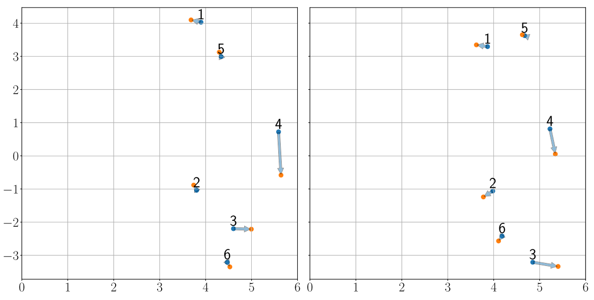
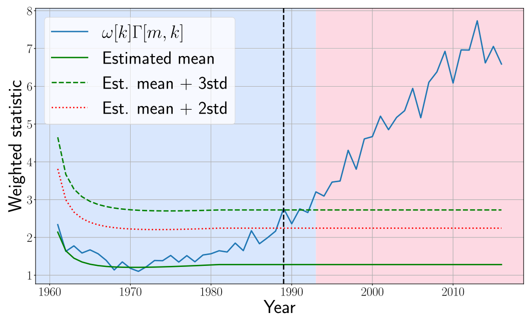


Fig. 9. Online CPD for the RSSI dataset. Top: MOSUM statistic. A change in background color indicates a change-point detected by the offline algorithm [17]. The dashed vertical line shows the detected change-point for the online algorithm. Algorithm 1 successfully detects that an AP was moved. Bottom, 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.

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 Fig. 9 (top). Note how Algorithm 1 rapidly detects the AP movement. The offline CPD baseline in [17] is also able to detect the change, at around the same date. Furthermore, we complement the threshold studies carried out in Section V-A 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 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 [27], 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



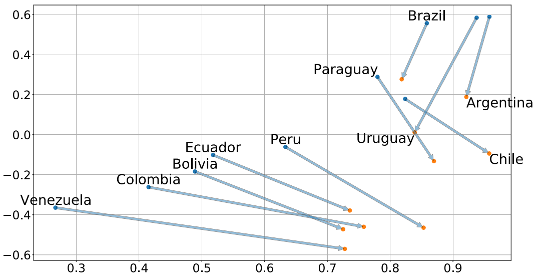


Fig. 10. Online CPD for the South American football matches. Top: evolution of MOSUM statistic. The dashed vertical line shows the detected change-point, that can be traced to a change in the *Copa América* organization format. A change in background color indicates a change-point detected by the offline algorithm [17]. 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.

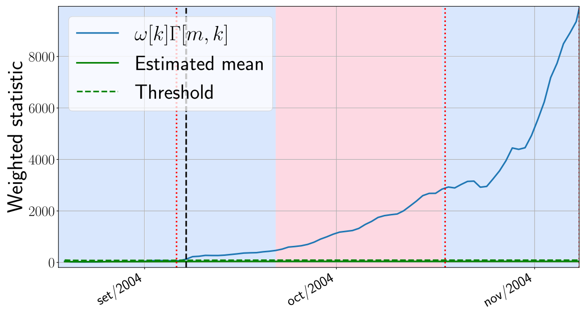


Fig. 11. Online CPD for the MIT proximity dataset (using the MOSUM window). A change in background color indicates a change-point detected by the offline algorithm of [17]. The dashed vertical line shows the detected change-point for the online algorithm. Dotted vertical lines indicate the beginning of the semester and the "sponsor week". The offline algorithm misses the first change-point.

(two here), as the size of $\mathbf{M}$ increases rapidly with the number of matrices considered.

Nodal vectors ($d = 2$) are depicted in Fig. 9 (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 Section IV-B 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.

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 (data obtained from https://www.eloratings.net/). We used the first $m = 20$ years for training and the evolution of the resulting weighted CUSUM statistic is shown in Fig. 10 (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 Fig. 10 (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 Fig. 10). But as mentioned in Section V-A, using the mean plus three standard deviations clearly provides more robustness to false positives, particularly in high noise settings as in this test case.

3) MIT Proximity Network: Lastly, let us consider the stream of social graphs introduced in [38]. 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. 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.[2] A collection of labeled events are described in [7, 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 Fig. 11. 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 [17] (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

[2]We used Jeremy Kun's scripts in https://github.com/j2kun/reality-mining.

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.

VI. Conclusion and Future Work

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. Generalizations of the RDPG model to directed and weighted graphs markedly broaden the applicability of the novel online CPD framework, as illustrated through various real-data case studies.

This work opens up several exciting and challenging avenues for future work. For instance, while still relying on RDPG modeling it would be of interest to explore sequential CPD formulations that minimize (or provide an explicit handle on) detection delay. Even in the present setting, carrying out a rigorous delay analysis would constitute a valuable contribution. In all fairness, accomplishing this goal would be central towards fully solving the online change-point detection problem. With regards to ASE-induced model estimation error, although in this work we presented a simple yet effective "leave-one-out" approach to approximate its value, a worthwhile future direction in our agenda is the study of theoretical bounds and guarantees for this plug-in statistic. Our methodology detects changes in the model with respect to a training set of nominal graphs, and assumes that the number of nodes in the network does not change. Depending on the particular application, it may be interesting to consider the case where certain nodes are not always present on the network, and we are interested in only a subset of them. Along these lines, we believe it would be worthwhile to develop embedding and CPD algorithms for partially observed graph streams, say due to sampling. Lastly, one could also envision online CPD schemes using just graph signal observations, because ASE-type embeddings are likely still computable from empirical signal covariance matrices under diffusion model assumptions.

Appendix

A. Proof of the Bound in Example 3

A sequence of ER graphs with connection probability p changes to $q = p - \Delta$ at a certain time-step. Equation $\|\mathbf{e} + \boldsymbol{\delta}\|_2^2 > \|\mathbf{e}\|_2^2$ in this case may be written as

$$2\sum_{i=1}^{N}\sum_{j=i+1}^{N} E_{ij} > -\Delta\frac{N(N-1)}{2}, \tag{27}$$

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 (27) as

$$\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}} > \left(p - \frac{\Delta}{2}\right)N(N-1). \tag{28}$$

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 $\boldsymbol{\Sigma} = \frac{(1-p)}{Nm}\mathbf{I}$ [26], [39], [40], we consider this asymptotic regime and use results about the statistics of quadratic forms of Gaussian vectors [41, Ch. 5]. For instance, the resulting mean is

$$\begin{aligned}\mathbb{E}[\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}}] &= \text{tr}\,[(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma}] + \boldsymbol{\mu}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\mu} \\ &= pN(N-1).\end{aligned}$$

Comparing the equation above to (28), 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 $\sigma^2 := (1-p)/(Nm)$)

$$\begin{aligned}\text{var}[\hat{\mathbf{x}}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\hat{\mathbf{x}}] &= 2\text{tr}\left[((\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma})^2\right] \\ &\quad + 4\boldsymbol{\mu}^\top(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\Sigma}(\mathbf{1}_{N\times N} - \mathbf{I})\boldsymbol{\mu} \\ &= 2\sigma^2 N(N-1)(\sigma^2 + 2(N-1)p).\end{aligned}$$

Applying Chebyshev's inequality, the result follows. ■

B. Proof Sketch for Theorem 1

We now give an overview of the necessary steps to prove Theorem 1. We adapt the arguments used in [32] to accommodate our setting; therefore, we will outline how their proof can be adapted to our case.

The following notation will be used throughout this section. We will denote the eigendecomposition of matrix $\mathbf{B} \in \mathbb{R}^{N\times N}$ as $\mathbf{V}_B\boldsymbol{\Lambda}_B\mathbf{V}_B^\top$, with the elements in the diagonal of $\boldsymbol{\Lambda}_B$ in decreasing order. $\hat{\boldsymbol{\Lambda}}_B \in \mathbb{R}^{d\times d}$ will denote the diagonal matrix with the d largest eigenvalues of $\mathbf{B}$, and $\hat{\mathbf{V}}_B \in \mathbb{R}^{N\times d}$ will be the corresponding d dominant eigenvectors. Recall we assume that $0 \leq B_{ij} < M$, for some $M > 0$, and that $\{B_{ij}\}_{i<j}$ are independent with $\mathbb{E}[B_{ij}] = P_{ij}$, where $\mathbf{P} = \mathbf{X}\mathbf{X}^\top$ for some fixed $\mathbf{X} \in \mathbb{R}^{N\times d}$. The eigendecomposition of $\mathbf{P}$ is $\mathbf{V}_P\boldsymbol{\Lambda}_P\mathbf{V}_P^\top$. Matrices $\hat{\boldsymbol{\Lambda}}_P$ and $\hat{\mathbf{V}}_P$ are similarly defined for $\mathbf{P}$.

The first step of the proof is to show that, for large N, it almost surely holds that:

$$\|\mathbf{B}^2 - \mathbf{P}^2\|_F < \sqrt{3\,M^4\,N^3\log N},$$

where $\mathbf{B}^2 = \mathbf{B}\times\mathbf{B}$ denotes the usual matrix product. The argument (in a more general setting) can be found in [42, Lemma 2]. In our setting, the basic idea is to write

$$\mathbf{B}_{ij}^2 - \mathbf{P}_{ij}^2 = \sum_{k\neq ij}(\mathbf{B}_{ik}\mathbf{B}_{kj} - \mathbf{P}_{ik}\mathbf{P}_{kj}) - \mathbf{P}_{ii}\mathbf{P}_{ij} - \mathbf{P}_{ij}\mathbf{P}_{jj}.$$

Since $\mathbf{B}_{ik}\mathbf{B}_{kj}$ are independent for $k \neq i, j$ and $\mathbf{P}_{ij}$'s are bounded by M, we use Hoeffding's inequality to show that

$$\text{P}\left(\left(\mathbf{B}_{ij}^2 - \mathbf{P}_{ij}^2\right)^2 \geq 2\,M^4(N-2)\log N + M^4(4N-4)\right) \leq \frac{2}{N^4}.$$

Then, using the subadditivity property of probability and the Borel-Cantelli Lemma, we can show that almost always

$$\sum_{i,j:i\neq j}\left(\mathbf{B}_{ij}^2 - \mathbf{P}_{ij}^2\right)^2 \leq \frac{5}{2}M^4\,N^3\log N.$$

Since $(\mathbf{B}_{ii}^2 - \mathbf{P}_{ii}^2)^2 \leq M^4$, we finally conclude that

$$\|\mathbf{B}^2 - \mathbf{P}^2\|_F^2 \leq \frac{5}{2} M^4 N^3 \log N + NM^4 < 3\, M^4 N^3 \log N$$

for sufficiently large N.

Once this is established, we apply a variant of the Davis-Kahan theorem to $\mathbf{B}^2$ and $\mathbf{P}^2$. Since the eigenvectors of $\mathbf{B}$ and $\mathbf{B}^2$ coincide (the same is true for $\mathbf{P}$ and $\mathbf{P}^2$) and we assume the eigengap for $\mathbf{P}$ is greater than δN (and thus the eigengap for $\mathbf{P}^2$ is greater than $\delta^2 N^2$), [43, Corollary 3] ensures that it is possible to choose the columns of $\mathbf{V}_B$ such that

$$\|(\mathbf{V}_B)_{\cdot i} - (\mathbf{V}_P)_{\cdot i}\|_2 \leq \frac{2^{3/2}}{\delta^2} \sqrt{3\, M^4 \frac{\log N}{N}},$$

for every $i \leq d$, where $(\mathbf{V}_B)_{\cdot i}$ denotes the i-th column of matrix $\mathbf{V}_B$. Since the first d columns of $\mathbf{V}_B$ are the columns of $\hat{\mathbf{V}}_B$ (the same is true for $\mathbf{V}_P$ and $\hat{\mathbf{V}}_P$) this in turn implies that, for such a choice,

$$\|\hat{\mathbf{V}}_B - \hat{\mathbf{V}}_P\|_F \leq C\sqrt{d}\sqrt{\frac{\log N}{N}},$$

where C is a constant.

The rest of the proof follows, *mutatis mutandis*, that of [32]. First, by writing

$$\|\hat{\mathbf{V}}_B \hat{\mathbf{\Lambda}}_B^{1/2} - \hat{\mathbf{V}}_P \hat{\mathbf{\Lambda}}_P^{1/2}\|_F \leq \|\hat{\mathbf{V}}_B (\hat{\mathbf{\Lambda}}_B^{1/2} - \hat{\mathbf{\Lambda}}_P^{1/2})\|_F + \|(\hat{\mathbf{V}}_B - \hat{\mathbf{V}}_P)\hat{\mathbf{\Lambda}}_P^{1/2}\|_F$$

using the previous bounds we can show that

$$\|\hat{\mathbf{V}}_B \hat{\mathbf{\Lambda}}_B^{1/2} - \hat{\mathbf{V}}_P \hat{\mathbf{\Lambda}}_P^{1/2}\|_F \leq Cd\sqrt{\log N}. \tag{29}$$

Because $\text{rank}(\mathbf{P}) = d$, by defining $\mathbf{Y} := \hat{\mathbf{V}}_P \hat{\mathbf{\Lambda}}_P^{1/2}$ we have that $\mathbf{Y}\mathbf{Y}^\top = \mathbf{P} = \mathbf{X}\mathbf{X}^\top$, so $\mathbf{X} = \mathbf{Y}\mathbf{W}$ for some orthogonal $\mathbf{W}$. Thus, the bound in (29) also holds for $\|\hat{\mathbf{V}}_B \hat{\mathbf{\Lambda}}_B^{1/2}\mathbf{W} - \mathbf{X}\|_F$. Since the ASE estimation of the latent positions is $\hat{\mathbf{X}} = \hat{\mathbf{V}}_B \hat{\mathbf{\Lambda}}_B$, this implies that almost surely

$$\|\hat{\mathbf{X}}\mathbf{W} - \mathbf{X}\|_F \leq Cd\sqrt{\log N}.$$

The proof then concludes as in [32]. ■

References

[1] F. Larroca, P. Bermolen, M. Fiori, and G. Mateos, "Change-point detection in weighted and directed random dot product graphs," in *Proc. Eur. Signal Process. Conf.*, 2021, pp. 1810–1814.

[2] B. Marenco, F. Larroca, P. Bermolen, M. Fiori, and G. Mateos, "Online change-point detection for random dot product graphs," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, 2021.

[3] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[4] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Process.*, vol. 167, 2020, Art. no. 107299.

[5] X. He, Y. Xie, S.-M. Wu, and F.-C. Lin, "Sequential graph scanning statistic for change-point detection," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, 2018, pp. 1317–1321.

[6] H. Keshavarz, G. Michaildiis, and Y. Atchade, "Sequential change-point detection in high-dimensional Gaussian graphical models," *J. Mach. Learn. Res.*, vol. 21, no. 82, pp. 1–57, 2020.

[7] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," in *Proc. AAAI Conf. Artificial Intell.*, 2015, pp. 2914–2920.

[8] C. Kaushik, T. M. Roddenberry, and S. Segarra, "Network topology change-point detection from graph signals with prior spectral signatures," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5395–5399.

[9] M. Zhang, L. Xie, and Y. Xie, "Online community detection by spectral cusum," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 3402–3406.

[10] A. Ferrari, C. Richard, and L. Verduci, "Distributed change detection in streaming graph signals," in *Proc. IEEE Workshop Comput. Adv. Multi-Sensor Adaptive Process.*, 2019, pp. 166–170.

[11] H. Chen, "Sequential change-point detection based on nearest neighbors," *Ann. Stat.*, vol. 47, no. 3, pp. 1381–1407, 2019. [Online]. Available: https://doi.org/10.1214/18-AOS1718

[12] H. Wang, M. Tang, Y. Park, and C. E. Priebe, "Locality statistics for anomaly detection in time series of graphs," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 703–717, Feb. 2014.

[13] Y. Yu, O. H. M. Padilla, D. Wang, and A. Rinaldo, "Optimal network online change point localisation," 2021, *arXiv:2101.05477*.

[14] S. J. Young and E. R. Scheinerman, "Random dot product graph models for social networks," in *Algorithms and Models for the Web-Graph*, A. Bonato and F. R.K. Chung, Eds., Berlin, Germany: Springer, 2007, pp. 138–149.

[15] A. Athreya *et al.*, "Statistical inference on random dot product graphs: A survey," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 8393–8484, Jan. 2017.

[16] C. Kirch and J. Tadjuidje Kamgaing, "On the use of estimating functions in monitoring time series for change points," *J. Statist. Plan. Inference*, vol. 161, pp. 25–49, 2015.

[17] O. H. M. Padilla, Y. Yu, and C. E. Priebe, "Change point localization in dependent dynamic nonparametric random dot product graphs," 2019, *arXiv:1911.07494*.

[18] C. E. Priebe *et al.*, "Semiparametric spectral modeling of the drosophila connectome," 2017, *arXiv:1705.03297*.

[19] R. Tang, M. Tang, J. T. Vogelstein, and C. E. Priebe, "Robust estimation from multiple graphs under gross error contamination," 2017, *arXiv:1707.03487*.

[20] D. R. DeFord and D. N. Rockmore, "A random dot product model for weighted networks," 2016, *arXiv:1611.02530*.

[21] E. D. Kolaczyk, *Topics at the Frontier of Statistics and Network Analysis: (Re)Visiting the Foundations, Ser. SemStat Elements*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[22] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *J. Amer. Statist. Assoc.*, vol. 97, no. 460, pp. 1090–1098, 2002.

[23] E. Scheinerman and K. Tucker, "Modeling graphs using dot product representations," *Comput. Stat.*, vol. 25, pp. 1–16, 2010.

[24] P. Rubin-Delanchy, J. Cape, M. Tang, and C. E. Priebe, "A statistical interpretation of spectral embedding: The generalised random dot product graph," 2017, *arXiv:1709.05506*.

[25] M. Zhu and A. Ghodsi, "Automatic dimensionality selection from the scree plot via the use of profile likelihood," *Comput. Stat. Data. Anal.*, vol. 51, no. 2, pp. 918–930, 2006.

[26] R. Tang *et al.*, "Connectome smoothing via low-rank approximations," *IEEE Trans. Med. Imag.*, vol. 38, no. 6, pp. 1446–1456, Dec. 2018.

[27] K. Levin, A. Athreya, M. Tang, V. Lyzinski, and C. E. Priebe, "A central limit theorem for an omnibus embedding of multiple random dot product graphs," in *Proc. Int. Conf. Data Mining Workshops*, 2017, pp. 964–967.

[28] C. Kirch and S. Weber, "Modified sequential change point procedures based on estimating functions," *Electron. J. Statist.*, vol. 12, no. 1, pp. 1579–1613, 2018.

[29] V. Bentkus, "On the dependence of the berry-esseen bound on dimension," *J. Stat. Plan. Inference*, vol. 113, no. 2, pp. 385–402, 2003.

[30] J. P. Imhof, "Computing the distribution of quadratic forms in normal variables," *Biometrika*, vol. 48, no. 3/4, pp. 419–426, 1961.

[31] M. Sankaran, "Approximations to the non-central chi-square distribution," *Biometrika*, vol. 50, no. 1/2, pp. 199–204, Jun. 1963.

[32] D. L. Sussman, M. Tang, and C. E. Priebe, "Consistent latent position estimation and vertex classification for random dot product graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 1, pp. 48–57, Jan. 2014.

[33] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, pp. 357–362, 2020.

[34] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proc. 7th Python in Sci Conf.*, G. Varoquaux, T. Vaught, and J. Millman, Eds., Pasadena, CA, USA, 2008, pp. 11–15.

[35] W. McKinney, "Data structures for statistical computing in python," *Proc. 9th Python Sci. Conf.*, vol. 445, no. 1, pp. 56–61, 2010.

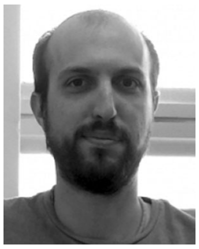
[36] J. Chung, B. D. Pedigo, E. W. Bridgeford, B. K. Varjavand, H. S. Helm, and J. T. Vogelstein, "Graspy: Graph statistics in python," *J. Mach. Learn. Res.*, vol. 20, no. 158, pp. 1–7, 2019.

[37] G. Capdehourat, F. Larroca, and G. Morales, “A nation-wide Wi-Fi RSSI dataset: Statistical analysis and resulting insights,” in *Proc. IFIP Netw. Conf. (Netw.)*, 2020, pp. 370–378.
[38] N. Eagle and A. Sandy Pentland, “Reality mining: Sensing complex social systems,” *Pers. Ubiquitous Comput.*, vol. 10, no. 4, pp. 255–268, Mar. 2006.
[39] A. Athreya, C. E. Priebe, M. Tang, V. Lyzinski, D. J. Marchette, and D. L. Sussman, “A limit theorem for scaled eigenvectors of random dot product graphs,” *Sankhya A*, vol. 78, no. 1, pp. 1–18, 2016.
[40] P. BourgadeJ. Huang, and H.-T. Yau, “Eigenvector statistics of sparse random matrices,” *Electron. J. Probability*, vol. 22, pp. 1–38, 2017. [Online]. Available: https://doi.org/10.1214/17-EJP81
[41] A. C. Rencher and G. B. Schaalje, *Linear Models in Statistics*. Hoboken, New Jersey: Wiley, 2008.
[42] D. E. Fishkind, D. L. Sussman, M. TangJ. T. Vogelstein, and C. E. Priebe, “Consistent adjacency-spectral partitioning for the stochastic block model when the model parameters are unknown,” *SIAM J. Matrix Anal. Appl.*, vol. 34, no. 1, pp. 23–39, 2013. [Online]. Available: https://doi.org/10.1137/120875600
[43] Y. Yu, T. Wang, and R. J. Samworth, “A useful variant of the Davis-Kahan theorem for statisticians,” *Biometrika*, vol. 102, no. 2, pp. 315–323, 2015.

Bernardo Marenco received the B.Sc. degree in electrical engineering and the M.Sc. degree in mathematical engineering from the Universidad de la República (UdelaR), Montevideo, Uruguay, in 2015 and 2019, respectively. He is currently working toward the Ph.D. degree in mathematics with PEDECIBA, Uruguay, under the supervision of Dr. Paola Bermolen and Dr. Gonzalo Mateos. He is currently a Teaching Assistant with Mathematical Department, School of Engineering, UdelaR. His research interests include statistical inference and network modeling.

Paola Bermolen received the degree in Mathematics from the Universidad de la República, Montevideo, Uruguay, in 2004, and the Ph.D. degree in computer science and networks from Telecom ParisTech, Paris, France, in 2010. In 1998, she joined the School of Engineering, Universidad de la República, where she is currently an Associate Professor with Mathematical Department. She works in Probability and Statistics. Her research interests include stochastic modeling of networks in telecommunications and other applications domains such as ecology. Her current research interests include the performance analysis of wireless networks, including stochastic geometry, random graphs models, and large deviations theory. She is co-responsible for the Project CICADA Interdisciplinary Center in Data Science and Machine Learning. She is a Researcher with the Basic Sciences Development Program – PEDECIBA Mathematics and a Member of the National Researchers System, Uruguay.

Marcelo Fiori received the B.Sc. degree in electrical engineering, the M.Sc. in mathematical engineering, and the Ph.D. in electrical engineering from Universidad de la República, Montevideo, Uruguay, in 2008, 2011, and 2015, respectively. From 2010 to 2015, he held different visiting positions with the University of Minnesota, Minneapolis, MN, USA, and Duke University, Durham, NC, USA. He is currently an Assistant Professor with the Mathematics Department of the Engineering School, Universidad de la República. His main research interests include spectral graph theory, graph isomorphism and graph matching problems, and graph representation and optimization.



Federico Larroca received the degree in telecommunication engineering from the Universidad de la República, Montevideo, Uruguay, in 2006, and the Ph.D. degree in computer science and networking from Telecom ParisTech, Paris, France, in December 2009, under the advisoring of Prof. Jean-Louis Rougier. He is currently an Assistant Professor with the Engineering School of the Universidad de la República. He was a Research Engineering (Postdoc) with Telecom ParisTech (formerly ENST) from January to March 2010. From 2004 to 2011, he held a Teaching Assistant position with the Universidad de la República. His research interests include the analysis and modeling of communication systems, and development on software defined radio.

Gonzalo Mateos (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the Universidad de la República, Montevideo, Uruguay, in 2005, and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Minnesota, Minneapolis, MN, USA, in 2009 and 2011, respectively. From 2004 to 2006, he was a Systems Engineer with Asea Brown Boveri, Uruguay. During 2013, he was a Visiting Scholar with the Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA, and an Asaro Biggar Family Fellow in data science. His research interests include statistical learning from complex data, network science, decentralized optimization, and graph signal processing.