Online Change Point Detection for Random Dot Product Graphs

Bernardo Marenco*, Paola Bermolen*, Marcelo Fiori*, Federico Larroca*, and Gonzalo Mateos†
*Facultad de Ingeniería, Universidad de la República, Uruguay
Email: {bmarenco,paola,mfiori,flarroca}@fing.edu.uy
†Dept. of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA.
Email: gmateosb@ur.rochester.edu

***Abstract*—Given a sequence of random graphs, we address the problem of online monitoring and detection of changes in the underlying data distribution. To this end, we adopt the Random Dot Product Graph (RDPG) model which postulates each node has an associated latent vector, and inner products between these vectors dictate the edge formation probabilities. Existing approaches for graph change-point detection (CPD) rely either on extensive computation, or they store and process the entire observed time series. In this paper we consider the cumulative sum of a judicious monitoring function, which quantifies the discrepancy between the streaming graph observations and the nominal model. This reference distribution is inferred via spectral embeddings of the first few graphs in the sequence, and the monitoring function can be updated in an efficient, online fashion. We characterize the distribution of this running statistic, allowing us to select appropriate thresholding parameters that guarantee error-rate control. The end result is a lightweight online CPD algorithm, with a proven capability to flag distribution shifts in the arriving graphs. The novel method is tested on both synthetic and real network data, corroborating its effectiveness in quickly detecting changes in the input graph sequence.**

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

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. This is a classic statistical signal processing problem that can be traced back to the seminal work of Page [1], where, in the context of quality control, sought to detect a change on the distribution of an univariate random variable (r.v.). The challenge is to flag a problem (in order to take corrective actions) as soon as it happens, while controlling the probability of false alarm. Unlike the offline or batch case (see e.g., [2]), here we do not have access to the full data sequence (which may actually 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 [3], financial markets [4], or social networks [5], [6]. 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 (unweighted and undirected) graphs, we would like to signal if and when the data generating process changes.

This work was partially funded by ANII (grant FMV_3_2018_1_148149) and the NSF (awards CCF-1750428 and ECCS-1809356).

Related work. To address this problem, a non-parametric approach was developed in [7], where the only requirement is to define a pairwise distance between samples (e.g., the Frobenius distance between adjacency matrices is adopted in the case of graphs). Then, a k-nearest-neighbors-based statistic is calculated sequentially using the incoming data. As we argue in the comparisons of Sec. IV, said distance is prone to overlooking simple changes that generative models easily capture. Noteworthy model-based efforts include [5], which considered the Generalized Hierarchical Random Graph (GHRG) model as generating mechanism. A computationally expensive posterior Bayes factor is calculated for all partitions of the data over a sliding window. If this value exceeds a certain threshold (estimated by means of bootstrapping) a change-point is declared. The work in [8] is somewhat more general, as it considers the workhorse Stochastic Block Model (SBM). The distribution of two so-termed scan statistics is derived for certain particular cases and used to signal a change on the graph sequence.

Going beyond the SBM model, the recent work [9] 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 the rest of the pairs. At every timestep, two statistics are computed for a logarithmic grid of previous instants to check whether they exceed a certain threshold. Evaluating these statistics necessitates computing the eigen-decomposition of an $n \times n$ matrix (where n is the number of nodes in the graph). In addition to being computationally intensive, the algorithm in [9] needs to store all historical data in memory, which may pose a major hurdle even when graphs are of moderate size. In any case, the procedure comes with solid theoretical guarantees in the form of detection delay, average run length and a mini-max lower bound.

Proposed approach and contributions. Here we resort to the Random Dot Product Graph (RDPG) model, a particular but very versatile case of the inhomogeneous Bernoulli graph [10], [11]. In RDPGs each node has an associated latent vector in $\mathbb{R}^d$, and p_{ij} is simply the inner product between the corresponding vectors. As we discuss in Sec. II, RDPGs capture phenomena commonly encountered with real-world graphs

(e.g., statistical dependencies among edges) and subsume the SBM as a special case; they are interpretable while still being amenable to analysis [11].

Building on [12], we will assume a historical dataset with no change-points is available, from which we estimate the latent vectors in an offline training phase. As new data arrive in a streaming fashion, the novel online CPD algorithm (Sec. III) recursively updates a *monitoring function* statistic (whose distribution we characterize analytically). 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 need to 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 eigen-decompositions. Numerical tests in Sec. IV corroborate the effectiveness of the proposed online CPD method, using both simulated and real network datasets.

II. Preliminaries and problem statement

Here we introduce the necessary background on RDPG modeling and inference. 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 $G = (V, E)$, with nodes $V = \{1, \ldots, n\}$ and edges $E \subseteq V \times V$. In the RDPG model each node $i \in V$ has an associated column vector $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$, and edge (i, j) exists with probability $\mathbf{x}_i^\top \mathbf{x}_j$ (a particular case of the latent space model [13]). Note that the set $\mathcal{X}$ of possible $\mathbf{x}_i$ is such that $\mathbf{x}^\top \mathbf{y} \in [0, 1]$, $\forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}$. In general, vectors $\mathbf{x}_i$ may be random, drawn from a distribution in $\mathcal{X}$.

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

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

That is, *given* $\mathbf{X}$, edges are conditionally independent with $A_{ij} \sim \mathrm{Ber}(\mathbf{x}_i^\top \mathbf{x}_j)$. The RDPG model is a tractable yet expressive family of random graphs that subsume Erdös-Rényi (ER) and SBM ensembles as particular cases. Indeed, if $\mathbf{x}_i = \sqrt{p}\ \forall\, i$, 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 [11] for additional examples.

B. Inference on RDPG

Given a graph stemming from an RDPG with adjacency matrix $\mathbf{A}$, we now discuss how to estimate the matrix $\mathbf{X}$ of latent vertex positions. In lieu of a maximum-likelihood estimator that is intractable beyond toy graphs, the key intuition is that $\mathbf{A}$ is a noisy observation of

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

the matrix of edge probabilities p_{ij}, since $\mathbb{E}\left[\mathbf{A} \,\middle|\, \mathbf{X}\right] = \mathbf{P}$. It is thus natural to adopt the estimator

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

The solution to (3) is readily given by

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

where $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$ is the spectral decomposition 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 $\hat{\mathbf{Q}} \in \mathbb{R}^{n \times d}$ are the corresponding d dominant eigenvectors. We are assuming that $\hat{\mathbf{\Lambda}}$ has only non-negative values, a limitation that may be easily circumvented [14]. The constraint $\operatorname{diag}(\mathbf{X}\mathbf{X}^\top) = \mathbf{0}$ can be handled as well [15]. In practice, d is likely unknown but can be estimated by looking for "elbows" on the so-termed eigenvalue scree plot [16]. 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 [11].

The RDPG model is invariant to rotations in $\mathbf{X}$. 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.

Consider now $\mathbf{A}[1] \ldots, \mathbf{A}[m]$ to be a sequence of $n \times n$ independent adjacency matrices, all adhering to an RDPG with latent position matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, i.e., every graph in this sequence satisfies (1) for the same (but unknown) fixed matrix $\mathbf{X}$. We 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 $\bar{\mathbf{A}}$. Since $\bar{\mathbf{A}}$ is also an unbiased estimator of $\mathbf{P}$ and $\operatorname{var}(\bar{\mathbf{A}})_{ij} = \frac{1}{m} p_{ij}(1 - p_{ij})$, then the estimated latent positions will, when $n \to \infty$, follow a normal distribution with variance scaled by $\frac{1}{m}$ relative to the variance of the ASE obtained from a single graph [17].

C. Problem statement

Suppose we are given a batch sequence of m graphs as in the previous paragraph, 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. Given a (possibly infinite) sequence of streaming adjacency matrices $\mathbf{A}[m+1], \mathbf{A}[m+2], \ldots$, we 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 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).

III. Online Change-Point Detection

A. General algorithmic framework

We build on the so-called estimating function approach [12], [18], 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 $\mathbf{A}[t]$, such that $\mathbb{E}[\mathbf{H}] = \mathbf{0}$ under the null hypothesis. If one monitors a cumulative sum of $\mathbf{H}$, that quantity should 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 [12], we first estimate the parameters of the underlying RDPG model using the training data set, i.e., we estimate the 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)]. 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 $\mathbf{G}(\mathbf{A}[t], \hat{\mathbf{X}}) = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}[t]\right)\hat{\mathbf{X}}$ as the estimating function. Accordingly, $\mathbf{G}$ amounts to projecting the residual $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}[t]$ onto $\hat{\mathbf{X}}$.

In order to detect a change on the underlying model, we will track the cumulative sum 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 Sec. IV that adopting the residual itself instead of a projection results 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 compute 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

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

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

If the norm of the partial sum

$$\mathbf{s}[m,k] = \sum_{t=m+1}^{m+k} \mathbf{h}(\mathbf{A}[t], \hat{\mathbf{X}}) \tag{7}$$

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

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.

B. Statistical analysis

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 most applications this is not necessarily true, we will then extend the analysis for the imperfect case.

1) Perfect ASE estimation: In this 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 variables a_{ij} are 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}).

Proposition 1. *Under the perfect ASE estimation assumption, as $k \to \infty$ the test statistic sequence converges in distribution, namely*

$$k^{-1}\Gamma[m,k] \xrightarrow{D} \sum_{l=1}^{r} p_l(1-p_l)y_l^2, \tag{8}$$

where $\{y_l\}_{l=1}^r$ are i.i.d. standard Gaussian random variables.

Proof. Invoking the Central Limit Theorem (CLT), as $k \to \infty$ the distribution of $k^{-1/2}\mathbf{s}[m,k]$ in (7) 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.

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

Hence, $k^{-1}\Gamma[m,k]$ also converges in distribution because $k^{-1}\Gamma[m,k] = (k^{-1/2}\mathbf{s}[m,k])^\top k^{-1/2}\mathbf{s}[m,k]$. Using the convergence of $k^{-1/2}\mathbf{s}[m,k]$ and the fact that $\mathbf{\Sigma}_H$ is a diagonal matrix whose nonzero entries are $p_l(1-p_l)$, the desired result in (8) follows. □

Since y_l is a standard Gaussian r.v. then $y_l^2 \overset{D}{=} \chi_1^2$, and also

$$\mathbb{E}\left[\Gamma[m,k]\right] = k\sum_{l=1}^{r} p_l(1-p_l),$$

$$\text{var}\left[\Gamma[m,k]\right] = 2k^2\sum_{l=1}^{r} p_l^2(1-p_l)^2,$$

where we have used that the $\{y_l\}_{l=1}^r$ are mutually independent.

To control the variance of Γ, the weighting function for the perfect ASE case can be chosen as $\omega[k] = (rk)^{-1}$. The threshold $c_\alpha[k]$ is selected as the $(1-\alpha)$-quantile of the distribution in (8) (after weighting with $\omega[k]$), 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 defined differently.

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

$$\mathbf{h}(\mathbf{A}[t], \hat{\mathbf{X}}) = \text{vec}\left[\text{triu}\left(\mathbf{X}\mathbf{X}^\top - \mathbf{A}[t]\right)\right] + \mathbf{e}, \tag{9}$$

where $\mathbf{e} = \text{vec}\left[\text{triu}\left(\mathbf{E}\right)\right]$. So the first term in (9) 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.

Again, invoking the CLT, the distribution of the cumulative sum $\mathbf{s}$ can be 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 allow us to approximate the distribution of $\Gamma[m,k]$ by the distribution of the random variable

$$\bar{\Gamma} = k\sum_{l=1}^{r} p_l(1-p_l)\left(y_l + b_l\right)^2, \tag{10}$$

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 $\mathbf{b} = \sqrt{k}\mathbf{\Sigma}_H^{-1/2}\mathbf{e}$.

From (10), for large k we can compute the expected value and variance of Γ as in the perfect case. The difference is that each $(y_l + b_l)^2$ is distributed as a non-central $\chi^2(1, b_l^2)$ distribution with one degree of freedom and parameter $b_l^2 = \dfrac{k}{p_l(1-p_l)}e_l^2$ (as before, $\{e_l\}_{l=1}^r$ denote the entries of vector $\mathbf{e}$). Hence,

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

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

where for notational convenience we defined the auxiliary vector $\boldsymbol{\sigma}$ with entries $\{p_l(1-p_l)\}_{l=1}^r$ and $\mathbf{e}^2$ denotes the entry-wise square of $\mathbf{e}$. The preceding arguments suffice to establish the following result on the convergence of $\Gamma[m,k]$.

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

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

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. The threshold $c_\alpha[k]$ is set as the $(1-\alpha)$-quantile of the generalized chi-squared distribution defined in (10) after weighting. Since its cumulative distribution function has a complex form which would require numerical integration, in practice one could simply use the mean plus three standard deviations to obtain a type-I error of approximately $\alpha = 0.01$.

C. Implementation details

The above procedure requires prior knowledge on the values of $\mathbf{P}$ and $\mathbf{e}$ in order to set the threshold $c_\alpha[k]$. In most applications these values are not known, so it is necessary to estimate them. To that end, we use the training set.

For $\mathbf{P}$ we simply use the estimate $\hat{\mathbf{P}} = \hat{\mathbf{X}}\hat{\mathbf{X}}^T$, i.e. we estimate $\mathbf{P}$ using the ASE of $\bar{\mathbf{A}}$ in (5), computed over the training set. To estimate $\mathbf{E}$ (and subsequently $\mathbf{e}$) we perform "leave-one-out" passes over the training set: we randomly select an index j in $1,\ldots,m$ and compute the ASE of $\mathbf{A}[j]$ and of

$$\bar{\mathbf{A}}_{(-j)} = \frac{1}{m-1}\sum_{\substack{t=1\\ t\neq j}}^{m}\mathbf{A}[t],$$

the mean adjacency matrix over the left-out samples. Because $\text{var}\left[\overline{\mathbf{X}}_j\overline{\mathbf{X}}_j^\top - \mathbf{P}\right] = \text{var}\left[\hat{\mathbf{X}}_j\hat{\mathbf{X}}_j^\top - \mathbf{P}\right]/(m-1)$, with $\overline{\mathbf{X}}_j$ and $\hat{\mathbf{X}}_j$ the ASE of $\bar{\mathbf{A}}_{(-j)}$ and $\mathbf{A}[j]$, respectively [17], 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.

IV. Numerical tests

A. Simulated data

A usual problem in networks is to detect when communities arise. So, we first test the proposed online CPD method by generating a sequence of ER graphs with $n = 100$ nodes and connection probability $p = 0.3$. After $t^* = 120$, the model shifts to a two-block SBM with 50 nodes in each community and connection probability 0.275 for nodes in the same community and 0.325 for nodes in different blocks. We use the first $m = 100$ graphs as the training set, and the value of d is automatically chosen (via scree plot) by

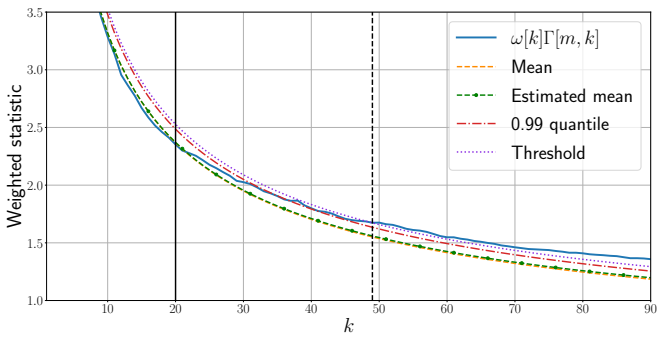

Fig. 1. Evolution of $\omega[k]\Gamma[m,k]$, its mean and the estimated mean, for simulated data. The solid vertical line indicates the actual change-point, while the dashed one is the detection. Two thresholds are shown: the 0.99-quantile of the distribution in (10) and three standard deviations away from the mean.

the `graspologic` library used to obtain the ASE. Note that since the index k in $\Gamma[m,k]$ measures how much time has elapsed since monitoring started, the change-point is at $k^* = 20$.

Figure 1 shows the results for that scenario. We show two thresholds: the 0.99 quantile of the estimated distribution (i.e., the distribution given by (10) but with $\hat{\mathbf{e}}$ instead of $\mathbf{e}$) and the estimated mean plus three standard deviations. As we can see, the difference between those two thresholds is small, so the latter is preferred due to its reduced complexity. Using that threshold a change-point is declared at $k = 48$ (or $t = 148$), so our algorithm is identifying the change in the model. The detection delay can be explained if we look at the estimated mean: since we are estimating the error $\mathbf{E}$ as the 0.99-quantile over the training set, we obtain an estimate that is always greater than the true value. Also, there is an inertia effect associated with monitoring the cumulative sum (7), so the drift in $\Gamma[m,k]$ will not be noticed immediately; see also the discussion in Sec. V.

Comparison with [7]. Had we adopted the approach proposed in [7] (monitoring the distance induced by the Frobenius norm of the adjacency matrices), we would have missed the change altogether. Indeed, if $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n\times n}$ are adjacency matrices of two ER graphs with connection probability p, then

$$\mathbb{E}\left[\|\mathbf{A}-\mathbf{B}\|_F^2\right] = (n^2-n)2p(1-p),$$

since all entries A_{ij} and B_{ij} are Ber(p) r.v.s. Thus $\|\mathbf{A}-\mathbf{B}\|_F^2 \approx 2p(1-p)n^2$ if n is sufficiently large. Suppose now that $\mathbf{C}$ and $\mathbf{D}$ are two adjacency matrices from a two-block SBM, where each community has $n/2$ nodes and the connection probabilities are q_1 for nodes in the same cluster and q_2 for nodes in different communities. Then the connection probability matrix for $\mathbf{C}$ and $\mathbf{D}$ is

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

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 $\mathbf{J}_m$ denoting the $m \times m$ all-ones matrix and $\mathbf{I}_m$ the identity matrix of size m. So those matrices have $n^2/2$ entries of whose expected value is q_2 and $(n/2-1)n \approx n^2/2$ entries whose expected value is q_1. All in all, similarly to the ER case we have

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

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, when $q_1+q_2 = 2p$ we have that

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

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. As an example, in the previous setup the algorithm proposed in [7] (using the author's implementation in the `R` package `gStream`) found no change-points in the data.

On the choice of the monitoring function. This same example allows us to illustrate why choosing the estimating function $\mathbf{G}$ as monitoring function $\mathbf{H}' = \left(\hat{\mathbf{X}}\hat{\mathbf{X}}^\top - \mathbf{A}[t]\right)\hat{\mathbf{X}}$ is not a good idea. 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 $\hat{\mathbf{X}}\hat{\mathbf{X}}^\top = p\mathbf{J}_n$. If there is a change in the nominal model and the observed matrices are subsequently as $\mathbf{C}$ above (i.e., we shift from an ER to a two-block SBM), then $\mathbb{E}[\mathbf{H}'] = (p\mathbf{J}_n - \mathbf{P}_{\text{SBM}})\sqrt{p}\mathbf{J}_{n\times d}$. 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

$$\left(\mathbb{E}[\mathbf{H}']\right)_{ij} \approx n\sqrt{p}\left(p - \frac{q_1+q_2}{2}\right),$$

for large n. Accordingly, choosing p, q_1 and q_2 as before we find that $\mathbb{E}[\mathbf{H}'] = \mathbf{0}$, i.e. we do not expect to see a change in the monitoring function after the change.

B. Real data

The Enron email dataset [19] consists of emails exchanged between $n = 151$ company employees from 1998 to 2002. Using a weekly grouping of these mails (where an edge exists between two employees if they have exchanged at least one mail in the corresponding period), we applied our online CPD method to the one year period between December 1, 1999, and December 1, 2000. As a baseline for comparison and to ensure we had a training period with no changes, we first ran the offline CPD algorithm in [20]. We found that in that period of time there is only one change-point, in the week of July 10th. This is consistent with the known timeline of events for the company, since on July 19, 2000, a partnership between Blockbuster and Enron to provide movie-on-demand services was announced [21]. Since this dataset comprises internal emails only, it is natural for the email network to change before the partnership was publicly announced.

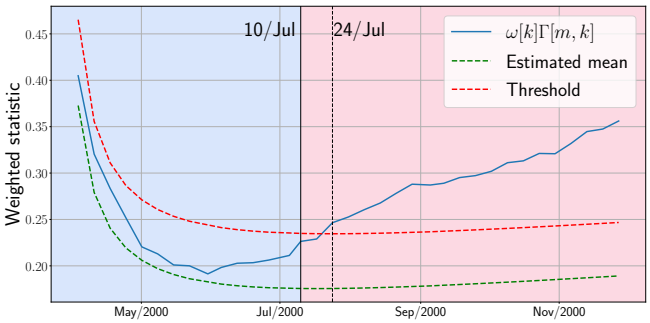

Fig. 2. Online CPD for the Enron dataset. A change in background color indicates the actual change-point, as detected by the offline algorithm [20]. The dashed vertical line shows the detected change-point for the online algorithm.

For the online detection, we used the first 17 weeks as the training set, which correspond to the period between December 1, 1999, and May 31, 2000. The threshold was set at three standard deviations away from the estimated mean. Results are shown in Figure 2. After monitoring starts on April 1st, a change-point is detected on July 24, in line with what was obtained with the offline algorithm.

V. Discussion and Future Work

We developed an 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, the novel algorithm computes (offline) the ASE of the first graphs (i.e., a training set) and then efficiently updates the cumulative sum of a monitoring function $\mathbf{h}$ as data arrive sequentially-in-time. Statistical analysis of the monitored random sequence facilitates deriving meaningful detection thresholds to control type-I error rates.

Under imperfect ASE estimation, we show the distribution of the statistic Γ (and its weighted version $\omega\Gamma$) contains a term depending on the model estimation error. Although in this work we presented a practical and 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. This is another exciting and challenging avenue for future work.

Since the monitoring function is accumulated over time in an infinite-memory fashion, the "reaction time" or "inertia" (i.e., the detection delay) is likely to increase as time goes by. Possible workarounds are to adaptively select the time interval over which the accumulation takes place [18], use fixed-length sliding windows, or, to impose an exponentially-decaying weighting policy (as in adaptive filters for tracking).

Finally, it is important to highlight that the scope of our statistical analysis is limited to undirected and unweighted graphs. A gamut of network science applications call for the non-trivial extensions to the directed and weighted cases [22].

References

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

[2] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Process.*, vol. 167, pp. 107299, 2020.

[3] X. He, Y. Xie, S.-M. Wu, and F.-C. Lin, "Sequential graph scanning statistic for change-point detection," in *52nd Asilomar Conference on Signals, Systems, and Computers*, 2018, pp. 1317–1321.

[4] 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.

[5] L. Peel and A. Clauset, "Detecting change points in the large-scale structure of evolving networks," in *AAAI Conference on Artificial Intelligence*, 2015, vol. 29, pp. 2914–2920.

[6] C. Kaushik, T. M. Roddenberry, and S. Segarra, "Network topology change-point detection from graph signals with prior spectral signatures," *arXiv:2010.11345 [stat.ML]*, 2020.

[7] H. Chen, "Sequential change-point detection based on nearest neighbors," *Ann. Stat*, vol. 47, no. 3, pp. 1381 – 1407, 2019.

[8] 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, 2014.

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

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

[11] A. Athreya, D. E. Fishkind, M. Tang, C. E. Priebe, Y. Park, J. T. Vogelstein, K. Levin, V. Lyzinski, and Y. Qin, "Statistical inference on random dot product graphs: A survey," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 8393–8484, Jan. 2017.

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

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

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

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

[16] 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.

[17] R. Tang, M. Ketcha, A. Badea, E. D. Calabrese, D. S. Margulies, J. T. Vogelstein, C. E. Priebe, and D. L. Sussman, "Connectome smoothing via low-rank approximations," *IEEE Trans. Med. Imaging*, vol. 38, no. 6, pp. 1446–1456, 2018.

[18] 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.

[19] C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park, "Scan statistics on Enron graphs," *Computational & Mathematical Organization Theory*, vol. 11, no. 3, pp. 229–247, 2005.

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

[21] "Enron, Blockbuster Partner For Movie Mania," https://www.forbes.com/2000/07/20/mu4.html?sh=11e6b41a3541, Accessed: 2021-04-29.

[22] A. G. Marques, S. Segarra, and G. Mateos, "Signal processing on directed graphs," *IEEE Signal Process Mag.*, vol. 37, no. 6, pp. 99–116, 2020.