# Learning Mixtures of Graphs from Epidemic Cascades

Jessica Hoffmann [1] Soumya Basu [1] Surbhi Goel [1] Constantine Caramanis [1]

## Abstract

We consider the problem of learning the weighted edges of a balanced mixture of two undirected graphs from epidemic cascades. While mixture models are popular modeling tools, algorithmic development with rigorous guarantees has lagged. Graph mixtures are apparently no exception: until now, very little has been known about whether this problem is solvable.

To the best of our knowledge, we establish the first *necessary and sufficient* conditions for this problem to be solvable in polynomial time on edge-separated graphs. When the conditions are met, $i.e.$, when the graphs are connected with at least three edges, we provide an efficient algorithm for learning the weights of both graphs with optimal sample complexity (up to log factors).

We give complementary results and provide sample-optimal (up to log factors) algorithms for mixtures of $directed$ graphs of out-degree at least three, for mixtures of undirected graphs of unbalanced and/or unknown priors.

## 1. Introduction

Epidemic models represent spreading phenomena on an underlying graph (Newman, 2014). Such phenomena include diseases spreading through a population, security breaches in networks (malware attacks on computer/mobile networks), chains of activations in various biological networks (activation of synapses, variations in the levels of gene expression), circulation of information/influence (rumors, (fake) news, viral videos, advertisement campaigns), and so on.

Most settings assume the underlying graph is known ($e.g.$, the gene regulatory network), and focus on modeling epidemics (Del Vicario et al., 2016; Wu & Liu, 2018; Gomez-Rodriguez et al., 2013; Cheng et al., 2014; Zhao et al., 2015; Liu et al., 2019), detecting them (Arias-Castro et al., 2011; Arias-Castro & Nov; Milling et al., 2015; 2012; Meirom et al., 2014; Leskovec et al., 2007; Khim & Loh, 2017), detecting communities (Prokhorenkova et al., 2019; Xie et al., 2019), finding their source (Shah & Zaman, 2010a; 2012; 2010b; Spencer & Srikant, 2015; Wang et al., 2014; Sridhar & Poor, 2019; Dong et al., 2019), obfuscating the source, (Fanti et al., 2016; 2015; 2017), or controlling their spread (Kolli & Narayanaswamy, 2019; Drakopoulos et al., 2014; 2015; Hoffmann & Caramanis, 2018; Farajtabar et al., 2017; Wang et al., 2019; Yan et al., 2019; Ou et al., 2019).

The inverse problem, learning the graph from times of infection during multiple epidemics, has also been extensively studied. The first theoretical guarantees were established by Netrapalli and Sanghavi (Netrapalli & Sanghavi, 2012) for discrete-time models. Abrahao et al. (Abrahao et al., 2013) tackled the problem for some continuous-time models, for exponential distributions. Daneshmand et al. (Daneshmand et al., 2014) solved the problem for a wide class of continuous models which fit real-life diffusions. Pasdeloup et al. (Pasdeloup et al., 2017) characterized a set of graphs for which this problem is solvable using spectral methods. Khim and Loh (Khim & Loh, 2018) solved the problem for correlated cascades. Subsequently, Trouleau et al. (Trouleau et al., 2019) showed how to learn the causal structure of Hawkes processes under synchronization noise. In parallel, Hoffmann and Caramanis (Hoffmann & Caramanis, 2019) showed that it is possible to robustly learn the graph from noisy epidemic cascades, even in the presence of arbitrary noise.

However, this line of research always assumes that the epidemic cascades are all of the same kind, and spread on one unique graph which entirely captures the dynamics of the spread. In reality, our observations of cascades are far more granular: different kinds of epidemics spread on the same nodes but through different mechanisms, i.e., different spreading graphs. Epidemic cascades we observe are often a *mixture* of different kinds of epidemics. Without knowledge of the *label* of the epidemic, can we recover the individual spreading graphs? For a concrete example, let us consider the ubiquitous Twitter graph. Individuals usually have multiple interests, and will share tweets differently according to the underlying topics of the tweets. For instance,

[1] University of Texas at Austin. Correspondence to: Jessica Hoffmann <hoffmann@cs.utexas.edu>.

*Proceedings of the $37^{th}$ International Conference on Machine Learning*, Vienna, Austria, PMLR 119, 2020. Copyright 2020 by the author(s).

two users may have aligned views on football and diametrically opposed political views, and hence may retweet each others' football tweets but not political posts. Interesting settings are those where the epidemic labels (in this simple case, *football* and *politics*) is not observable. While *football* and *politics* may be easy to distinguish via basic NLP, the majority of settings will not enjoy this property (e.g., she retweets football posts relating to certain teams, outcomes, or special plays). In fact, the focus on recovering the spreading graph stems precisely from the desire to study very poorly-understood epidemics whose spreading mechanisms, symptoms, etc. remain elusive. Examples outside the Twitter realm ($e.g.$, human epidemics with multiple spreading vectors) abound.

In such cases, applying existing techniques for estimating the spreading graph would recover the union of graphs in the mixture. For Twitter and other social networks, this is essentially already available. However, this union is typically not informative enough to predict the spread of tweets, and may even be misleading.

We address precisely this problem. We consider a mixture of epidemics that spread on two unknown weighted graphs when, for each cascade, the kind of epidemic (and hence the spreading graph) remains hidden. We aim to accurately recover the weights of both the graphs from such cascades.

Mixture models in general have attracted significant focus. Even for the most basic models, e.g., Gaussian mixture models, or mixed regression, rigorous recovery results have proved elusive, and only recently has there been significant progress (e.g., (Balakrishnan et al., 2017; Yi et al., 2014; 2016; Chen et al., 2017; Diakonikolas et al., 2018; Kwon et al., 2019; Xu et al., 2016; Daskalakis et al., 2017; Kwon & Caramanis, 2019)). This work reveals some similarities to prior work. For example, here too, moment-based approaches play a critical role; moreover, here too, there are conditions on separation of the two classes needed for recovery. Interestingly, however, the technical key to our work is much more combinatorial in nature, rather than appealing to more general-purpose tools (like tensor decomposition or EM). As we outline below, the crux of the proof of correctness of our algorithm is a combination of a characterization of *forbidden graphs* that cannot be learned, and a decomposition-reduction of a general graph to smaller subgraphs that can be learned and later patched to produce a globally consistent solution.

### 1.1. Contributions

To the best of our knowledge, this is the first paper to study the inverse problem of learning mixtures of weighted undirected graphs from epidemic cascades. We address the following questions:

**Recovery:** Under the assumption that the underlying graphs are connected, have at least three edges and under some separability condition (detailed in the next section), we prove the problem is solvable and give an efficient algorithm to recover the weights of $any$ mixture of connected graphs with equal priors on the same set of vertices.

**Identifiability:** We show the problem is not solvable in polynomial time of one if the condition mentioned above is violated. The problem is unidentifiable when one of the graphs of the mixture has a connected component with less than three edges. Moreover, there exist (many) graphs which violate the separability condition, and for which any algorithm would require at least exponential (in the number of nodes) sample complexity.

**Sample Complexity:** We prove a lower bound on the sample complexity of the problem, and show that our algorithm always matches the lower bound up to log factors in terms of the number of nodes $N$. It also matches the bound exactly in terms of the dependency in the separation parameter $\frac{1}{\Delta}$ if the graphs have min-degree at least 3.

**Extensions:** We give similar guarantees for the case of directed graphs of min-degree at least 3, and of undirected graphs with unbalanced and/or unknown mixtures priors. Finally, we discuss how to obtain numerical solutions for $K > 2$ mixtures.

## 2. Preliminaries

We consider an instance of the *independent cascade model* (Goldberg et al., 2001; Kempe et al., 2003). We observe independent epidemics spreading on a mixture of two graphs. In this section, we specify the dynamics of the spreading process, the observation model, and the learning task.

### 2.1. Mixture Model

We consider two $weighted$ graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ on the *same set of vertices* $V$. Unless specified otherwise, the graphs considered are $undirected$: $p_{ij} = p_{ji}$ and $q_{ij} = q_{ji}$. Note that $p_{ij}$ ($q_{ij}$) is 0 if there is no edge between $i$ and $j$ in $G_1$ ($G_2$).

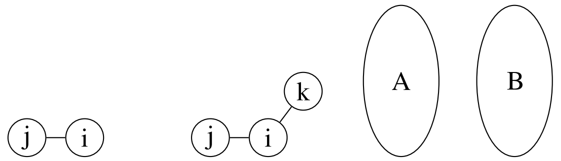
We say that the mixture is $\underline{\Delta\text{-separated}}$ if:

$$\min_{(i,j)\in E_1\cap E_2} |p_{ij} - q_{ij}| \geq \Delta > 0.$$

We denote the minimum edge weight by $p_{min} := \min_{(i,j)\in E_1} \min_{(k,l)\in E_2} \min(p_{ij}, q_{kl})$. Note that $p_{min}$ is positive.

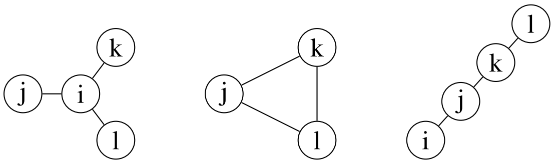
### 2.2. Dynamics of the Spreading Process

We observe $M$ independent identically distributed epidemic cascades, which come from the following generative model.



(a) One edge (b) Two edges (c) Disconnected components

Figure 1: Unsolvable structures



(a) Star (b) Triangle (c) Line

Figure 2: Solvable local structure

**Component Selection:** At the start of a cascade, an i.i.d. Bernoulli random variable $b \in \{1, 2\}$ with parameter $\alpha$ ($\Pr[b = 1] = \alpha$) decides the component of the mixture, $i.e.$, the epidemic spreads on graph $G_b$. We say that the mixture is balanced if $\alpha = 0.5$, and we call $\alpha$ and $1 - \alpha$ the priors of the mixture. Unless specified otherwise, the results presented are for balanced mixtures.

**Epidemic Spreading:** Once the component of the mixture $G_b$ is fixed, the epidemic spreads in *discrete time* on graph $G_b$ according to a regular one-step *Susceptible → Infected → Removed* (SIR) process (Netrapalli & Sanghavi, 2012; Hoffmann & Caramanis, 2019). At $t = 0$, the epidemic starts on a $unique$ source, chosen uniformly at random among the nodes of $V$. The source is in the Infected state, while all the other nodes are in the Susceptible state. Let $I_t$ (resp. $R_t$) be the set of nodes in the Infected (resp. Removed) state at time $t$. At each time step $t \in \mathbb{N}$, all nodes in the Infected state try to infect their neighbors in the Susceptible state, before transitioning to the Removed state during this same time step (*i.e.*, $R_{t+1} = R_t \cup I_t$) [1]. If $i$ is in the Infected state at time $t$, and $j$ is in the Susceptible state at the same time ($i.e\ i \in I_t, j \in S_t$), then $i$ infects $j$ with probability $p_{ij}$ if $b = 1$, and $q_{ij}$ if $b = 2$, with $0 \leq p_{ij} \leq 1$. Note that multiple nodes in the Infected state can infect the same node in the Susceptible state. The process ends at the first time step such that all nodes are in the Susceptible or Removed state (*i.e.*, no node is in the Infected state).

One realization of such a process, from randomly picking the component of the mixture and the source at $t = 0$ to the end of the process, is called a cascade.

### 2.3. Observation Model

For each cascade we do not have the knowledge of the underlying component, that is, we do not observe $b$ and we treat this as a missing label. For each cascade, we have access to the complete list of infections: we know which node infected which node at which time (one node can have been infected by multiple nodes). This list constitutes a sample from the underlying mixture model.

### 2.4. Learning Objective

Our goal is to learn the weights of all the edges of the underlying graphs of the mixture, up to precision $\epsilon < \min(\Delta, p_{min})$. Specifically, we want to provide $\hat{p}_{ij}$ and $\hat{q}_{ij}$ for all vertex pairs $i, j \in V$ such that $\max_{i,j \in V^2} \max(|p_{ij} - \hat{p}_{ij}|, |q_{ij} - \hat{q}_{ij}|) < \epsilon$.

### 2.5. When is this problem solvable?

Prior to presenting our main results, we offer some intuition. We show that it is not always possible to learn the weights of both components of the mixture, even for settings that appear deceptively easy.

Indeed, it is impossible to learn the graph on two nodes $i$ and $j$, with only one directed edge from $i$ to $j$ (see Figure 1a). To see this, consider a balanced mixture, for which edge $(i, j)$ has weight $\beta$ in $G_1$, and weight $1 - \beta$ in $G_2$. Node $i$ will infect node $j$ half of the time, independently of the value of $\beta$. This shows that we cannot recover the original weights, and the mixture problem is not solvable. If we add another edge, and $i$ is now connected to a new node $k$ (see Figure 1b), the problem is still not solvable (see Supplementary Material).

Surprisingly, if $i$ has a third neighbor $l$ (see Figure 2a), it becomes possible to learn the weights of the mixture. Learning this local structure is one of the main building blocks of our algorithm.

One might think that four nodes are needed for this problem to be solvable. However, we can learn the edges of a triangle (see Figure 2b). Similarly, the intuition that nodes need to be of degree at least three is misleading. If a line has more than three nodes (see Figure 2c), it is solvable. The line on four nodes is the other local structure which forms the foundation of our algorithm.

On the other end, the setting for which there exists (at least) two parts of the graph $A$ and $B$ for which cascades never overlap is a general unsolvable setting (see Figure 1c). We write $A_i = A \cap E_i, B_i = B \cap E_i$. Let $E'_1 = A_1 \cup B_2$, $E'_2 = A_2 \cup B_1$. We notice that a mixture spreading on edges $E_1$ and $E_2$ yields the same cascade distribution as a mixture on $E'_1$ and $E'_2$. Therefore, the solution is not unique.

[1] Once a node is in the Removed state, the spread of the epidemic proceeds as if this node were no longer on the graph.

The three simple shapes in Figure 2 form the core of this paper. Our key insight is in showing that any graph that can be built up using these three building blocks (i.e., each node belongs in at least one of these structures) is solvable. This effective decomposition succeeds in reducing a general problem to a small number of sub-problems, for which we provide a solution.

## 3. Main Results

In this section we present our main results on the impossibility and recoverability of edge weights for a balanced mixture.

### 3.1. Balanced Mixture of Undirected Graphs

**Impossibility Result Under Infinite Samples**

**Condition 1.** *The graph $G = (V, E_1 \cup E_2)$ is connected and has at least three edges: $|E_1 \cup E_2| \geq 3$.*

**Claim 1.** *Suppose Condition 1 is violated. Then it is impossible to recover the edge weights corresponding to each graph (even with infinite samples).*

**Impossibility Result Under Polynomial Samples**

**Condition 2.** *The mixture is $\Delta$-separated: for every edge $(i, j) \in E_1 \cup E_2$, $p_{ij} \neq q_{ij}$.*

**Claim 2.** *Suppose Condition 2 is violated. Then there exist (many) graphs for which we need at least exponential (in the number of nodes $N$) samples to recover the edge weights.*

**Recoverability Result with Finite Samples**

**Theorem 1.** *Suppose Conditions 1 and 2 are true. Then there exists an algorithm that runs on epidemic cascades over a balanced mixture of two undirected, weighted graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, and recovers the edge weights corresponding to each graph up to precision $\epsilon$ with probability at least $1 - \delta$, in time $O(N^2)$ and sample complexity $O\left(\frac{N}{\epsilon^2 \cdot \Delta^4} \log(\frac{N}{\delta})\right)$, where $N = |V|$.*

**Remark on Partial Recovery:** An important element of our results is that if Conditions 1 and 2 are not satisfied for the entire graph we can still recover the biggest subgraph which follows these conditions. In particular, if the graph we obtain by removing all non $\Delta$-separated edges is still connected, we can detect and learn all the edges of the graph (see Supplementary Material for more details). This is important, as it effectively means that we are able to learn the mixtures in the parts of the graph that matter most. On a practical note, this also means that our algorithm is resistant to the presence of bots in the network that would repost everything indifferently.

### 3.2. Extensions

**Extension to Directed Graphs** Interestingly, the techniques used to prove the theorem above can be immediately applied to learn mixtures of directed graphs of out-degree at least three (see Supplementary Material for complete proof). Note that the better dependency in $\frac{1}{\Delta}$ comes from the assumption on the degree [2]. Since many applications on social networks can ignore nodes of out-degree less than three, as thoses nodes have very little impact on any diffusion phenomena, this result is of independent interest:

**Theorem 2.** *Suppose Conditions 1 and 2 are true. Then there exists an algorithm that runs on epidemic cascades over a balanced mixture of two* ***directed****, weighted graphs of minimum out-degree three $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, and recovers the edge weights of each graph up to precision $\epsilon$ with probability at least $1 - \delta$, in time $O(N^2)$ and sample complexity $O\left(\frac{N}{\epsilon^2 \cdot \Delta^2} \log(\frac{N}{\delta})\right)$, where $N = |V|$.*

**Extension to Unbalanced/Unknown Priors** If the mixture is unbalanced, but the priors are known, we can adapt our algorithm to learn the mixture *under the same conditions as above*, at the price of a higher dependency in $\frac{1}{\Delta}$. If the priors are unknown, we can only recover graphs of min-degree at least three.

### 3.3. Lower Bounds

We provide two lower bounds for mixtures of two graphs, one for undirected graphs, one for directed graphs.

**Theorem 3.** *When learning the edge weights of a balanced mixture on two $\Delta$-separated graphs on $N$ nodes up to precision $\epsilon < \Delta$, we need:*

1. *$\Omega\left(\frac{N}{\Delta^2}\right)$ samples for undirected graphs, which proves our algorithm is optimal in $N$ up to log factors in this setting.*

2. *$\Omega\left(N \log(N) + \frac{N \log\log(N)}{\Delta^2}\right)$ samples for directed graphs of minimum out-degree three, which proves our algorithm has optimal dependency in $N$ and in $\frac{1}{\Delta^2}$ in this setting.*

## 4. Balanced Mixture of Undirected Graphs

In this section, we provide our main algorithm (Algorithm 1) that recovers the edge weights of the graphs under the conditions presented in Theorem 1.

[2] This immediately implies a better dependency in $\frac{1}{\Delta}$ for learning undirected graphs of minimum degree three.

**Algorithm 1** Learn the weights of undirected edges

```
**Input:** Vertex set $V$
**Output:** Edge weights for the two epidemics graphs
$E \leftarrow$ LEARNEDGES($V$)
$S, W \leftarrow$ LEARN2NODES($V, E$)
**while** $S \neq V$ **do**
    Select $u \in S, v \in V \backslash S$ such that $(u, v) \in E$
    **if** deg(v) $\geq$ 3 **then**
        $W \leftarrow W \cup$ LEARNSTAR($v, E, W$)
    **end if**
    **if** deg(v) = 2 **then**
        Set $w \in S$ such that $(u, w) \in E$
        Set $t \in V$ such that $(v, t) \in E$ and $t \neq u$
        **if** $t \notin S$ **then**
            $W \leftarrow W \cup$ LEARNLINE($t, v, u, w, S, W$)
        **end if**
    **end if**
    $S \leftarrow S \cup \{v\}$
**end while**
**Return** $W$
```

### 4.1. Overview of Algorithm 1

First, the algorithm learns the edges of the underlying graph using the procedure LEARNEDGES. To detect whether an edge $(u, v)$ exists in $E_1 \cup E_2$, we use a simple estimator (Section 4.2.1). This also provides us with the degree of each node with respect to $E_1 \cup E_2$.

With the knowledge of the structure of the graph, to learn the edge weights adjacent to a node, our algorithm uses two main procedures, LEARNSTAR and LEARNLINE. If a node is of degree at least three (*e.g.*, node $u$ in Figure 3), procedure LEARNSTAR recovers all the edge weights (i.e., the weights of the two mixtures for these edges) adjacent to this node *independently* of the rest of the graph. Otherwise, if a node is of degree two (*e.g.*, node $u$ in Figure 4), procedure LEARNLINE learns all the edge weights adjacent to this node *independently*. Both procedures use carefully designed estimators that exploit the respective structures. We present the above estimators for balanced mixtures in Section 4.2. We require Condition 2 for the existence of the proposed estimators.

Our main algorithm maintains a set of *learned nodes*. A node is a *learned node* if the weights for all the edges adjacent to it have been learned. The algorithm begins with learning two connected nodes (two nodes with an edge in between) using procedure LEARN2NODES. Next it proceeds iteratively, by learning the weights of the edges connected to one *unlearned neighbor* of the *learned nodes* using the two procedures discussed above. The algorithm terminates when all the nodes in $V$ are learned.

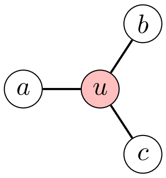


Figure 3: A star vertex $u$, with edges $(u, a)$, $(u, b)$ and $(u, c)$ in $E_1 \cup E_2$.

### 4.2. Learning Edges, Star Vertices, and Line Vertices

In this section, we show how we recover the weights for local structures using moment matching methods. Our proof relies on a few crucial ideas. First, we introduce local estimators, which can be computed from observable events in the cascade and are polynomials of the weights of the mixture. General systems of polynomial equations are hard to solve. However, we found ways of combining these specific estimators to decouple the problem, and obtain $\mathcal{O}(|E|)$ systems of six polynomial equations of maximum degree three, with six unknowns. Finally, we show how to elegantly obtain a closed-form solution for these systems.

#### 4.2.1. LEARNING THE EDGES IN $E_1 \cup E_2$

We recall that $I_0$ is the random variable indicating the set containing the unique source of the epidemic for a cascade. If an epidemic cascade starts from node $u$, then for any node $a$ that is infected in time step 1 there is an edge $(u, a) \in E_1 \cup E_2$. This provides us with the average weight of the edge $(u, a)$ as $X_{ua}$,

**Claim 3.** *If $u$ and $a$ are two distinct nodes of $V$ such that $(u, a) \in E_1 \cup E_2$, then:*

$$X_{ua} := \Pr(u \to a \mid u \in I_0) = \frac{p_{ua} + q_{ua}}{2} \geq \frac{p_{min}}{2}.$$

*Furthermore, there exists an edge between $u$ and $a$ in $E_1 \cup E_2$, if and only if $X_{ua} \geq \frac{p_{min}}{2} > 0$.*

The above claim can be leveraged to design algorithm LEARNEDGES, which takes as inputs all the $X_{ua}$ for all pairs $(u, a)$, and returns all the edges of $E_1 \cup E_2$ (see Supplementary Materials).

**Conditioning on Source Node:** We notice that the expression of $X_{ua}$ is a function of weights of edges $(u, a)$. Here, conditioning on the event "$u \in I_0$" is crucial. Indeed, if the source had been any node other than $u$, the probability that $a$ was in the Susceptible state when $u$ was infected would have depended on the (unknown) weights of the paths connecting the source and node $a$. We could not have obtained the simple expression above.

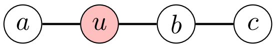


Figure 4: A line vertex $u$, with edges $(u, a)$, $(u, b)$ and $(b, c)$ in $E_1 \cup E_2$.

### 4.2.2. Star vertex

A star vertex is a vertex $u \in V$ of degree at least three in $E_1 \cup E_2$ (Figure 3). We consider:
$Y_{ua,ub}$: the probability the star vertex $u$ infects neighbors $a$ and $b$, conditioned on $u$ being the source vertex.

**Claim 4.** *For u and a, b and c as in Figure 3:*

$$Y_{ua,ub} = \Pr(u \to a, u \to b \mid u \in I_0) = \frac{p_{ua}p_{ub} + q_{ua}q_{ub}}{2}.$$

We emphasize that the conditioning is once again crucial to obtain such a simple form for $Y_{ua,ub}$. Further, we make a key observation that for $i \neq j \in \{a, b, c\}$,

$$Y_{ui,uj} - X_{ui}X_{uj} = \frac{(p_{ui} - q_{ui})(p_{uj} - q_{uj})}{4}. \quad (1)$$

This directly leads to the closed-form expressions for the weights of the edges adjacent to the star vertex $u$.

**Lemma 1.** *Suppose Conditions 1 and 2 are true and $\alpha = 1/2$. Let $s_{ua} \in \{-1, 1\}$. The weight of any edge $(u, a)$ connected to a star vertex u, with distinct neighbors a, b and c in $E_1 \cup E_2$, is given by:*

$$p_{ua} = X_{ua} + s_{ua}\sqrt{\frac{(Y_{ua,ub} - X_{ua}X_{ub})(Y_{ua,uc} - X_{ua}X_{uc})}{Y_{ub,uc} - X_{ub}X_{uc}}},$$

$$q_{ua} = X_{ua} - s_{ua}\sqrt{\frac{(Y_{ua,ub} - X_{ua}X_{ub})(Y_{ua,uc} - X_{ua}X_{uc})}{Y_{ub,uc} - X_{ub}X_{uc}}}.$$

*Furthermore, any two signs $s_{ui}$ and $s_{uj}$, for $i \neq j$ and $i, j \in \{a, b, c\}$, satisfy $s_{ui}s_{uj} = \mathsf{sgn}(Y_{ui,uj} - X_{ui}X_{uj})$.*

**Resolving Mixture Ambiguity**: Separating the weights of both graphs in the mixture is not enough to learn the mixture: we also have to assign the two weights to the right component of the mixture. The identity $s_{ui}s_{uj} = \mathsf{sgn}(Y_{ui,uj} - X_{ui}X_{uj})$ allows us to identify three weights belonging to the same mixture component: $p_{ua}, p_{ub}$ and $p_{uc}$. If one of these weights had been learned before, it is immediate to assign the two new weights to the same component. This leads to the following algorithm:

**LearnStar:** This algorithm takes as input a star vertex $u$, the set of edges of $E_1 \cup E_2$, and all the $X_{ui}$ and $Y_{ui,uj}$ for all $(i, j)$ distinct neighbors of $u$, and returns all the weights of the edges connected to $u$ in both mixtures using the above closed-form expressions (see Supplementary Materials).

### 4.2.3. Line vertex

We now consider a node $u$ that has degree exactly two in $E_1 \cup E_2$ and forms a line structure. Specifically, let $u, a, b$ and $c$ be four distinct nodes of V, such that $(a, u)$, $(u, b)$ and $(b, c)$ belong in $E_1 \cup E_2$. We call such a node $u$ a line vertex (see Figure 4).

To recover the weights of the edges adjacent to a line vertex, only considering events in the first two timesteps is insufficient. Contrary to a star vertex, for a line vertex we have only one second moment.

We circumvent the problem by considering:
1) $Y_{ub,bc}$: the probability of the event when (in Figure 4) $u$ infects only $b$, and in turn $b$ infects $c$, conditioned on $u$ being the source.
2) $Z_{ua,ub,bc}$: the probability of the event when (in Figure 4) $u$ infects both $a$ and $b$, and in turn $b$ infects $c$, conditioned on $u$ being the source.

**Claim 5.** *For a line vertex u and nodes a, b and c as in Figure 4:*

- $Y^{|}_{ua,ub} = Pr(u \to a, u \to b \mid u \in I_0) = \frac{p_{ua}p_{ub}+q_{ua}q_{ub}}{2}$,
- $Y^{|}_{ub,bc} = \Pr(u \to b, b \to c \mid u \in I_0) = \frac{p_{ub}p_{bc}+q_{ub}q_{bc}}{2}$,
- $Z^{|}_{ua,ub,bc} = \Pr(u \to a, u \to b, b \to c \mid u \in I_0)$
$= \frac{p_{ua}p_{ub}p_{bc}+q_{ua}q_{ub}q_{bc}}{2}$.

The result for $Y^{|}_{ua,ub}$ is similar to Claim 4.. However, the proof for $Y^{|}_{ub,bc}$ and $Z^{|}_{ub,bc,bc}$ not only requires $u$ to be the source, but also relies on the fact that $u$ is of degree 2, which implies $p_{uc} = q_{uc} = 0$.

We note that $Y^{|}_{bc,ua}$ does not exist, as $c$ cannot be infected if $b$ is not. So we cannot immediately replicate the star vertex proof. Let us define $R^{|} := X_{ua}X_{bc} + \frac{Z^{|}_{ua,ub,bc} - X_{ua}Y^{|}_{ub,bc} - X_{bc}Y^{|}_{ua,ub}}{X_{ub}}$. However, we prove the following equality, which acts as a surrogate for $(Y^{|}_{ua,bc} - X_{ua}X_{bc})$:

$$R^{|} = \frac{1}{4}(p_{ua} - q_{ua})(p_{bc} - q_{bc}). \quad (2)$$

As in Lemma 1, we now obtain the closed-form expressions for the weights associated with the line vertex $u$. For the sake of notation consistency, we define $Y^{|}_{bc,ua} := (R^{|} + X_{bc}X_{ua})$ (it has no probabilistic interpretation).

**Lemma 2.** *Suppose Conditions 1 and 2 hold and $\alpha = 1/2$, then for $s_{ua}, s_{ub}$, and $s_{bc}$ in $\{-1, 1\}$, the weights of the* *edges for a line structure are given by:*

$$\forall (e1, e2, e3) \in \{(ua, ub, bc), (ub, bc, ua), (bc, ua, ub)\},$$

$$p_{e1} = X_{e1} + s_{e1}\sqrt{\frac{(Y^{|}_{e1,e2} - X_{e1}X_{e2})(Y^{|}_{e3,e1} - X_{e3}X_{e1})}{Y^{|}_{e2,e3} - X_{e2}X_{e3}}},$$

$$q_{e1} = X_{e1} - s_{e1}\sqrt{\frac{(Y^{|}_{e1,e2} - X_{e1}X_{e2})(Y^{|}_{e3,e1} - X_{e3}X_{e1})}{Y^{|}_{e2,e3} - X_{e2}X_{e3}}}.$$

*Furthermore, for all* $e1, e2 \in \{ua, ub, uc\}$ *and* $e1 \neq e2$, $s_{e1}s_{e2} = \mathsf{sgn}(Y_{e1,e2} - X_{e1}X_{e2})$.

**LEARNLINE:** In the same fashion as for the star vertex, we can use the expression in Lemma 2 to design an algorithm LEARNLINE, which takes as input a line vertex $u$, the set of the edges of $E_1 \cup E_2$, and the limit of the estimators $X_{ua}$, $X_{ub}$, $X_{bc}$, $Y^{|}_{ua,ub}$, $Y^{|}_{ub,bc}$, $Z^{|}_{ua,ub,bc}$ for $a$, $b$ and $c$ as in Figure 4, and returns the weights of the edges $(u, a)$, $(u, b)$ and $(b, c)$ in both mixtures (see Supplementary Materials).

**LEARN2NODES** Our main algorithm is initialized by learning weights associated with edges connected to two nodes using subroutine LEARN2NODES. As this algorithm is very similar in spirit to our general algorithm, we leave the details to the Supplementary Materials.

### 4.3. Correctness of Algorithm 1

To prove the correctness of the main algorithm, we show the following invariant:

**Lemma 3.** *At any point in the algorithm, the entire neighborhood of any node of* $S$ *has been learned and recorded in* $W$*:*

$$\forall u \in S,\ \forall v \in V,\ (u, v) \in E \implies (u, v) \in W.$$

*Proof.* We prove the above by induction on the iteration of the while loop. Due to the correctness of LEARN2NODES (proven in Supplementary material), after calling this function, $W$ contains all edges adjacent to the two vertices in $S$. Hence the base case is true. Let us assume that after $k$ iterations of the loop, the induction hypothesis holds.

We consider three cases in the $(k + 1)$-th iteration:
- $\mathbf{deg(v) \geq 3}$: We recover all edges adjacent to the star vertex $v$ by using LEARNSTAR (correct due to Lemma 1). Sign consistency is ensured using edge $(u, v) \in W$ since $u \in S$.
- $\mathbf{deg(v) = 2}$: There exists $w \in S$ such that $(u, w) \in E$ since $|S| \geq 2$ and is connected. Since $deg(v) = 2$, there exists $t \neq u$ such that $(t, v) \in E$. Now if $t \in S$ then $(t, v) \in W$ and we are done. If $t \notin S$ then $v$ is a line vertex for $t - v - u - w$. By using LEARNLINE we recover all edges on the line (correct due to Lemma 2). Sign consistency is ensured through edge $(v, u)$.
- $\mathbf{deg(v) = 1}$: Since $u \in S$, we have $(u, v) \in W$, so we are done.

Thus by induction, after every iteration of the for loop, the invariant is maintained. □

**Theorem 4.** *Suppose Conditions 1 and 2 are true; Algorithm 1 learns the edge weights of the two balanced mixtures in the setting of infinite samples.*

*Proof.* Since at every iteration, the size of $S$ increases by 1, after at most $|V|$ iterations, we have $S = V$. Using Lemma 3, we also have $W = E_1 \cup E_2$. □

### 4.4. Finite Sample Complexity

In this section, we investigate the error in estimating the quantities $X_{ui}$, $Y_{ui,uj}$ for $i, j \in \{a, b, c\}$ in the case of a star vertex, and $X_{e1}$, $Y^{|}_{e1,e2}$ and $Z_{ua,ub,bc}$ for $e1, e2 \in \{ua, ub, bc\}$ in the case of a line vertex, using a finite number of cascades. We further investigate the effect of the error in these quantities on the accuracy of the recovered weights.

We use a simple count-based estimator. Specifically, for events $\mathcal{E}_1$ and $\mathcal{E}_2$, we estimate the probability $\Pr(\mathcal{E}_1|\mathcal{E}_2) = \frac{\sum_{m=1}^{M} \mathbb{1}_{\mathcal{E}_1 \cap \mathcal{E}_2}}{\sum_{m=1}^{M} \mathbb{1}_{\mathcal{E}_2}}$. As a concrete example, we have the estimator for $X_{ua}$ as $\hat{X}_{ua} := \frac{\sum_{m=1}^{M} \mathbb{1}_{u \to a, u \in I_0^m}}{\sum_{m=1}^{M} \mathbb{1}_{u \in I_0^m}}$. Here $I_0^m$ denotes the source of the $m$-th cascade and $u \to a$ denotes that $u$ infects $a$. We can argue, using the law of large numbers and Slutsky's Lemma, that the above approach provides us with balanced estimators.

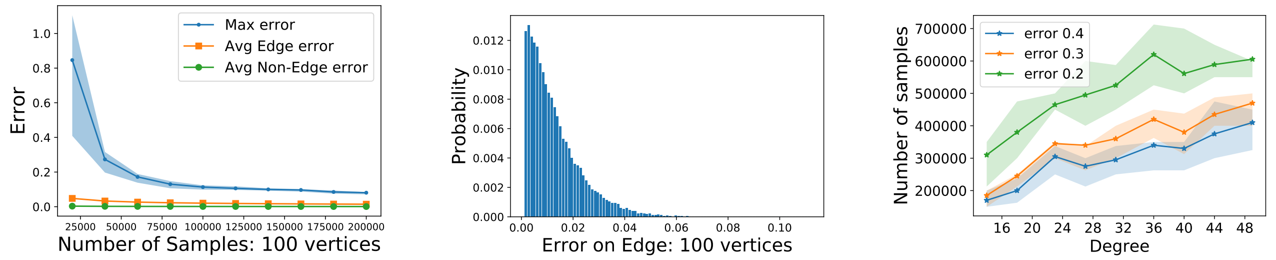
We first establish high probability error bounds for the base estimators with a finite number of cascades, for both the star vertex and the line vertex. Finally, using the above guarantees, we provide our main sample complexity result for the balanced mixture problem. See Supplementary Material for proofs.

**Theorem 5.** *Suppose Condition 1 and 2 hold. With* $M = \mathcal{O}\left(\frac{1}{p_{min}^6 \Delta^4} \frac{N}{\epsilon^2} \log\left(\frac{N}{\delta}\right)\right)$ *samples, Algorithm 1 learns the edge weights of a balanced mixture on two graphs within precision* $\epsilon$ *with probability at least* $1 - \delta$.

## 5. Extensions

### 5.1. Extension to Directed Graphs

We notice that in the case of directed graphs of minimum out-degree three, we can simply use the star structure to learn all the directed edges. This, however, would not be enough to ensure mixture consistency; we therefore need to also use two new structures to solve the problem. The algorithm is very similar to Algorithm 1, and the structures



(a) Maximum and average absolute error as a function of the number of cascades on $\mathcal{G}(N,p)$ graphs, with $N = 100$ vertices, $p = \sqrt{N}$. The first graph has 1044 edges, the second one 1026.

(b) Normalized histogram of absolute error after 200000 cascades on the same $\mathcal{G}(N,p)$ graphs, with $N = 100$ vertices, $p = \sqrt{N}$, and 100 bins. The first graph has 1044 edges, the second one 1026.

(c) Number of cascades needed to reach given precision, as a function of the degree. All experiments are on random $d$-regular graphs on 50 nodes, with 375 to 1225 undirected edges.

are very similar to the structures encountered so far. Precise details are left for the Supplementary Material.

### 5.2. Extension to Unbalanced/Unknown Priors

While previous results only considered balanced mixtures, $i.e.$ with parameter $\alpha = 0.5$, we focus here on unbalanced mixtures ($\alpha \neq 0.5$ known) and on mixtures of unknown priors ($\alpha$ unknown).

We first note that the main algorithm for recovering the graph does not depend on the prior once the correct LEARNSTAR and LEARNLINE primitives are provided. Therefore, we show how to design these primitives.

**Unbalanced Mixtures** We can easily extend Equation 1 for star vertices in the case of unbalanced mixtures. Specifically, we have for all $i \neq j \in \{a, b, c\}$:

$$Y_{ui,uj} - X_{ui}X_{uj} = \alpha(1-\alpha)(p_{ui} - q_{ui})(p_{uj} - q_{uj}).$$

However, Equation 2 does not extend easily (see Supplementary Material for details) in the general case:

**Theorem 6.** *Suppose Conditions 1 and 2 are true. Then there exists an algorithm that runs on epidemic cascades over an* ***unbalanced*** *mixture of two undirected, weighted graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, with $|V| = N$, and recovers the edge weights corresponding to each graph up to precision $\epsilon$ in time $O(N^2)$ and sample complexity:*

- $O\left(\frac{N \log N}{\epsilon^2}\text{poly}(\frac{1}{\Delta})\text{poly}(\frac{1}{\min(\alpha,1-\alpha)})\right)$ *in general.*
- $O\left(\frac{N \log N}{\epsilon^2\Delta^2}\text{poly}(\frac{1}{\min(\alpha,1-\alpha)})\right)$ *for graphs of minimum degree three.*

**Mixtures of Unknown Priors** If the graph has at least one star vertex, we can learn the entire mixture by learning the parameter $\alpha$ from this node, and use the results from above to learn the rest of the graph once $\alpha$ has been recovered. Details can be found in Supplementary Material.

### 5.3. Extension to Mixtures of $K > 2$ Graphs

For graphs of minimum degree $2K-1$, writing the equations using first and second moments ($i.e.$ the $X_{ua}$ and $Y_{ua,ub}$) as above yields at least as many equations as unknowns. Using Quadratic Constraints Quadratic Programming, we can obtain numerical solutions. Note that the constraints are not convex, so there is no guarantee this problem is solvable in polynomial time. As there is also no immediate reduction to a NP-hard problem, we do not know the complexity of learning mixtures of $K > 2$ graphs.

## 6. Experiments

We validate our results on synthetic data. We first draw random graphs from a distribution (specified below), and each sample is a simulation of a cascade spreading on it. Once the graphs are drawn, the experiments are run 10 times. The shaded area represents the $25^{th}$ to $75^{th}$ percentiles.

Our first two experiments are on Erdös-Renyi $\mathcal{G}(N,p)$ graphs. In Figure 5a, we investigate the maximum error on learned edges and compare it with the average error. We find that the "Max error" curve follows the dependence predicted by our theory, of $\epsilon = \mathcal{O}\left(1/\sqrt{N\log(N)}\right)$. It is also worth noting that the average error on the existing edges is one order of magnitude smaller than the max error. By plotting the normalized histogram of the absolute value of the errors (Figure 5b), we confirm that only a few edges keep the maximum error high. Finally, by varying the degree on random $d$-regular graphs with a fixed number of vertices (Figure 5c), we see that the sample complexity is multiplied by 2 as the number of edges grows from $\Theta(N)$ to $\Theta(N^2)$, as predicted since the dependence in the degree is logarithmic. Therefore, our algorithm is as sample-efficient (up to small constants) on dense graphs as on sparse graphs.

## 7. Conclusion

We provided an efficient algorithm for learning the edge weights of a balanced mixture of two undirected graphs from epidemic cascades, as well as matching lower bounds (up to log factors). We extended our results to directed graphs of min-degree three, and unbalanced/unknown mixtures.

Our algorithm is robust, in the sense that it has partial recovery guarantees, and it is unaffected by adversarial examples which would consist of adding nodes/edges. Due to its local structure, it is also easily parallelizable.

Learning mixtures of more than two graphs, or mixtures of directed graphs without restriction on the minimum degree, are still open problems, and are left for future work.

## 8. Acknoledgements

The authors would like to thank the anonymous reviewers for helpful comments, and Brooke Elizabeth Frank for a fertile workspace.

The work is partially supported by the National Science Foundation under Grants No.: 1302435, No.: 1609279 and No.: 1704778.

## References

Abrahao, B., Chierichetti, F., Kleinberg, R., and Panconesi, A. Trace complexity of network inference. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13*, pp. 491, 2013. ISSN 9781450321747. doi: 10.1145/2487575.2487664. URL http://dl.acm.org/citation.cfm?doid=2487575.2487664.

Arias-Castro, E. and Nov, S. T. Detecting a Path of Correlations in a Network. pp. 1–12.

Arias-Castro, E., Candès, E. J., and Durand, A. Detection of an anomalous cluster in a network. *The Annals of Statistics*, 39(1):278–304, 2011. doi: 10.1214/10-AOS839.

Balakrishnan, S., Wainwright, M. J., Yu, B., et al. Statistical guarantees for the em algorithm: From population to sample-based analysis. *The Annals of Statistics*, 45(1): 77–120, 2017.

Chen, Y., Yi, X., and Caramanis, C. Convex and nonconvex formulations for mixed regression with two components: Minimax optimal rates. *IEEE Transactions on Information Theory*, 64(3):1738–1766, 2017.

Cheng, J., Adamic, L. A., Dow, P. A., Kleinberg, J., and Leskovec, J. Can Cascades be Predicted? In *Proceedings of the 23rd international conference on World wide web (WWW' 14)*, 2014. ISBN 9781450327442. doi: 10.1145/2566486.2567997.

Daneshmand, H., Gomez-Rodriguez, M., Song, L., and Schoelkopf, B. Estimating Diffusion Network Structures: Recovery Conditions, Sample Complexity & Soft-thresholding Algorithm. In *Proceedings of the 31 st International Conference on Machine Learning, Beijing, China, 2014 - ICML 14*, 2014. ISBN 9781634393973. doi: 10.1002/aur.1474.Replication. URL http://arxiv.org/abs/1405.2936.

Daskalakis, C., Tzamos, C., and Zampetakis, M. Ten steps of em suffice for mixtures of two gaussians. In *30th Annual Conference on Learning Theory*, 2017.

Del Vicario, M., Bessi, A., Zollo, F., Petroni, F., Scala, A., Caldarelli, G., Stanley, H. E., and Quattrociocchi, W. The spreading of misinformation online. *Proceedings of the National Academy of Sciences*, pp. 201517441, 2016. ISSN 0027-8424. doi: 10.1073/pnas.1517441113. URL http://www.pnas.org/content/early/2016/01/02/1517441113.abstract.

Diakonikolas, I., Kane, D. M., and Stewart, A. List-decodable robust mean estimation and learning mixtures of spherical gaussians. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1047–1060. ACM, 2018.

Dong, M., Zheng, B., Quoc Viet Hung, N., Su, H., and Li, G. Multiple rumor source detection with graph convolutional networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 569–578, 2019.

Drakopoulos, K., Ozdaglar, A., and Tsitsiklis, J. N. An efficient curing policy for epidemics on graphs. *arXiv preprint arXiv:1407.2241*, (December):1–10, 2014. doi: 10.1109/TNSE.2015.2393291. URL http://arxiv.org/abs/1407.2241.

Drakopoulos, K., Ozdaglar, A., and Tsitsiklis, J. N. A lower bound on the performance of dynamic curing policies for epidemics on graphs. (978):3560–3567, 2015. ISSN 07431546. doi: 10.1109/CDC.2015.7402770.

Fanti, G., Kairouz, P., Oh, S., and Viswanath, P. Spy vs. Spy: Rumor Source Obfuscation. *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS' 14)*, pp. 271–284, 2015. ISSN 01635999. doi: 10.1145/2745844.2745866. URL http://arxiv.org/abs/1412.8439.

Fanti, G., Kairouz, P., Oh, S., Ramchandran, K., and Viswanath, P. Rumor source obfuscation on irregular trees. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science (SIGMETRICS' 16 )*, pp. 153–164. ACM, 2016.

Fanti, G., Kairouz, P., Oh, S., Ramchandran, K., and Viswanath, P. Hiding the Rumor Source. *IEEE Transactions on Information Theory*, 63(10):6679–6713, 2017. ISSN 00189448. doi: 10.1109/TIT.2017.2696960.

Farajtabar, M., Yang, J., Ye, X., Xu, H., Trivedi, R., Khalil, E., Li, S., Song, L., and Zha, H. Fake News Mitigation via Point Process Based Intervention. In *Proceedings of the 34th International Conference on Machine Learning (ICML' 17)*, 2017. URL `http://arxiv.org/abs/1703.07823`.

Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Information Retrieval*, 4(2):133–151, 2001. ISSN 13864564. doi: 10.1023/A:1011419012209.

Gomez-Rodriguez, M., Leskovec, J., and Schölkopf, B. Structure and Dynamics of Information Pathways in Online Media. In *6th International Conference on Web Search and Data Mining (WSDM 2013)*, 2013. ISBN 9781450318693.

Hoffmann, J. and Caramanis, C. The Cost of Uncertainty in Curing Epidemics. *Proceedings of the ACM on Measurement and Analysis of Computing Systems (SIGMETRICS' 18)*, 2(2):11–13, 2018. doi: 10.1145/3219617.3219622. URL `http://doi.acm.org/10.1145/3219617.3219622`.

Hoffmann, J. and Caramanis, C. Learning graphs from noisy epidemic cascades. *arXiv preprint arXiv:1903.02650*, 2019.

Kempe, D., Kleinberg, J., and Tardos, É. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '03*, 2003. ISBN 1581137370. doi: 10.1145/956755.956769.

Khim, J. and Loh, P.-L. Permutation Tests for Infection Graphs. pp. 1–28, 2017. URL `http://arxiv.org/abs/1705.07997`.

Khim, J. and Loh, P.-L. A theory of maximum likelihood for weighted infection graphs. pp. 1–47, 2018. URL `https://arxiv.org/pdf/1806.05273.pdf`.

Kolli, N. and Narayanaswamy, B. Influence maximization from cascade information traces in complex networks in the absence of network structure. *IEEE Transactions on Computational Social Systems*, 6(6):1147–1155, 2019.

Kwon, J. and Caramanis, C. Em converges for a mixture of many linear regressions. *arXiv preprint arXiv:1905.12106*, 2019.

Kwon, J., Qian, W., Caramanis, C., Chen, Y., and Davis, D. Global convergence of the em algorithm for mixtures of two component linear regression. In *32nd Annual Conference on Learning Theory*, pp. 2055–2110. PMLR, 2019. URL `http://proceedings.mlr.press/v99/kwon19a.html`.

Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N. Cost-effective Outbreak Detection in Networks. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '07)*, pp. 420, 2007. doi: 10.1145/1281192.1281239. URL `http://eprints.pascal-network.org/archive/00005342/http://dl.acm.org/citation.cfm?id=1281239`.

Liu, S., Shen, H., Zheng, H., Cheng, X., and Liao, X. Ct lis: Learning influences and susceptibilities through temporal behaviors. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(6):1–21, 2019.

Meirom, E. A., Milling, C., Caramanis, C., Mannor, S., Orda, A., and Shakkottai, S. Localized epidemic detection in networks with overwhelming noise. pp. 1–27, 2014. URL `http://arxiv.org/abs/1402.1263`.

Milling, C., Caramanis, C., Mannor, S., and Shakkottai, S. Network Forensics : Random Infection vs Spreading Epidemic. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems (SIGMETRICS' 12)*, 2012. ISBN 9781450310970.

Milling, C., Caramanis, C., Mannor, S., and Shakkottai, S. Local detection of infections in heterogeneous networks. *Proceedings - IEEE INFOCOM*, 26:1517–1525, 2015. ISSN 0743166X. doi: 10.1109/INFOCOM.2015.7218530.

Netrapalli, P. and Sanghavi, S. Learning the Graph of Epidemic Cascades. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems (SIGMETRICS' 12)*, pp. 211–222, 2012. ISBN 9781450310970. doi: 10.1145/2318857.2254783. URL `http://arxiv.org/abs/1202.1779`.

Newman, M. E. J. *Networks: An Introduction*, volume 23. 2014. ISBN 9780199206650. doi: 10.1017/S0963180113000479. URL `http://www.journals.cambridge.org/abstract{_}S0963180113000479`.

Ou, H.-C., Sinha, A., Suen, S.-C., Perrault, A., and Tambe, M. Who and when to screen: Multi-round active screening for recurrent infectious diseases under uncertainty, 2019.

Pasdeloup, B., Gripon, V., Mercier, G., Pastor, D., and Rabbat, M. G. Characterization and inference of graph diffusion processes from observations of stationary signals. *IEEE transactions on Signal and Information Processing over Networks*, 4(3):481–496, 2017.

Prokhorenkova, L., Tikhonov, A., and Litvak, N. Learning clusters through information diffusion. In *The World Wide Web Conference*, pp. 3151–3157, 2019.

Shah, D. and Zaman, T. Rumors in a Network : Who ' s the Culprit ? *IEEE Transactions on information theory*, 57 (8):1–43, 2010a. doi: 10.1109/TIT.2011.2158885. URL http://arxiv.org/abs/0909.4370.

Shah, D. and Zaman, T. Detecting sources of computer viruses in networks: theory and experiment. In *ACM SIGMETRICS Performance Evaluation Review*, volume 38, pp. 203–214. ACM, 2010b.

Shah, D. and Zaman, T. Rumor centrality: a universal source detector. In *ACM SIGMETRICS Performance Evaluation Review*, volume 40, pp. 199–210. ACM, 2012.

Spencer, S. and Srikant, R. On the impossibility of localizing multiple rumor sources in a line graph. *ACM SIGMETRICS Performance Evaluation Review*, 43(2): 66–68, 2015.

Sridhar, A. and Poor, H. V. Sequential estimation of network cascades. *arXiv preprint arXiv:1912.03800*, 2019.

Trouleau, W., Etesami, J., Grossglauser, M., Kiyavash, N., and Thiran, P. Learning hawkes processes under synchronization noise. In *International Conference on Machine Learning*, pp. 6325–6334, 2019.

Wang, S., Chen, S., Cheng, X., Lv, W., and Yu, J. Analysis of antagonistic dynamics for rumor propagation. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1253–1263. IEEE, 2019.

Wang, Z., Dong, W., Zhang, W., and Tan, C. W. Rumor source detection with multiple observations: Fundamental limits and algorithms. In *ACM SIGMETRICS Performance Evaluation Review*, volume 42, pp. 1–13. ACM, 2014.

Wu, L. and Liu, H. Tracing Fake-News Footprints: Characterizing Social Media Messages by How They Propagate. In *(WSDM 2018) The 11th ACM International Conference on Web Search and Data Mining*, 2018. ISBN 9781450355810. doi: 10.1145/3159652.3159677. URL http://www.public.asu.edu/{~}liangwu1/WSDM18{_}TraceMiner.pdf.

Xie, Y., Jiang, H., Liu, F., Zhao, T., and Zha, H. Meta learning with relational information for short sequences. In *Advances in Neural Information Processing Systems*, pp. 9901–9912, 2019.

Xu, J., Hsu, D. J., and Maleki, A. Global analysis of expectation maximization for mixtures of two gaussians. In *Advances in Neural Information Processing Systems*, pp. 2676–2684, 2016.

Yan, W., Loh, P.-L., Li, C., Huang, Y., and Yang, L. Conquering the worst case of infections in networks. *IEEE Access*, 2019.

Yi, X., Caramanis, C., and Sanghavi, S. Alternating minimization for mixed linear regression. In *International Conference on Machine Learning*, pp. 613–621, 2014.

Yi, X., Caramanis, C., and Sanghavi, S. Solving a mixture of many random linear equations by tensor decomposition and alternating minimization. *arXiv preprint arXiv:1608.05749*, 2016.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., and Leskovec, J. SEISMIC: A Self-Exciting Point Process Model for Predicting Tweet Popularity. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15 )*, 2015. doi: 10.1145/2783258.2783401. URL http://arxiv.org/abs/1506.02594http://dx.doi.org/10.1145/2783258.2783401.