Monitoring network changes in social media

Cathy Yi-Hsuan Chen [*] Yarema Okhrin [†] Tengyao Wang [‡]

December 7, 2020

Abstract

Econometricians are increasingly working with high-dimensional networks and their dynamics. Econometricians, however, are often confronted with unforeseen changes of network dynamics. In this paper, we develop a method and the corresponding algorithm for monitoring changes in dynamic networks. We characterize two types of changes, edge-initiated and degree-initiated, to feature the complexity of networks. The proposed approach accounts for three potential challenges in the analysis of networks. First, networks are high-dimensional objects causing the standard statistical tools to suffer from the curse of dimensionality. Second, any potential changes in social networks are likely driven by a few nodes or edges in the network. Third, in many dynamic network applications such as monitoring network connectedness or its centrality, it will be more practically applicable to detect the change in an *online* fashion than the offline version. The proposed detection method at each time point projects the entire network onto a low-dimensional vector by taking the sparsity into account, then sequentially detects the change by comparing consecutive estimates of the optimal projection direction. As long as the change is sizeable and persistent, the projected vectors will converge to the optimal one, leading to a plunge in sine angle distance between them. A change is therefore declared. We evaluate the performance of our method in terms of the length of the delay and the false alarm rate under various sparsity and signal-to-noise ratios. The application to the social media messages network supports the usefulness of the algorithm for real data.

Keywords: Change point, network, CUSUM, sparsity, social media

[*]Adam Smith Business School, University of Glasgow, UK; IRTG 1792 High Dimensional Non Stationary Time Series, Humboldt-Universität zu Berlin (*e-mail: CathyYi-Hsuan.Chen@glasgow.ac.uk*)

[†]Department of Statistics, University of Augsburg, Germany (*e-mail: yarema.okhrin@wiwi.uni-augsburg.de*)

[‡]Department of Statistical Science, University College London, UK (*e-mail: tengyao.wang@ucl.ac.uk*)

1 Introduction

Modeling network and its dynamics has been an active area of modern statistics and econometrics research, partly due to the increasing popularity of network data generated e.g. from social media and communication networks. Literature in this area is abundant (Jochmans, 2018; Zhu et al., 2017; Han et al., 2019), with some involving the modelling and analysis of very large-scale networks (Chen et al., 2019). However, so far, relatively little emphasis has been placed on detecting changes in dynamic networks. An overlook of a potential change of networks may come at risk; the network has different structures if there is a change and the models based on pre-change network data are no longer applicable in prediction or risk management tasks. More importantly, any change of the network structure may signify a change of connectivity between nodes or a replacement of key players, also known as central nodes, within the network.

The problem of detecting changes in a dynamic network falls into a broad area of *change point analysis*, where the goal is to identify a change in the data generating mechanism in a data stream. Monitoring a univariate time series has been well studied under the banner of statistical process control (e.g. Duncan, 1952; Page, 1954; Barnard, 1959). Motivated by applications, there has been an increased interest in developing change point detection and estimation procedures in multivariate or even high-dimensional data sets. Examples include detection methods for changes in the mean structure (Zhang et al., 2010; Horváth and Hušková, 2012; Enikeeva and Harchaoui, 2019; Cho and Fryzlewicz, 2015; Jirak, 2015; Zou et al., 2015; Cho, 2016; Wang and Samworth, 2018; Chen et al., 2020) and for changes in the covariance structure Lavielle and Teyssiere (2006); Aue et al. (2009); Bücher et al. (2014); Preußet al. (2015); Wang et al. (2019); Dette et al. (2020) of the data. Detecting or monitoring a change in the network is even more methodologically challenging than the conventional detecting practice applied to the univariate or multivariate process. The connectivity structure of the network exhibits a time-varying feature. Network modeling strives to characterize the interactions within the network and their dynamics. The curse of dimensionality, which also happens to general change points detection tasks, exacerbates in a network given network's complexity and the featured interactions between nodes. Recently, Wang et al. (2020) studied the problem of change point detection in a dynamic

network with binary edge weights. Their method and analysis have focused on the so-called *offline* problem, requiring full knowledge of the entire history of the network dynamics before retrospectively identifying the change point locations. However, in many dynamic network applications such as monitoring network connectedness or its centrality, it will be more practically relevant to detect a change in an *online* fashion than the offline version as data are sequentially observed and most importantly, any change upon the new arrival observation needs to be scrutinized.

This paper presents a novel method for detecting changes in a dynamic network sequentially over time. The proposed method does not rely on direct estimation of the network at each time point but rather tracks the evolvement of networks over time. In other words, regardless of the complexity of the original network, the procedure works as long as the change is sufficiently structured. Depending on the nature of the network (e.g. directed or undirected) and the nature of the change (e.g. whether all edges incident on the same node change together), we propose a *degree-based* and also an *edge-based* scheme to tackle the variety of changes. Motivated by Wang and Samworth (2018), our key idea is to represent the dynamic network as a high-dimensional time series and estimate the optimal projection directions that can best aggregate the data into a one-dimensional series with the largest signal-to-noise ratio when a change has occurred. Given the sequential arrival of observation, we construct a sequence of the estimators of the projection directions. We declare a change when a sufficiently large number of consecutive projection direction estimators are well-aligned with each other.

More specifically, in line with Wang et al. (2020), a sparsity assumption is the underpinning of the framework, in the sense that the changes are driven by a small subset of nodes or edges, which translates to a sparse change in the data vectors. The imposed sparsity allows us to consistently estimate an optimal projection direction from a high-dimensional network vector space. This machinery is iterated for every new arrival observation at t and delivers a sequence of projection vectors. The sine angle distance is used to measure the alignment between consecutive estimates and a change is declared when a sufficient number of such sine angle distances all fall below the specified threshold. The proposed algorithm **NSM** (Network Sequential Monitoring) delineates this procedure for a detection

convenience. The theoretical justification and support for the invented method and algorithm are provided, including the choice of threshold and tail sequence in a consideration of false alarm rate.

In the simulation section, we extensively simulate the network dynamics under different natures of changes and network structures. We evaluate the performance of the proposed methods and confirm its effectiveness in terms of false alarm rate and the length of the delay. We end up with the following tactical guidance for detection strategies. As long as the magnitude of the change is sizable, both degree-based and edge-based methods are comparable in terms of the average detection delay. On the other hand, for the changes that are edge-sparse (i.e. only a small number of edges change) or heterogeneous (i.e. changes in different edges can have different signs), a degree-based detection scheme suffers a longer detection delay, which is potentially caused by an offsetting effect between positive and negative shifts. In such a scenario, the edge-based detection method is recommended.

To further illustrate the utility of our procedure, we apply the proposed method and monitoring algorithm to the social media networks. The social media networks can be very dynamic and sparse, reflecting a changing interaction and few opinion leaders in a social network. The rise of social media platforms is in favour of exploitation towards social network data. With the ubiquitous use of social media in modern society, an unprecedented amount of data become available and of interest to many fields of study. Applying the state-of-art methods to social media data benefits a good understanding of social behaviour and human interaction, and the dynamics of interaction. The social media data are featured with three characteristics that impose challenges to econometricians: the data are immense, noisy, and dynamic. We can translate these challenges into the statistical context- the curse of dimensionality, dynamic networks and sparsity.

Concerning an intriguing 'cryptocurrency bubble' during the end of 2017 and the beginning of 2018 (Hafner, 2020; Cavaliere et al., 2020; Chen and Hafner, 2019), we limit our attention to the network comprising of cryptocurrency-specific messages. The mainstream topics related to such bubble are framing a bubble test and exploiting the proposed test on this unregulated market. We take a different stance from the previous studies as we are interested in a change of network dynamics during the lifetime of the bubble. We, therefore,

contribute to a bubble examination through the lens of a change in the network dynamics. Lux (1995) studies herd behaviour among speculative traders, bubbles and crashes, and indicates a vital driving force of bubble stemmed from speculators' herd behaviour. In terms of the causes of a bubble, social media data is vital because the expressed opinions shared among users, measured by their contents' similarity, is one of the indications of herd behaviour. One, therefore, can measure messages' similarity at each time point and monitor the change of similarity in order to elicit the lifetime of a bubble, from its growth or its collapse.

The paper is organized as follows. Section 2 describes a network, an algorithm for change detection, applies the algorithm to the models and discusses the theoretical properties. Section 3 conducts simulation exercises for node-initiated and edge-initiated change, respectively, and robustly checks when the temporal and spatial dependence is present. We present and discuss the application of our model to the social media network in Section 4. Section 5 concludes. We dedicate Section 6 to the proofs. The developed codes for this study can be found at github.com/wangtengyao/NSM.

2 Online change detection via sequential estimation

2.1 Description of a network

Data on networks becomes increasingly popular in modern statistical applications. A convenient way to a formal definition and visualization of networks offers the graph theory. A graph G is defined to be a pair (V, E), where $\mathsf{V} = (\nu_1, \cdots, \nu_p)$ is a finite set of *vertices* or *nodes*, and $\mathsf{E} = (e_{ij})_{i,j\in\{1,\ldots,p\}}$ is a $p \times p$ *matrix of edge weights*. Within the context of networks the matrix E is frequently referred to as *adjacency matrix*. In this paper we deal with time-varying networks characterized by *edge-dynamic* graphs, i.e. the graph has a fixed vertex set and its adjacency matrix E_t varies with time $t \in \mathbb{N}$. We write $\mathsf{G}_t = (\mathsf{V}, \mathsf{E}_t)$ with $\mathsf{E}_t = (e_{ij,t})$ for $i, j \in \{1, \ldots, p\}$ and $t \in \mathbb{N}$. If the adjacency matrix is symmetric, i.e. $e_{ij,t} = e_{ji,t}$ for all i, j, we refer to the underlying network as a *undirected network*. Alternatively it is featured as a *directed* one. Moreover, let $e_{ii,t} = 0$ for all i implying no loops on the vertices.

The network edges are assumed to be random variables reflecting the stochastic nature of real world networks. Let $e_{ij,t} \sim F$ and $E(e_{ij,t}) = \mu_{ij,t}$ implying a potentially non-constant expected edge weights. We postulate temporal and cross-sectional independence of the weights. Although these assumptions might be violated for real data, we show below that the consequences for online change point detection are negligible. The dynamics of the model is assumed to be affected by a location shift at time point τ, leading to a change-point model fromalized as

$$\mu_{ij,t} = \begin{cases} \mu_{ij}^{(0)} & \text{for } t \leq \tau \\ \mu_{ij}^{(1)} & \text{for } t \geq \tau + 1 \end{cases}.$$

The objective is to detect this shift in the network as soon as possible after its occurrence. For monitoring purposes, we vectorise the adjacency matrices and consider the edge-based monitoring with

$$X_t = \begin{cases} \mathsf{vech}(\mathsf{E}_t) & \text{for undirected networks} \\ (\mathsf{vech}(\mathsf{E}_t), \mathsf{vech}(\mathsf{E}_t')) & \text{for directed networks} \end{cases},$$

where $\mathsf{vech}(\cdot)$ denotes the half-vectorization operator for the lower triangular part of a matrix excluding the main diagonal.

Note that monitoring large networks will suffer from the curse of dimensionality and make most algorithms computationally infeasible. In the economic context, the researchers are inclined to draw a conclusion to the extent which nodes are responsible for the change. The supervisory authorities may particularly keep eyeballing on these detected responsible nodes for a supervisory convenience and effectiveness. To overcome computational problem and benefit the supervisory practice, we consider in the case of undirected graphs the degree of the ith vertex of a graph at t defined as $\mathsf{deg}(\nu_{i,t}) = \sum_{j=1}^{p} e_{ij,t}$, i.e. the average of edge weights, $e_{ij,t} \in \mathbb{R}$, that are incident to that vertex i. In the case of directed graphs we distinguish between ingoing and outgoing degrees and define $\mathsf{deg}^{(in)}(\nu_{i,t}) = \sum_{j \neq i} e_{ji,t}$ and $\mathsf{deg}^{(out)}(\nu_{i,t}) = \sum_{j \neq i} e_{ij,t}$ respectively. Thus, the ingoing degree is computed as the average of incoming connections, whereas the outgoing degree is the average of connections starting at the vertex i. The use of degrees instead of the edges can be seen as a dimension reduction technique signifying the connectedness of nodes in the network. The node-based monitoring

controls the vector of degrees with

$$X_t = \begin{cases} (\mathsf{deg}(\nu_{1,t}), \ldots, \mathsf{deg}(\nu_{p,t}))' & \text{for undirected networks} \\ (\mathsf{deg}^{(in)}(\nu_{1,t}), \ldots, \mathsf{deg}^{(out)}(\nu_{p,t}))' & \text{for directed networks} \end{cases}.$$

that quantifies the connectedness of nodes in G_t. We note that if edge weights are independently generated with a light-tail distribution, the vector of degrees only has a weak correlation across coordinates.

2.2 Algorithm for change detection

Using either the edge-based or the degree-based monitoring as described in the previous subsection, we have summarised the dynamic network at each time point as a data vector. This leads us to consider the following simplified mathematical model for change detection, which motivates our Algorithm 1. Let $X_1, X_2, \cdots$ be independent and sequentially observed with $X_t \sim N_p(\mu_t, I_p)$, where $\mu_t = \mu^{(1)}$ for $t \leq \tau$ and $\mu_t = \mu^{(2)}$ for $t \geq \tau + 1$. For sparse changes where $\|\mu^{(1)} - \mu^{(2)}\|_0 \ll p$ (we write $\|\mathbf{x}\|_0 := \sum_i \mathbf{1}(x_i \neq 0)$ for the number of nonzero entries in a vector $\mathbf{x}$), Wang and Samworth (2018) proposed a projection-based approach to retrospectively identify the location of the change point after observing the entire history of data series. A key component of their approach is to estimate the optimal direction of projection, which is parallel to the sparse vector of change. Building upon this idea, we propose in Algorithm 1 to sequentially detect the change by comparing consecutive estimates of the optimal projection direction. Heuristically, after a sufficient number of post-change data points are observed, the estimated projection directions denoted as $\hat{v}^{(t)}$ at different time point t will all be close to the optimal one, and hence are close to each other. On the other hand, prior to the change, there is no signal in the data to favour any particular direction, and thus consecutive estimates are unlikely to align with each other.

In Step 5 of Algorithm 1, the soft-thresholding operator is defined as $\mathbf{soft}(T_{j,t}^{(n)}, \lambda_n) := \mathrm{sgn}(T_{j,t}^{(n)}) \max\{|T_{j,t}^{(n)}| - \lambda_n, 0\}$. This step constructs a projection direction estimator equivalent to the result of the `sparse.svd` function in the `inspect` package (Wang and Samworth, 2016) using argument `schatten=2`. This choice simplifies both the presentation and implementation of the algorithm since the estimator has a close form solution, and in our

experience, it does not significantly impact the empirical performance of the procedure. However, if desired, Step 5 of Algorithm 1 can be replaced with calling `sparse.svd` using argument `schatten=1`. The latter corresponds to a slightly more accurate projection direction estimator based on a nuclear-norm-based (instead of Frobenius norm) convex relaxation, which comes at the expense of a much higher computational cost. We also remark that if $\|T^{(t)}\|_\infty := \max_{j\in[p],s\in[t-1]} |T^{(t)}_{j,s}|$ falls below the soft-threshold level λ_t, Step 7 of Algorithm 1 will generate a projection direction estimator $\hat{v}^{(t)}$ uniformly at random on the unit sphere to reflect our lack of knowledge of the change direction when all CUSUM statistics are too small in absolute values.

After obtaining projection direction estimators at different time points, Algorithm 1 will compute the sine angle distance between successive estimates $\hat{v}^{(t)}$ and declare a change when a sufficient number (provided by the input sequence $(b_t)_{t\in\mathbb{N}}$) of such sine angle distances all fall below the threshold $1/2$. Here, the exact choice of $1/2$ is not an issue. In fact, any number in $(0, 1)$ will have the same effect theoretically. We have chosen the threshold at $1/2$ for convenience and since it performs well in finite sample numerical experiments.

2.3 Theoretical guarantee

We now give theoretical justification of Algorithm 1. We assume in our theoretical analysis that $X_1, X_2, \ldots$ are mutually independent

$$X_t \sim \begin{cases} N_p(\mu^{(1)}, I_p), & \text{if } t \le \tau, \\ N_p(\mu^{(2)}, I_p), & \text{if } t \ge \tau + 1. \end{cases}$$

Moreover, we write $s := \|\mu^{(1)} - \mu^{(2)}\|_0$ for the sparsity of the change and $\rho := \|\mu^{(1)} - \mu^{(2)}\|_2$ for the Euclidean norm of change.

Our first result below shows that with a suitable sequence of soft-thresholding parameters $(\lambda_t)_{t\in\mathbb{N}}$, Algorithm 1 can achieve theoretical control of false alarm rate.

Theorem 1. *Fix some $\alpha \in (0, 1/2]$. Let N be the output of Algorithm 1 with input $X_1, X_2, \ldots \overset{\text{iid}}{\sim} N(\mu, I_p)$, $\lambda_t := 2\sqrt{\log(t^2p/\alpha)}$ and tail sequence $b_t := \lceil \frac{2\log t + \log(2/\alpha)}{p/8} \rceil$. We have*

$$\mathbb{P}(N < \infty) \le \alpha.$$

Algorithm 1: **N**etwork **S**equential **M**onitoring (**NSM**) sparse mean change

Input: $X_1, X_2, \ldots \in \mathbb{R}^p$ observed sequentially, threshold sequence $(\lambda_t)_t$, tail length sequence $(b_t)_{t\in\mathbb{N}}$.

Set $t \leftarrow 0$ and draw $\hat{v}^{(0)}$ uniformly from the unit sphere $\mathbb{S}^{p-1}$ (w.r.t. the Haar measure).

repeat

  $t \leftarrow t + 1$

  Compute the CUSUM matrix $T^{(t)} = (T^{(t)}_{j,s})_{j\in[p],s\in[t-1]}$ as

$$T^{(t)}_{j,s} \leftarrow \sqrt{\frac{s(t-s)}{t}}\left(\frac{1}{s}\sum_{r=1}^{s} X_{r,j} - \frac{1}{t-s}\sum_{r=s+1}^{t} X_{r,j}\right).$$

  if $t \geq 2$ *and* $\|T^{(t)}\|_\infty > \lambda_n$ **then**

    Define $\hat{v}^{(t)}$ to be the leading left singular vector of $\mathbf{soft}(T^{(n)}, \lambda_n)$, where **soft** is the entrywise soft-thresholding operator.

  else

    Sample $\hat{v}^{(t)}$ uniformly on the unit sphere $\mathbb{S}^{p-1}$.

  Compute $A^{(t)} \leftarrow \sin\angle(\hat{v}^{(t)}, \hat{v}^{(t-1)})$.

until $t \geq b_t$ *and* $\max_{t-b_t+1\leq i\leq t} A^{(i)} < 1/2$;

Output: $N = t$

Here, the choice of the specific form of b_t, which increases logarithmically with respect to t, is to ensure that when no change point is present, the procedure will not produce a false alarm, even for an arbitrarily long sequence of observations. We remark that for most practical applications of moderately large dimension and not too long a monitoring time span, the tail sequence b_t defined in Theorem 1 is often very small. For example, for $p \geq 80$ and $\alpha \geq 0.01$, we have $b_t \leq 2$ for all $t \leq 1500$ and $b_t \leq 3$ for all $t \leq 200,000$, which means that it suffices compare the most recent three or four estimates of projection direction $\hat{v}^{(t)}$ to detect a change. However, practitioners may opt to use a slightly larger value of b_t to be sure that changes have indeed occurred. Of course, this comes at the price of a slightly longer detection delay.

Theorem 1 guarantees that regardless of the location of change, the procedure in Algorithm 1 is unlikely to declare a change before it happens. The following theorem complements it by showing that our procedure will declare a change not long after the actual change point.

Theorem 2. *Fix $\alpha \in (0, 1/2]$. Let $z \in \mathbb{N} \cup \{\infty\}$ and let N be the output of Algorithm 1 with input $X_1, X_2, \ldots X_\tau \overset{\text{iid}}{\sim} N(\mu^{(1)}, I_p)$, $X_{\tau+1}, X_{\tau+2}, \ldots \overset{\text{iid}}{\sim} N(\mu^{(2)}, I_p)$, $\lambda_t := 2\sqrt{\log(t^2 p/\alpha)}$ and tail sequence $b_t := \lceil \frac{2\log t + \log(2/\alpha)}{p/8} \rceil$. Assume condition 1*

$$\tau \geq \frac{2^{15/2}\lambda_{2\tau}\sqrt{s\tau}}{\rho} + b_{2\tau}, \tag{1}$$

then we have

$$\mathbb{P}\left(N - \tau \leq \frac{2^{15/2}\lambda_{2\tau}\sqrt{s\tau}}{\rho} + b_{2\tau} \right) \geq 1 - \alpha.$$

From the discussion after Theorem 1, we see that we may regard $b_{2\tau}$ essentially as a constant. Hence, condition 1 amounts to $\tau \gtrsim s\rho^{-2}\log(\tau^2\rho/\alpha)$. On the other hand, the response delay is shown to be inversely proportional to the signal strength ρ. We justify the theoretical choice of threshold in the simulation section.

3 Simulation exercise

In this section, we study the numerical performance of our proposed procedure in simulated settings. In all our numerical examples below, we work with the following data generating

mechanism. Let $\mathsf{G}_t = (\mathsf{V}, \mathsf{E}_t)$ be an edge-dynamic graph with a deterministic vertex set $\mathsf{V} = \{1, \ldots, p\}$ and independent edge weight matrices E_t over time t. For simplicity, we assume all diagonal entries of E_t are equal to 0. We consider both directed and undirected networks. In the former case, $\mathsf{E}_t = (e_{ij,t})_{i,j\in\{1,\ldots,p\}}$ has independent entries and in the latter case, E_t has independent upper-triangular entries. Let $\boldsymbol{\mu}_t = (\mu_{ij,t}) \in [0,1]^{p\times p}$ be some deterministic matrix, we assume that entries of E_t are marginally distributed as

$$e_{ij,t} \sim \text{Beta}\big(10\mu_{ij,t}, 10(1-\mu_{ij,t})\big).$$

Specifically, we have $\mathbb{E}(\mathsf{E}_t) = \boldsymbol{\mu}_t$. In our change point setting, we assume $\boldsymbol{\mu}_t = \boldsymbol{\mu}^{(0)}$ for all $t < \tau$ and $\boldsymbol{\mu}_t = \boldsymbol{\mu}^{(1)}$ for all $t \geq \tau$, where $\tau \in \mathbb{N}$ is the change point of interest.

We will mainly focus on vertex-initiated changes in our simulations. Specifically, let $\mathsf{V}_0 \subseteq \mathsf{V}$ be a subset of vertices of cardinality k, we assume that

$$\mu_{ij}^{(1)} - \mu_{ij}^{(0)} = 0 \quad \forall i \notin \mathsf{V}_0,\, j \notin \mathsf{V}_0.$$

In other words, changes only occur in the edges incident on a vertex belonging to the set V_0.

We consider the following three different scenarios of how changes are developed.

(S1) Undirected graph with aligned changes: $\mu_{ij}^{(1)} - \mu_{ij}^{(0)} = c$ for all $i, j \in \mathsf{V}_0$ and $i \neq j$.

(S2) Undirected graph with random-signed changes: $\mu_{ij}^{(1)} - \mu_{ij}^{(0)} = \mu_{ji}^{(1)} - \mu_{ji}^{(0)} \sim \text{Unif}\{c, -c\}$ with equal probability independently for all $i, j \in \mathsf{V}_0$ and $i < j$.

(S3) Directed graph with random-signed changes: $\mu_{ij}^{(1)} - \mu_{ij}^{(0)} \sim \text{Unif}\{c, -c\}$ with equal probability independently for all $i, j \in \mathsf{V}_0$ and $i \neq j$.

As described in Section 2.1, we will apply the NSM Algorithm to the simulated datasets generated from three different scenarios (S1), (S2) and (S3) via either an edge-based monitoring scheme (where the data vectors X_t are constructed by vectorising the edge matrix E_t) or a degree-based monitoring scheme (where the data vectors X_t are constructed as the vector of degrees). Before discussing our simulation results in detail, we first describe here our tuning parameter choices. Algorithm 1 depends on two sets of tuning parameters $(\lambda_t)_{t\in\mathbb{N}}$ and $(b_t)_{t\in\mathbb{N}}$. As mentioned after Theorem 1, it is typically sufficient in practice

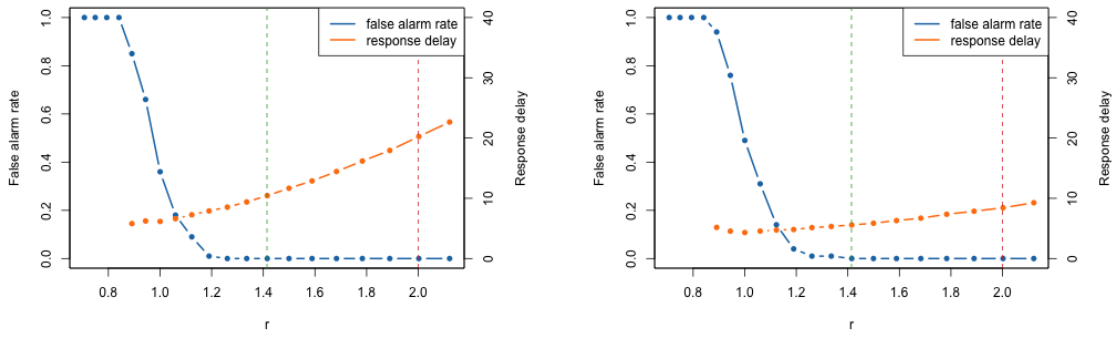


Figure 1: Dependence of false alarm rate and response delay on tuning parameter sequence $(\lambda_t)_{t\in\mathbb{M}}$. Horizontal axis show the scale r for the tuning parameter $\lambda_t = r\sqrt{\log(t^2p/\alpha)}$ and $b_t = 2$. In the left panel, data are generated in scenario (S1) with $N = 30$, $\tau = 200$, $k = 3$, $c = 0.05$ and $\alpha = 0.01$. In the right panel, data are generated by model (S3) with $N = 30$, $\tau = 200$, $k = 30$, $c = 0.25$ and $\alpha = 0.01$. Both false alarm and response delays are averaged over 100 Monte Carlo simulations. The theoretical and empirical suggestions for the tuning parameter λ are indicated with red ($r = 2$) and green ($r = \sqrt{2}$) vertical dashed lines respectively.

to choose the tail length parameter b_t to be a small constant. For simplicity, we will set $b_t = 2$ for all t throughout this section. The choice of the soft-thresholding parameter λ_t is rather subtle. To study the dependence of the algorithmic performance on λ_t, we compared the false alarm rate and average response delays of the procedure over a wide range of settings. Figure 1 shows the simulation results with respect to two performance indicators; results in other models are qualitatively similar. We see that the theoretical choice of $\lambda_t = 2\sqrt{\log(t^2p/\alpha)}$ (corresponding to $r = 2$ in the plot) is rather conservative in terms of the average response delays. Even with a choice of $\lambda_t = \sqrt{2\log(t^2p/\alpha)}$ (corresponding to $r = \sqrt{2}$ in the plot) the false alarm rate is well under control and small response delays can be achieved. Hence, we suggest using $\lambda_t = \sqrt{2\log(t^2p/\alpha)}$ in practice and stick to this choice throughout our simulations.

	FA rate	$c = 0.05$	$c = 0.1$	$c = 0.15$	$c = 0.2$	$c = 0.25$
			S1			
$k = 3$	0	10.4	4.84	3.81	3.17	3.02
$k = 10$	0	9.37	4.98	3.91	3.13	3.00
$k = 30$	0	8.70	5.13	3.98	3.13	3.00
$k = 3$	0	189	34.4	17.1	12.5	9.68
$k = 10$	0	162	28.2	15.6	11.3	10.0
$k = 30$	0	151	26.6	14.4	11.4	10.2
			S2			
$k = 3$	0	11.3	5.17	3.81	3.27	3.03
$k = 10$	0	11.2	5.46	4.05	3.65	3.02
$k = 30$	0	9.86	5.53	4.12	3.75	3.00
$k = 3$	0	187	36.7	17.2	12.2	10.1
$k = 10$	0	179	31.1	15.8	11.4	10.5
$k = 30$	0	154	26.6	14.6	11.8	9.7
			S3			
$k = 3$	0	238	150	35.2	18.5	12.6
$k = 10$	0	191	29.3	13.6	8.86	6.83
$k = 30$	0	99.1	19.3	10.2	7.17	5.56
$k = 3$	0	204	41.2	19.9	12.8	10.3
$k = 10$	0	173	32.3	16.9	12.2	10.1
$k = 30$	0	174	28.3	15.6	12.0	10.1

Table 1: Performance of NSM algorithm under sparsity and signal-to-noise ratio

S1: fixed-sign change in an undirected graph; S2: fixed-sign change in a directed graph; S3: random-sign change in a directed graph. In each model, the upper part is the results of degree-based detection and the lower part is those from edge-based detection. The first column shows the false alarm (FA) rate, followed by the results of detection delay.

3.1 Node-initiated changes

The performance of the proposed network sequential detection method hinges on the size of the signal as well as the sparsity of the network. Theorem 1 and 2 characterise how signal strength and soft-thresholding jointly determine the average response delay and the false alarm rate. To gain better insights, we vary both sparsity and signal strength over a grid of possible values. Specifically, we apply NSM algorithm with a degree-based detection scheme to each of the three scenarios (S1), (S2) and (S3) under $p = 30$, $\tau = 200$, $\alpha = 0.01$, $k \in \{3, 10, 30\}$ and $c \in \{0.05, 0.1, 0.15, 0.2, 0.25\}$. k controls for the sparsity of the network. The parameter c captures the magnitude of the signal and is chosen to range from where the change is just detectable to where the change can be declared almost immediately after it occurs if signals are considerably large. For each setting, we perform 100 Monte Carlo repetitions and estimate the false alarm rate $\mathbb{P}(N \leq \tau)$ via the empirical proportion of declared change points that is smaller than or equal to τ. Similarly, we estimate the response delay $\mathbb{E}(N - \tau \mid N > \tau)$ by the empirical average of $N - \tau$ over all repetitions where no false alarm occurs.

In Table 1, we summarise the false alarm rate and response delay under various sparsity and signal-size settings. The three panels correspond to the three scenarios (S1), (S2) and (S3). For each scenario, in addition to the degree-based detection scheme, we also deploy an edge-based detection scheme as described in Section 2.1 for a comparison purpose. We see that in all simulation settings, no false alarms were observed. In other words, the false alarm rate is well-controlled at the nominal level of $\alpha = 0.01$. The response delay shrinks as either the signal strength c or the sparsity k increases. Note that when we set $b_t = 2$, even if every $\hat{v}^{(}t)$ is estimated to the oracle projection direction, it takes at least three observations after the change point to activate a declaration of a change. Hence, we see that in the setting with the strongest signals, the response delay levels cut off at three. We notice also that the signal size c significantly impacts on the response delay, whereas the sparsity level k is less critical.

There exists a striking difference between the degree-based and edge-based detection scheme in terms of performance. By comparison, the degree-based method performs much better in S1 and S2, which corresponds to the scenarios where all edges weights change

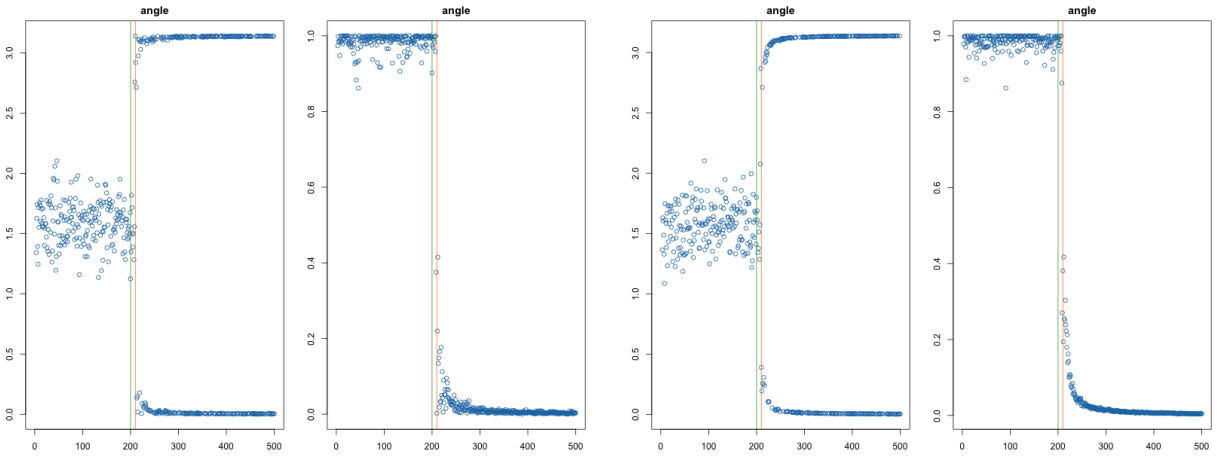
in the same direction in mean. The observed superiority of the degree-based scheme is attributed to the fact that degree-based aggregation exploits the 'grouping structure' of $\boldsymbol{\mu}^{(1)} - \boldsymbol{\mu}^{(0)}$ in a vertex-initiated change model, in the sense that the edges incident on the same vertex are likely to change coincidentally in the same direction. On the other hand, the comparison between vertex-based and edge-based detection schemes yields a mixed result in S3 of Table 1. Since the change is featured with random signs, degree aggregation may stumble caused by an offset effect. While aggregating signs, the positive sign cancels out the negative one, and vice versa. This offset effect exacerbates in the sparse change case when $k = 3$. For denser changes ($k = 10$ or $k = 30$), however, the advantage, e.g. the hidden grouping structure, adhered to the degree-based scheme outweighs the cost imposed by an offset effect, and the degree-based method becomes superior again.

Figure 2 illustrates how NSM Algorithm works in our simulated data. It depicts the evolution over time of the sine angle between two consecutive non-zero projection vectors $\hat{v}^{(t)}, \hat{v}^{(t-1)} \in \mathbb{R}^p$, and $\sin \angle(\hat{v}^{(t)}, \hat{v}^{(t-1)})$. In panels (c) and (d), where shift is sizable ($c = 0.2$), the detection works well in the sense that the value of $\sin \angle(\hat{v}^{(t)}, \hat{v}^{(t-1)})$ drops immediately and sharply at the predetermined change point $\tau = 200$ (green line). In the case of (a), the response delay is visible. We conclude that applying the proposed method is sensible as long as signal is considerably substantial.

Figure 3 conveys the similar messages as Figure 2, except for (a). Consistent with S3 in Table 1, in the case, the network is featured with direction and random-sign, under a weak signal and a sparsity the detection blows up as shown in (a).

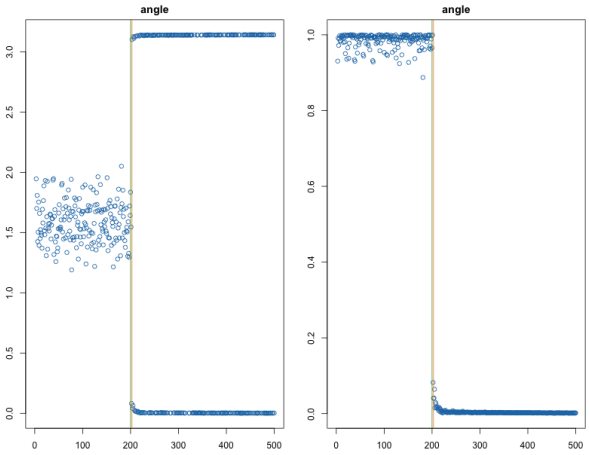
3.2 Edge-initiated changes

The sparsity considered above is restricted to shifts in all edges incident on a subset of vertices of cardinality k. Such shifts induce a well detectable changes in the degrees due to aggregation and lead to a superior performance of the degree-based detection method. This is particularly evident in the case of fixed-sign changes (see Panels S1 and S2 of Table 1). Here we relax the above restriction and allow only a few edges of each vertex to shift. Let Ω_0 be the set of indices $j \in \{1, ..., p\}$, such that there is a shift in $\mathbb{E}(e_{ij,t})$ for $i \in \mathsf{V}_0$ and

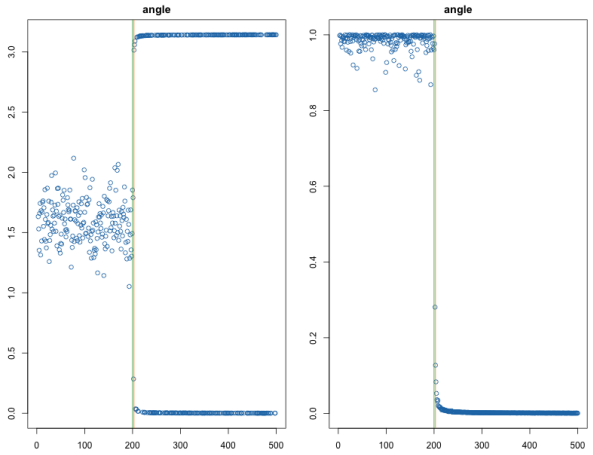


(a) sparse and small change

(b) dense but small change

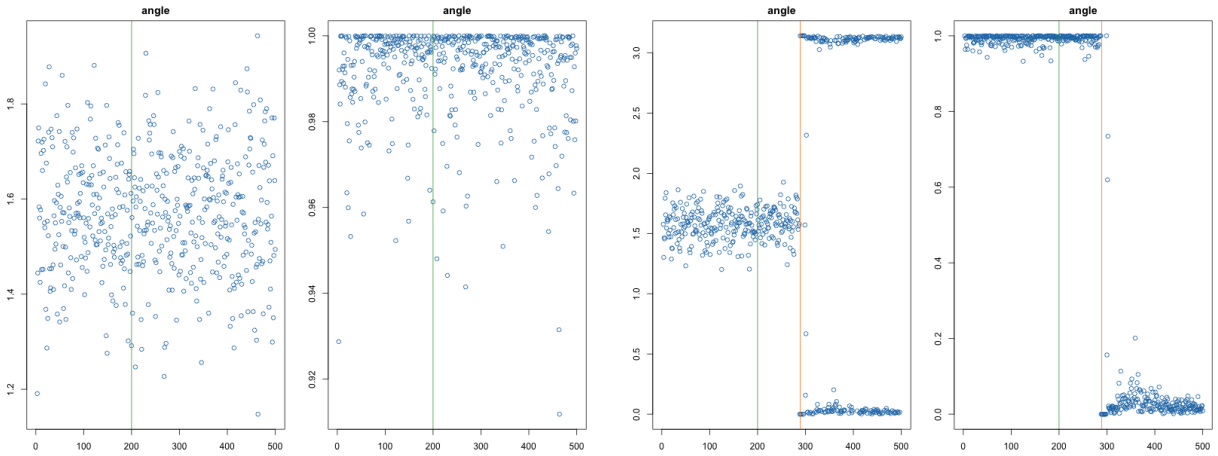


(c) sparse and remarkable change



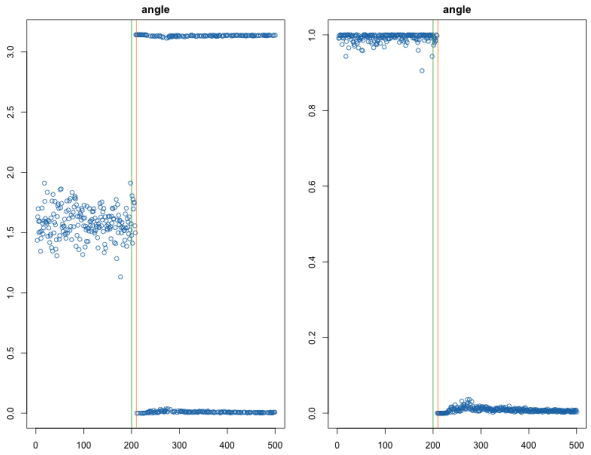
(d) dense and remarkable change

Figure 2: Simulate under scenario (S1) with different sparsity levels and magnitudes of change. Parameter values: (a) $k = 3$, $c = 0.05$; (b) $k = 30$, $c = 0.05$; (c) $k = 3$, $c = 0.25$; (d) $k = 30$, $c = 0.25$.

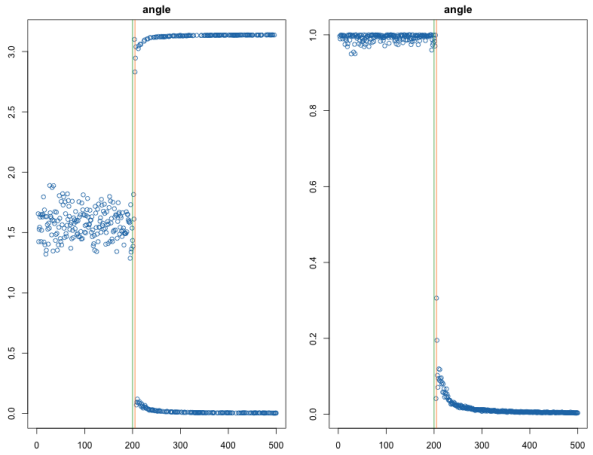


(a) sparse and small change

(b) dense but small change



(c) sparse and remarkable change



(d) dense and remarkable change

Figure 3: Simulate under scenario (S3) with different sparsity levels and magnitudes of change. Parameter values: (a) $k = 3$, $c = 0.05$; (b) $k = 30$, $c = 0.05$; (c) $k = 3$, $c = 0.25$; (d) $k = 30$, $c = 0.25$.

$j \in \Omega_0$ at time τ. To formalize this we specify

$$\mu_{ij}^{(1)} - \mu_{ij}^{(0)} = 0 \quad (\forall i \notin \mathsf{V}_0,\ \forall j \notin \mathsf{V}_0\ ) \vee (\forall i \in \mathsf{V}_0,\ \forall j \notin \Omega_0\ ).$$

We consider here only undirected graphs with random-signed changes. This implies that the shifts are highly sparse and heterogeneous. The node degrees are expected to react more reluctantly to this type of changes due to offsetting of positive and negative shifts and due to a minor impact of only a few edges on the node degree. The resulting model that modifies S2 is summarized as follows:

(S4) Undirected graph with random-signed changes and shifts in the first k_e edges: $\mu_{ij}^{(1)} - \mu_{ij}^{(0)} = \mu_{ji}^{(1)} - \mu_{ji}^{(0)} \sim \text{Unif}\{c, -c\}$ with equal probability independently for all $i \in \mathsf{V}_0$, $j \in \Omega_0$.

Specifically, we assume that for every vertex in V_0 only the first k_e edges shift, i.e. $\Omega_0 = \{1, ..., k_e\}$ and apply NSM Algorithm with $k \in \{2, 3, 5\}$ and $k_e \in \{10, 15, 20, 25\}$. The remaining parameters are set as in the case of node-initiated changes.

The results of a simulation study that are summarized in Figure 4 confirm our expectations. If the shift is small, then the node-detection leads to marginally shorter delays. Larger sizes of the shift reveal the superiority of the edge-detection technique. This becomes particularly evident if the number of affected nodes is small and the aggregated shift in the node degree is difficult to identify. Moreover, the node detection is extremely robust to the variation in the number of affected edges k_e. On the contrary, the variation in the delays for degree detection as a function of k_e is very substantial, mirroring the aggregation effect in the degrees. The dominance of edge detection diminishes, however, if we weaken the sparsity of the changes by increasing the number of vertices with shifts. For example, if most of the edges of the five nodes are subject to a change, then both approaches show almost equivalent performance.

3.3 Robustness analysis

The theoretical justifications for the suggested monitoring technique and the above simulation study assume independent edges and degrees both in time and in cross-section. This assumption can be obviously violated for real data. The aim of this section is to assess the

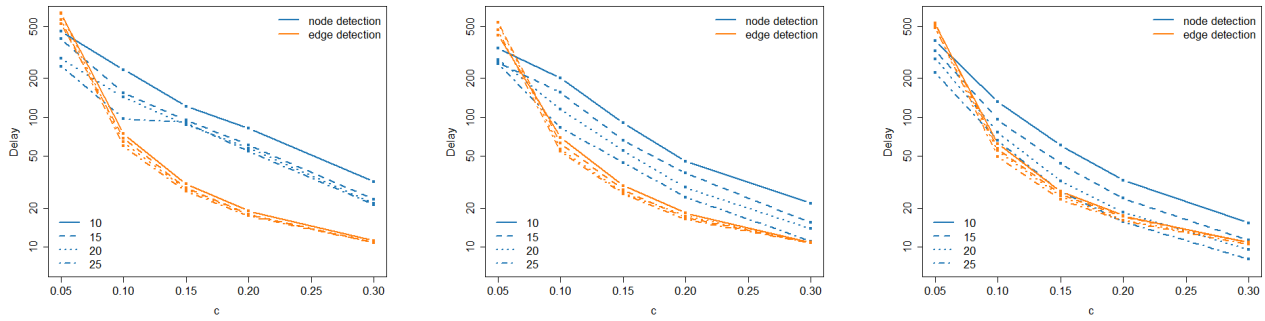


Figure 4: Detection delay initiated by random direction shifts of size c in 10, 15, 20, 25, and 29 (out of 29) edges of the first 2, 3 or 5 nodes (left to right) with node and edge monitoring.

robustness of NSM Algorithm to time-dependent and cross-correlated data. We restrict the discussion to node-initiated shifts and node-detection for undirected networks only. As above we monitor the vector of degrees

$$X_t = (\mathsf{deg}(\nu_{1,t}), \ldots, \mathsf{deg}(\nu_{p,t}))'.$$

For a cross-sectional dependence we impose a spatial correlation structure via

$$X_t \sim \mathrm{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}),$$

where $\boldsymbol{\Sigma} = (\sigma_{ij}aa)_{i,j=1,\ldots,p}$ with $\sigma_{ij} = \rho^{|i-j|}$ and $\sigma_{ii} = 1$ for all i. Since the node degrees are sums of the incident edge weights, we justify the use of normal approximation by the CLT. For consistency with the above results we assume that only three vertices are affected by shifts, i.e. $k = 3$. The results of the simulation study are visualized in the left panel Figure 5. Since the spatial correlation is typically characterized by positive dependence, we restict ρ to the unit interval. The figure reveals that the delay only slowly increases as a function of ρ, whereas this increase becomes more evident in absolute terms for small shift sizes and extreme correlations.

To render temporal dependence in the node degrees we impose an autoregressive dynamics on X_t in the from

$$X_t = \boldsymbol{\mu}_t + \boldsymbol{\varepsilon}_t$$

with $\boldsymbol{\varepsilon}_t = (\varepsilon_{1t}, ..., \varepsilon_{pt})'$ such that

$$\varepsilon_{it} = \alpha\varepsilon_{i,t-1} + \sqrt{1-\alpha^2}u_{it}, \qquad \text{and} \quad u_{it} \sim \mathrm{N}(0,1).$$

Scaling the residual u_{it} guarantees constant variance of the node degrees and comparability of the results. The right panel of Figure 5 shows a relatively strong variation in the delays if the autoregressive parameter α increases from -0.3 to 0.3. Although, a decrease in the delay appears to be advantageous, this trend simultaneously increases the false alarm rate. To lessen the impact of temporal dependence we can correct the time series of node degrees by fitting an appropriate time series model to pre-change data and monitoring the filtered residuals. Alternatively, a self-normalized version of the CUSUM transformation as suggested in Shao and Zhang (2010); Wang and Shao (2020) is capable of handling time-dependent data and can be directly applied in the considered setting too.

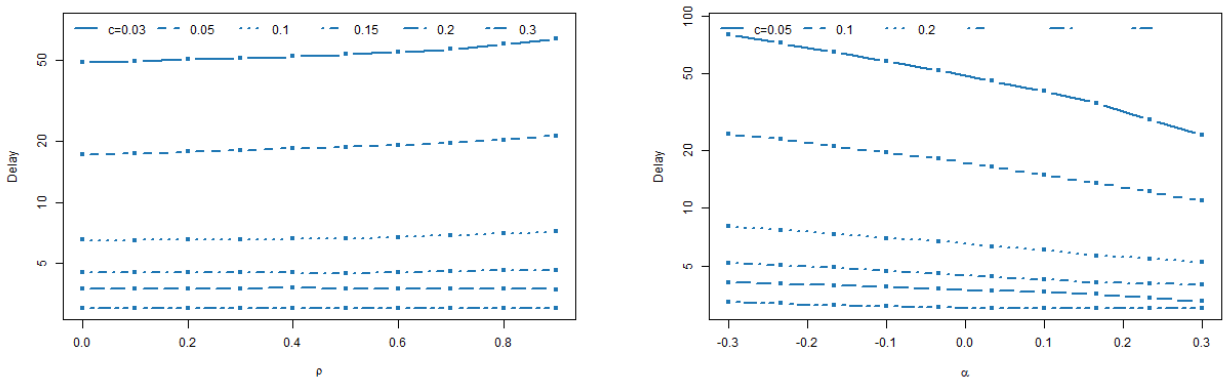


Figure 5: Detection delay of shifts in the first three nodes assuming spatial (left) and temporal (right) dependence of the node degrees.

4 An application to social media networks

4.1 Social media data

Among social media platforms, we are particularly interested in StockTwits for a number of reasons. Firstly, by the construction and design in this microblogging, researchers can easily spot the targeted symbols. The messages are organized around 'cashtags' (e.g., '$AAPL'

for APPLE; '$BTC.X' for BITCOIN) that allow investigators to narrow down streams on specific assets. The required cashtages constitute an effective reference between the message content and the referring stock symbols, as a way of indexing people's thoughts and ideas about a company and stock. Stocktwits users can express their sentiment/opinions by labelling their messages as 'Bearish' (negative) or 'Bullish' (positive) *via* a toggle button. These are so-called *self-report sentiments* which makes their messages semantically explicit.

It's is similar to Twitter but dedicates to the financial discussion. Individuals, investors, market professionals can publish 140-character messages to 'Tap into the Pulse of the Market and financial assets'. With an emphasis on financial discussions, it gains popularity from investors and traders. We limit our attention to cryptocurrency for the presence of 'bubble' during the end of 2017 and the beginning of 2018, known as a speculative bubble. The formation and burst of a bubble have been studied in Hafner (2020); Cavaliere et al. (2020); Chen and Hafner (2019), and more are in production.

Since 2014 StockTwits adds streams and symbology for cryptocurrencies and tokens, expanding from 100 cryptos to more than 400 cryptos recently. New cryptocurrencies are regularly added to the list of cashtags supported by StockTwits.[1] A cashtag refers to a cryptocurrency if and only if it ends with ".X" (e.g. $BTC.X for Bitcoin, $LTC.X for Litecoin). We use this convention and StockTwits Application Programming Interface (API) to download all messages containing a cashtag referring to a cryptocurrency. StockTwits API also provides for each message its user's unique identifier, the time it was posted at with a one-second precision, and the sentiment associated by the user ("Bullish", "Bearish" or unclassified).

Investors' interest in cryptos, revealed by message volumes, is clearly skewed toward to top 30 cryptos as shown in Table 2. These 30 leading cryptos have attracted 1,575,205 messages, amount to 48% of total retrieved messages across 425 cryptos in a time span 23.07.2017 —31.12.2018. Among these 30 coins, Bitcoin stands out the most popular coin from other alternative coins and has attracted around 1,141 messages per day. Roughly speaking, the messages relating to the top 6 cryptos predominate the opinions in the crypto community. We then decide to emphasize on these 30 coins in a consideration of message

[1]This list can be found at https://api.stocktwits.com/symbol-sync/symbols.csv

Figure 6: Wordcloud of messages

volumes.

The retrieved messages are unstructured and need to be pre-processed by using the Python NLTK toolkit. The text normalization begins with the segmentation of words, that can be executed by word-level tokenization. Word tokenization is the process of breaking the text down into its word-based unit, and this can be done easily by NLTK tokenize package.

Furthermore, each message is converted to lower case, non-alphabetic characters are removed and each word is lemmatized. Lemmatization takes the word's part of speech into account while converting the inflected word into its root. In addition to building the vocabulary from single tokens, also known as 1-grams, we consider the option of taking 2-grams into account. Figure 6 presents the wordcloud visualization for the lemmatized tokens up to 2-grams. The size of token depends on its occurrence in the corpus.[2]

4.2 Messages' similarity matrix as network

Social media is an ideal venue to build up social networks comprising of users or symbols as nodes. Once the vertex is chosen by the research of interest as "symbol" rather than

[2]Faws news is specialized for crypto news.

Crypto	message volume	percentile	volume per day
BTC.X	597045.0	0.184	1141.58
LTC.X	201884.0	0.062	386.01
TRX.X	172673.0	0.053	330.16
XRP.X	148531.0	0.046	284.00
ETH.X	136695.0	0.042	261.37
XVG.X	72552.0	0.022	138.72
BCH.X	36400.0	0.011	69.60
XLM.X	21791.0	0.007	41.67
NEO.X	20845.0	0.006	39.86
ADA.X	19889.0	0.006	38.03
IOT.X	16247.0	0.005	31.07
NYC.X	13645.0	0.004	26.09
DOGE.X	13608.0	0.004	26.02
ETC.X	10422.0	0.003	19.93
POE.X	9829.0	0.003	18.79
EOS.X	9712.0	0.003	18.57
RDD.X	9630.0	0.003	18.41
BNB.X	7072.0	0.002	13.52
PAC.X	6259.0	0.002	11.97
VET.X	6058.0	0.002	11.58
XMR.X	6017.0	0.002	11.50
DASH.X	4689.0	0.001	8.97
LEND.X	4636.0	0.001	8.86
ICX.X	4557.0	0.001	8.71
OST.X	4489.0	0.001	8.58
ZRX.X	4159.0	0.001	7.95
SC.X	4090.0	0.001	7.82
IOST.X	3959.0	0.001	7.57
BCN.X	3938.0	0.001	7.53
FUN.X	3884.0	0.001	7.43
Total	1,575,205	0.481	

Table 2: Cryptocurrencies are ranked by its message volumes in StockTwits

"user", a direct network is no longer observable. To cope with it, one may try to infer the hidden edges in the network by measuring the similarity between two symbols, in terms of user characteristics and their mentioning information regarding two symbols. In other words, the network to be formed makes use of social media message information as to the extent to which the expressed message tone/sentiment of symbol i resembles that of symbol j. We then infer nodes i, j are connected in terms of a shared opinion between them.

To represent a text document, one of heuristics is to convert the text into a bag of word representation in the form of word count vector. Statistically speaking, a document $\mathbf{x}_i = (x_{i,1}, \cdots, x_{w,1}, \cdots x_{i,V})$ is a vector of word counts where $x_{i,w}$ is the number of appearance of word w in document i, and V is the vocabulary size in the corpus. To ensure that the inclusive words are discriminative, we remove stopwords known as a set of commonly used words.

However, such simple way may come with the cost such as an artificial similarity caused by word frequency. The TF-IDF (Term Frequency – Inverse Document Frequency) function is a toolkit for improving word count representation. The term frequency is defined as a log-transform of the word count:

$$\text{tf}(x_{i,w}) \equiv \log(1 + x_{i,w}) \tag{2}$$

This mitigates the impact of words that occur many times within one document. The inverse document frequency (IDF) is then defined as:

$$\text{idf}(w) \equiv \log \frac{N}{1 + \sum_{i=1}^{N} \mathbf{1}(x_{i,w} > 0)} \tag{3}$$

where N is the total number of documents. The IDF is based on counting the number of documents in the collection being searched which contain the term in question. The intuition was that a query term which occurs in many documents is not a good discriminator and should be given less weight than one which occurs in few documents, and the measure was a heuristic implementation of this intuition. A consolidated term weighting scheme is the product of term frequency and inverse document frequency.

$$\text{tf-idf}(\mathbf{x}_i) \equiv \left[\text{tf}(x_{i,w}) \times \text{idf}(w)\right]_{w=1}^{V} \tag{4}$$

Thereby, the raw space vector $\mathbf{x}_i$ is transferred to another feature space vector tf-idf($\mathbf{x}_i$), provided by the feature function $\phi(\mathbf{x}) = \text{tf-idf}(\mathbf{x})$.

Having the represented feature vectors, one can undertake a computation for to what the extent two documents are similar in terms of the use of vocabulary and their occurrence. Thanks to the kernel method that operates the objects of interest (text, a bag of words) in a high dimensional feature space, we can perform an inner product of pairwise feature vectors without ever computing the coordinates of the raw data in that space for a sake of computational efficiency, known as "kernel trick". That is the reason for how it gains popularity.

One of well-known kernel function for document classifications is the **cosine similarity** which is defined by

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \frac{\phi(\mathbf{x}_i)'\phi(\mathbf{x}_j)}{\|\phi(\mathbf{x}_i)\|_2 \|\phi(\mathbf{x}_j)\|_2} \tag{5}$$

where $\phi(\mathbf{x}) = \text{tf-idf}(\mathbf{x})$. The quantity in (5) measures the cosine of the angle between two documents, each of which is represented by a feature vector. Since $\mathbf{x}_i$ or even $\phi(\mathbf{x}_i)$ is non-negative, the corresponding cosine similarity is in the range between zero (documents are orthogonal) and one (overwhelmingly similar).

4.3 Monitoring changes in cryptocurrency network dynamics

We apply the sequential network change detection method along with NSM algorithm to the prescribed social media network that evolves over time. The undirected and weighted network comprises of the fixed number of nodes in Table 2 and the edge between a pair of nodes is their messages' similarity. Messages arrive sequentially and text of messages vary across cryptocurrencies and time, constituting time-varying weighted edges and eventually a dynamic network. Starting from 23.07.2017, we sequentially, upon newly arrived messages, monitor a change of network dynamics until 31.12.2018. As such, given a weekly frequency, observation, n, can grow up to 78 during an investigation period. Figure 7 displays the selected snapshots of the network at the particular calendar day and confirms the dynamics of the network.

The detected network change point is at the 28-th observation, 2018-01-14, supported by Figure 9 where one observes the evolvement and growth rate of $\|T^{(t)}\|_\infty$, the maximum absolute-valued entry in $T^{(t)}$ that is a CUSUM transformation of X_t. The maximum CUSUM-transformed degrees in Panel (a) exhibits a location shift at the end of 2017. In

Panel (b) using the CUSUM-transformed edges, one also observes a persistent growth rate from the end of 2017 until June 2018. The corresponding computed sine angle between two non-zero projection vectors are displayed for detecting the change points. The green line indicates the estimated change-point location, 2018-01-14, corresponding to the threshold $\max_{1\leq i\leq b_t} A^{(t-i+1)} < 1/2$ where $A^{(t)} := \sin\angle(\hat{v}^{(t)}, \hat{v}^{(t-1i)})$. Interestingly, both degree- and edge-based detection manifest a similar evolvement of $A^{(t)}$, with a sharp drop straight to almost zero at the change-point location, and there is no further fluctuation after that point.

A consistent result between the degree- and edge-based detection method implies the underlying structure of change. The implied signal level is supposed to be substantial, which is analogous to the simulation in S1 and S2 of Table 1. The detection can be validated by an offline fashion. In line with Figure 7, the structures of the networks before 2018-01-14 are dissimilar to those afterwards. Prior to the change point, the networks are centred by a set of nodes and others are rather disconnected, whereas after the change point the networks appear to be fully connected. From a posteriori perspective, to quantify the signals over time one can deploy a binary segmentation over an entire period, known as a global investigation, and calculate the $\mu_t^{(1)}$ and $\mu_t^{(2)}$ and the corresponding signal, the Euclidean norm between $\mu_t^{(1)}$ and $\mu_t^{(2)}$. Figure 8 shows a rising signal until February 2018 and it declines afterwards.

An inspection of the leading left singular vectors via Figure 10a brings additional insights as to the extent which nodes are responsible for a change. Before the change being declared, the coordinates(nodes) evenly appear in the projection vectors, implying their homogeneous representability. Whenever there is a change, the projection vectors are led by a few nodes and this projection constellation lasts during the post-change period. We observe that some nodes remain its dominance over other nodes. For instance, BTC.X(Bitocin, the most bottom one) persistently acts as one of the supports. Other nodes that appear with persistence include ICON.X(ICON), ETH.X(Ethereum) and POE.X(Po.et). The characteristics of these cryptos persistently driving the projection can be studied in the future study. By comparison, the constellation of projection vectors in the case of an edge-initiated change in Figure 10b looks similar. In terms of sparsity, only 14% of edges drive the change,

which seems quantitatively comparable to the node-initiated one, that is, 16% nodes are in charge of the change. It is worth noting in the edge-initiated change that the responsible edges can be further clustered to stratify the responsible nodes for a monitoring convenience in practice. We decide to leave it to future research.

5 Conclusion

In this paper, we consider the problem of surveillance and detection of structural breaks in dynamic networks. In contrary to the modelling-related issues and ex-post change-point tests, which are widely discussed in the literature, we focus on sequential monitoring and aim to detect the change as soon as possible after its occurrence. Due to a specific structure of a network, we cannot apply the techniques commonly used for monitoring multivariate data. To overcome this problem we deploy the idea of optimal projection direction, that assumes a sparse change in a small subset of nodes or edges which is consistent with the empirical evidence from economic and financial networks. In practice, we can frequently hypothesise that a change is driven either by a few nodes of the network or by specific edges leading to a node- or edge-initiated shifts. To reflect this insight we suggest node- and edge-based monitoring schemes, where the objects of interest are the vectorised adjacency matrix or the vector of node degrees respectively. We recommend using the node-detection if we believe that all edges incident to a few nodes are affected by a shift or if the direction of the shift is not random, implying links getting either only weaker or only stronger. It is important to stress, that the NSM algorithm is capable of detecting both abrupt and smooth shifts in the network characteristics and thus can be used under very heterogeneous economic conditions. Additionally to the practical relevance of the problem, the suggested approach relies on a solid theoretical background. We show that with a suitable choice of parameters we attain the prespecified false alarm rate and declare a signal with high probability within a short time after an actual shift.

In an extensive simulation study, we confirm several important features of the algorithm. First, the detection delays are particularly short for fixed-sign changes and even for small sizes of the shifts. Second, the algorithm is computationally efficient and requires a few minutes even in the case of edge-based detection for networks of moderate size. Third, the

potential spatial dependence between the nodes and/or edges has only a very minor impact on the detection delay and can be mostly neglected in applications. The impact of time-dependent network characteristics is slightly stronger and might lead to more frequent false alarms. Monitoring the social media data in the empirical illustration reveals the relevance of the developed methodology for practical applications. Particularly, the core idea of optimal projection direction allows for deep insights into the structure of shifts and for a more precise economic interpretation of the results.

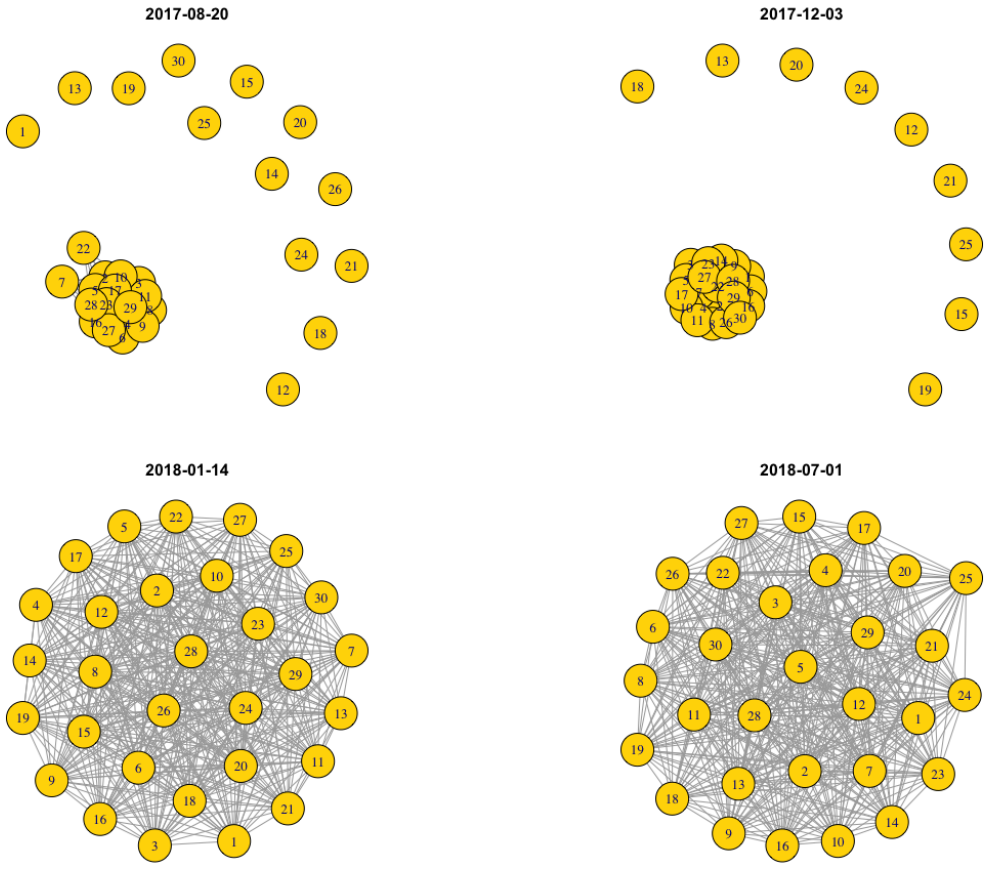


Figure 7: Snapshots of network dynamics

2018-01-14 is the detected change point by the sequential network change test.

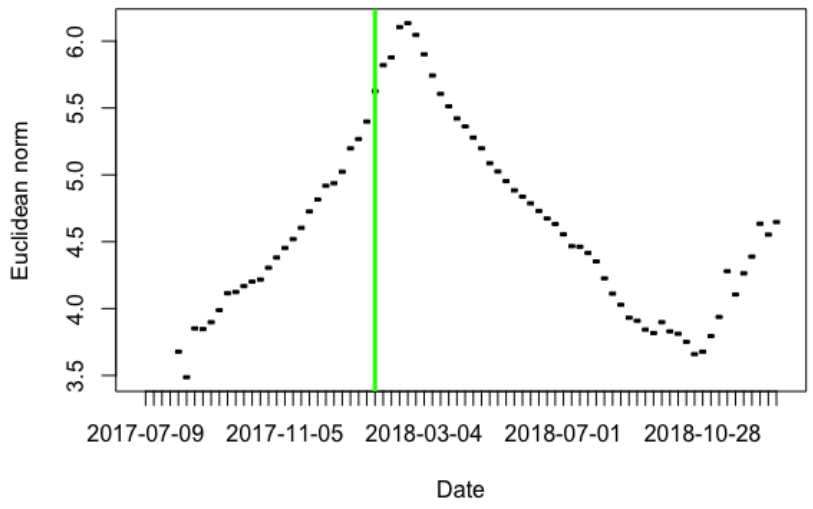


Figure 8: Signals in nodes' degree

The signal, $\rho_t := \|\mu_t^{(1)} - \mu_t^{(2)}\|_2$, is quantified by a binary segmentation between $(1, t)$ and $(t+1, n)$.

6 Proof of theoretical results

Proof of Theorem 1. We consider the following event

$$\Omega_1 := \bigcap_{t=2}^{\infty} \{\|T^{(t)}\|_\infty \leq \lambda_n\}. \tag{6}$$

By Lemma 4 of Wang and Samworth (2018), we have for each $t \geq 2$ and $\lambda > 0$ that

$$\mathbb{P}(\|T^{(t)}\|_\infty > \lambda) \leq \sqrt{\frac{2}{\pi}} p \lceil \log t \rceil (\lambda + 2/\lambda) e^{-\lambda^2/2}.$$

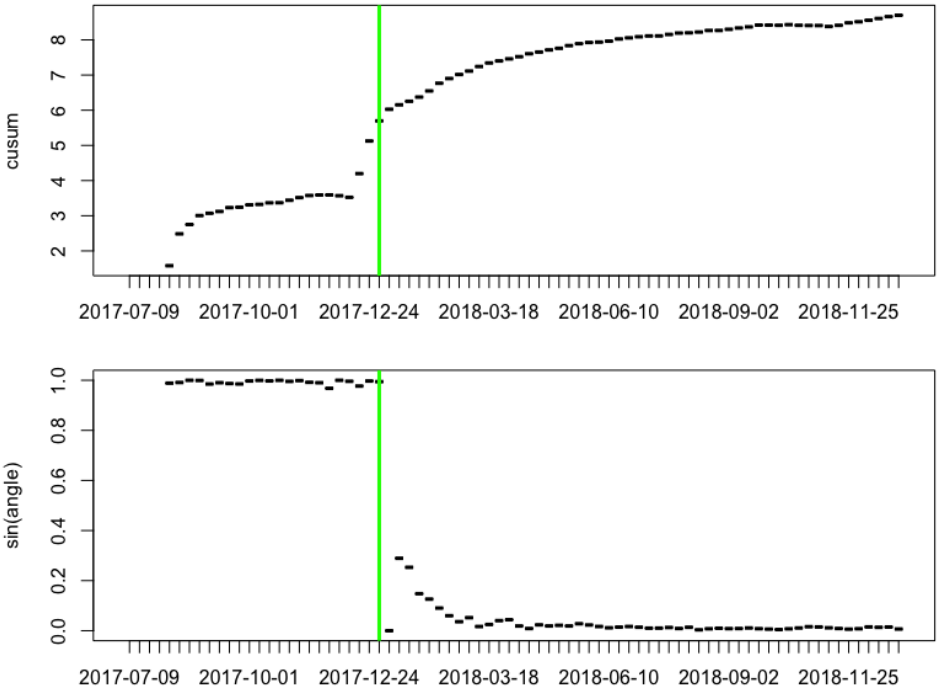
Since $\lambda \mapsto (\lambda + 2/\lambda)e^{-\lambda^2/4}$ is decreasing for $\lambda \in (0, \infty)$ and $\lambda_t = 2\sqrt{\log(t^2 p/\alpha)} \geq 2\sqrt{\log 8}$ for $t \geq 2$ and $\alpha \leq 1/2$, we have $(\lambda_t + 2/\lambda_t)e^{-\lambda_t^2/4} \leq 0.45$. Therefore,

$$\mathbb{P}(\|T^{(t)}\|_\infty > \lambda_t) \leq \sqrt{\frac{2}{\pi}} p \lceil \log t \rceil 0.45 e^{-\lambda_t^2/4} \leq \frac{0.36\alpha \lceil \log t \rceil}{t^2}.$$

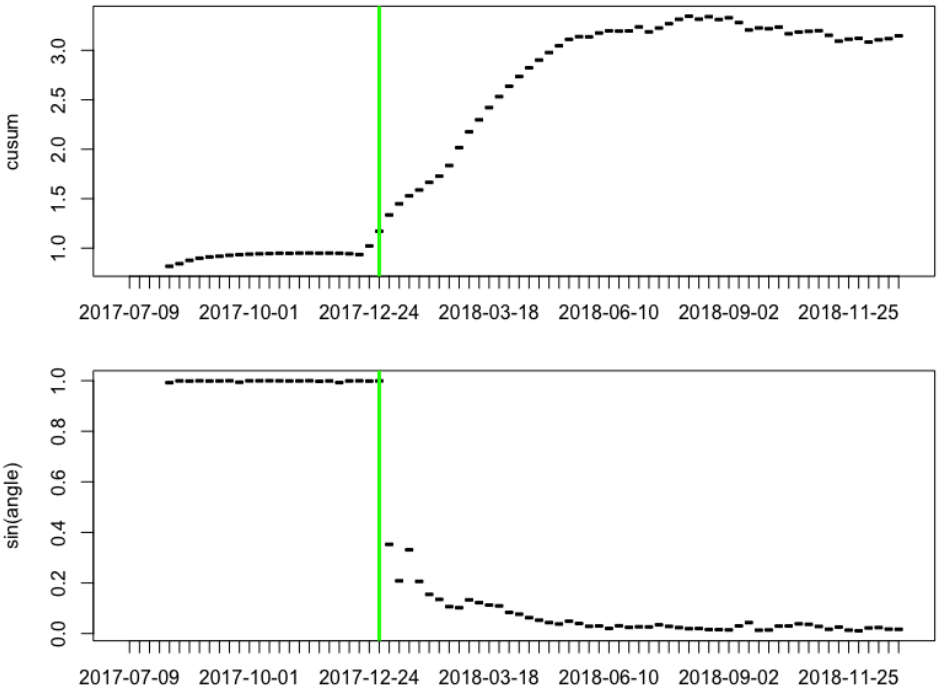
Taking a union bound of the above inequality for all $t \geq 2$, we obtain that

$$\mathbb{P}(\Omega_1^{\mathrm{c}}) \leq \sum_{t=2}^{\infty} \frac{0.36\alpha \lceil \log t \rceil}{t^2} \leq 0.451\alpha.$$

Working on the event Ω_1, we have $\hat{v}^{(t)} \overset{\text{iid}}{\sim} \mathrm{Unif}(\mathbb{S}^{p-1})$ for all $1 \leq t \leq \tau$. By rotational symmetry, we have $A^{(t)} \overset{\mathrm{d}}{=} \sin \angle(\hat{v}^{(t)}, e_1)$ for $1 \leq t \leq \tau$ where e_1 is the first standard basis



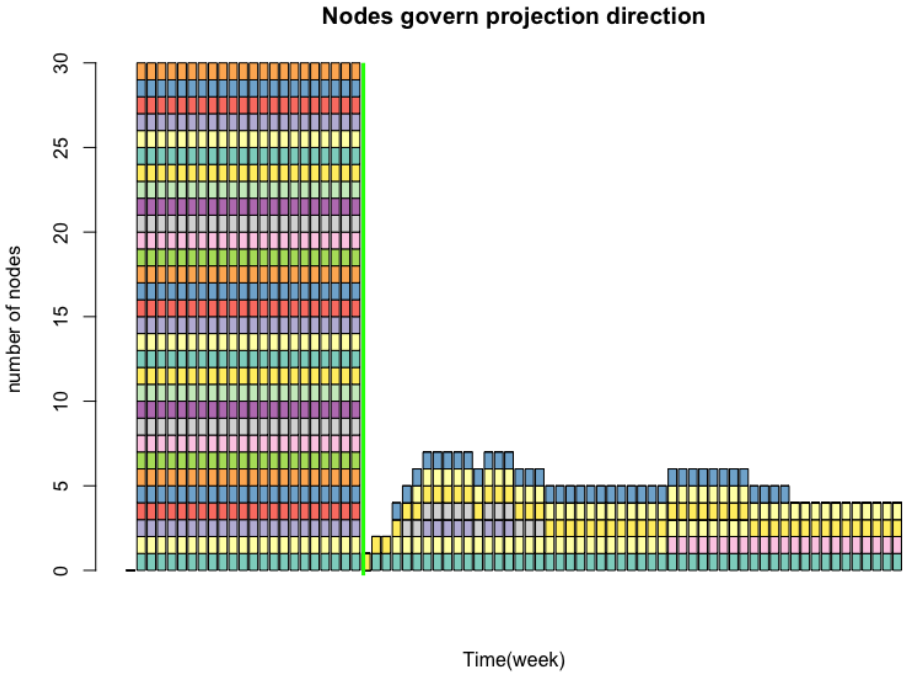
(a) Degree-based detection



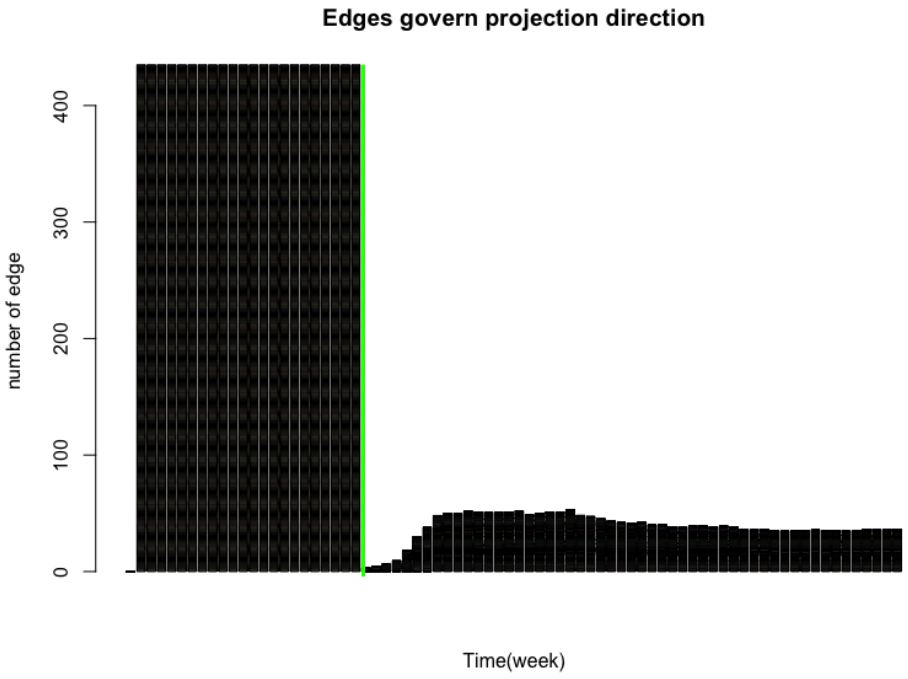
(b) Edge-based detection

Figure 9: Monitoring changes in dynamics of cryptocurrency networks

The green line indicates the location of change.



(a) Degree-based detection



(b) Edge-based detection

Figure 10: Dynamics of projection vectors

The bars represent the coordinates which are non-zero valued in the projection vectors. The emergence of coordinates change over time. The green line indicates the location of change.

vector in $\mathbb{R}^p$. In particular, $A^{(t)}$ is independent of $v^{(t-1)}$ and consequently $A^{(1)}, \ldots, A^{(\tau)}$ are independent and identically distributed. Let $Z \sim N_p(0, I_p)$. We see that

$$\langle v^{(t)}, e_1 \rangle^2 \stackrel{\mathrm{d}}{=} \frac{Z_1^2}{Z_1^2 + \cdots + Z_p^2} \sim \mathrm{Beta}\left(\frac{1}{2}, \frac{p-1}{2}\right).$$

Using a Beta distribution tail bound (see, e.g. Marchal et al., 2017, Theorem 2.1), we have

$$\mathbb{P}(A^{(t)} < 1/2 \mid \Omega_1) \le \mathbb{P}\left\{\mathrm{Beta}\left(\frac{1}{2}, \frac{p-1}{2}\right) \ge \frac{3}{4}\right\} \le e^{-2(p/2+1)(3/4-1/p)^2} \le e^{-p/8}.$$

Consequently, we have

$$\begin{aligned}\mathbb{P}(N \le \tau) &\le \mathbb{P}(\Omega_1^{\mathrm{c}}) + \sum_{t \ge 1 : t \ge b_t} \prod_{t - b_t + 1 \le i \le t} \mathbb{P}(A^{(i)} < 1/2 \mid \Omega_1) \\ &\le 0.451\alpha + \sum_{t=1}^{\infty} e^{-p b_t/8} \le 0.451\alpha + \sum_{t=1}^{\infty} \frac{\alpha}{3t^2} \le \alpha,\end{aligned}$$

as desired. □

Proof of Theorem 2. We work on the event Ω_1 defined in (6), which satisfies $\mathbb{P}(\Omega_1^{\mathrm{c}}) \le \alpha$ as shown in the proof of Theorem 1. We will show that the desired response delay bound holds on the event Ω_1.

Writing $v := \theta / \|\theta\|_2$, by the proof of Proposition 1 of Wang and Samworth (2018), we have on Ω_1 and for all $t > \tau$ that

$$\sin \angle(\hat{v}^{(t)}, v) \le \frac{32\lambda_t \sqrt{st}}{\min\{\tau, t-\tau\}\rho}.$$

In particular, for all t satisfying

$$\tau + \frac{2^{15/2}\lambda_t \sqrt{s\tau}}{\rho} \le t \le 2\tau, \tag{7}$$

we have

$$\sin \angle(\hat{v}^{(t)}, v) \le \frac{32\lambda_t \sqrt{2s\tau}}{(t-\tau)\rho} \le \frac{1}{4}.$$

By triangle inequality, we then have

$$A^{(t)} = \sin \angle(\hat{v}^{(t)}, \hat{v}^{(t-1)}) \le \sin \angle(\hat{v}^{(t)}, v) + \sin \angle(v, \hat{v}^{(t-1)}) \le 1/2,$$

for all but the first t satisfying (7). Condition (1) ensures that the set of t satisfying (7) has cardinality at least $b_{2\tau} + 1$. Consequently, we have on event Ω_1 that

$$N \le \tau + 2^{15/2}\lambda_{2\tau}\sqrt{s\tau}\rho + b_{2\tau},$$

as desired. □

References

Aue, A., S. Hörmann, L. Horváth, and M. Reimherr (2009). Break detection in the covariance structure of multivariate time series models. *Ann. Statist.*, 4046–4087.

Barnard, G. A. (1959). Control charts and stochastic processes. *J. Roy. Statist. Soc., Ser. B 21*, 239–271.

Bücher, A., I. Kojadinovic, T. Rohmer, and J. Seger (2014). Detecting changes in cross-sectional dependence in multivariate time series. *J. Mult. Anal. 132*, 111–128.

Cavaliere, G., H. B. Nielsen, and A. Rahbek (2020). Bootstrapping noncausal autoregressions: with applications to explosive bubble modeling. *J. Bus. Econom. Statist. 38*(1), 55–67.

Chen, C. Y.-H. and C. M. Hafner (2019). Sentiment-induced bubbles in the cryptocurrency market. *J. Risk Financ. Manag. 12*(2), 53.

Chen, C. Y.-H., W. K. Härdle, and Y. Klochkov (2019). Sonic: Social network with influencers and communities. *Available at SSRN 3657360*.

Chen, Y., T. Wang, and R. J. Samworth (2020). High-dimensional, multiscale online changepoint detection. *arXiv preprint*, arxiv:2003.03668.

Cho, H. (2016). Change-point detection in panel data via double cusum statistic. *Electron. J. Statist. 10*, 2000–2038.

Cho, H. and P. Fryzlewicz (2015). Multiple-change-point detection for high dimensional time series via sparsified binary segmentation. *J. Roy. Statist. Soc., Ser. B 77*, 475–507.

Dette, H., G. Pan, and Q. Yang (2020). Estimating a change point in a sequence of very high-dimensional covariance matrices. *J. Amer. Statist. Assoc.*, to appear.

Duncan, A. J. (1952). *Quality Control and Industrial Statistics*. Chicago: Richard D. Irwin Inc.

Enikeeva, F. and Z. Harchaoui (2019). High-dimensional change-point detection under sparse alternatives. *Ann. Statist. 47*(4), 2051–2079.

Hafner, C. M. (2020). Testing for bubbles in cryptocurrencies with time-varying volatility. *J. Financ. Econom. 18*(2), 233–249.

Han, X., C.-S. Hsieh, and S. I. Ko (2019). Spatial modeling approach for dynamic network formation and interactions. *J. Bus. Econom. Statist.*, to appear.

Horváth, L. and M. Hušková (2012). Change-point detection in panel data. *J. Time Series Anal. 33*, 631–648.

Jirak, M. (2015). Uniform change point tests in high dimension. *Ann. Statist. 43*, 2451–2483.

Jochmans, K. (2018). Semiparametric analysis of network formation. *J. Bus. Econom. Statist. 36*(4), 705–713.

Lavielle, M. and G. Teyssiere (2006). Detection of multiple change-points in multivariate time series. *Lith. Math. J. 46*, 287–306.

Lux, T. (1995). Herd behaviour, bubbles and crashes. *The Economic Journal 105*(431), 881–896.

Marchal, O., J. Arbel, et al. (2017). On the sub-gaussianity of the beta and dirichlet distributions. *Electron. Commun. Probab. 22*.

Page, E. S. (1954). Continuous inspection schemes. *Biometrika 41*, 100–115.

Preuß, P., R. Puchstein, and H. Dette (2015). Detection of multiple structural breaks in multivariate time series. *J. Amer. Statist. Assoc. 110*, 654–668.

Shao, X. and X. Zhang (2010). Testing for change points in time series. *J. Amer. Statist. Assoc. 105*(491), 1228–1240.

Wang, D., Y. Yu, and A. Rinaldo (2020). Optimal change point detection and localization in sparse dynamic networks. *Ann. Statist.*, to appear.

Wang, D., Y. Yu, A. Rinaldo, and R. Willett (2019). Localizing changes in high-dimensional vector autoregressive processes. *arXiv preprint*, arxiv:1909.06359.

Wang, R. and X. Shao (2020). Dating the break in high-dimensional data. *arXiv preprint*, arxiv:2002.04115.

Wang, T. and R. Samworth (2016). *InspectChangepoint: high-dimensional changepoint estimation via sparse projection.* R package version 1.1.

Wang, T. and R. J. Samworth (2018). High dimensional change point estimation via sparse projection. *J. Roy. Statist. Soc., Ser. B 80*(1), 57–83.

Zhang, N. R., D. O. Siegmund, H. Ji, and J. Z. Li (2010). Detecting simultaneous changepoints in multiple sequences. *Biometrika 97*, 631–645.

Zhu, X., R. Pan, G. Li, Y. Liu, H. Wang, et al. (2017). Network vector autoregression. *Ann. Statist. 45*(3), 1096–1123.

Zou, C., Z. Wang, X. Zi, and W. Jiang (2015). An efficient online monitoring method for high-dimensional data streams. *Technometrics 57*, 374–387.