Learning Graphs from Noisy Epidemic Cascades

JESSICA HOFFMANN, The University of Texas at Austin
CONSTANTINE CARAMANIS, The University of Texas at Austin

We consider the problem of learning the weighted edges of a graph by observing the noisy times of infection for multiple epidemic cascades on this graph. Past work has considered this problem when the cascade information, i.e., infection times, are known exactly. Though the noisy setting is well motivated by many epidemic processes (e.g., most human epidemics), to the best of our knowledge, very little is known about when it is solvable. Previous work on the no-noise setting critically uses the ordering information. If noise can reverse this – a node's reported (noisy) infection time comes after the reported infection time of some node it infected – then we are unable to see how previous results can be extended. We therefore tackle two versions of the noisy setting: the *limited-noise* setting, where we know noisy times of infections, and the *extreme-noise* setting, in which we only know whether or not a node was infected. We provide a polynomial time algorithm for recovering the structure of bidirectional trees in the extreme-noise setting, and show our algorithm matches lower bounds established in the no-noise setting, and hence is optimal. We extend our results for general degree-bounded graphs, where again we show that our (poly-time) algorithm can recover the structure of the graph with optimal sample complexity. We also provide the first efficient algorithm to learn the weights of the bidirectional tree in the limited-noise setting. Finally, we give a polynomial time algorithm for learning the weights of general bounded-degree graphs in the limited-noise setting. This algorithm extends to general graphs (at the price of exponential running time), proving the problem is solvable in the general case. All our algorithms work for any noise distribution, without any restriction on the variance.

CCS Concepts: • **Mathematics of computing → Graph theory**; **Graph algorithms**; **Probability and statistics**; **Computing most probable explanation**; *Combinatorics*; • **Networks → Network algorithms**; • **Theory of computation → Graph algorithms analysis**; *Sample complexity and generalization bounds*; Random walks and Markov chains; Social networks.

Additional Key Words and Phrases: Epidemics; SIR model; Contact process on graph; Noisy inverse problem; Graph reconstruction; Robust graph algorithms.

ACM Reference Format:
Jessica Hoffmann and Constantine Caramanis. 2019. Learning Graphs from Noisy Epidemic Cascades. *Proc. ACM Meas. Anal. Comput. Syst.* 3, 2, Article 40 (June 2019), 34 pages. https://doi.org/10.1145/3326155

1 INTRODUCTION

Epidemic models accurately represent (among other processes) the spread of diseases, information (rumors, viral videos, news stories, etc.), the spread of malevolent agents in a network (computer viruses, malicious apps, etc.), or even biological processes (pathways in cell signaling networks, chains of activation in the gene regulatory network, etc.). We focus on epidemics that spread on an underlying graph [31], as opposed to the fully mixed models introduced in the early literature [4].

Authors' addresses: Jessica Hoffmann, The University of Texas at Austin, Austin, TX, 78712, hoffmann@cs.utexas.edu; Constantine Caramanis, The University of Texas at Austin, Austin, TX, 78712, constantine@utexas.edu.

2476-1249/2019/6-ART40 $15.00
https://doi.org/10.1145/3326155

Most settings assume we know the underlying graph and aim to study properties of the spread. Much work has been done in detection [2, 3, 21, 25–28], where the goal is to decide whether or not there is indeed an infection. This problem is of importance in deciding whether or not a computer network is under attack, for instance, or whether a product gets sold through word-of-mouth or thanks to the advertisement campaign (or both [29]). More specifically, the problem of source detection [32–36] or obfuscation [11–13] has been extensively studied. On the other side of the spectrum, both experimental and theoretical work has tackled the problem of modeling [7, 17, 37], predicting the growth [6, 39], and controlling the spread of epidemics [9, 10, 14, 18].

In this work, we take the opposite approach: assuming we know some properties of the spread, can we recover the underlying graph? The early works on this subject proposed a few heuristics and experimentally proved their effectiveness [16, 19]. Netrapalli et al. [30] established the first theoretical guarantees for this problem for discrete-time infections. They proved one can recover the edges of any graph with correlation decay, with access to the times of infection for multiple cascades spreading on the graph. They introduced a likelihood, proved it decouples into convex subproblems, and demonstrated that the edges of the graph can therefore be obtained efficiently. They also proved a sample complexity lower bound and showed their method is within a log factor of it. Abrahao et al. [1] also introduced a method of solving this problem, this time for a more realistic, continuous-time infection model, through learning only the first edge of each cascade. Zarezade et al. [38] proposed a first experimental attempt to tackle the case of correlated cascades using Hawkes processes. Khim et al. [22] extended the theoretical results to the case where the the cascades spreading on the graph are not independent, which required completely new machinery involving martingales and weighted Pólya urns.

All the results above assume we have perfect knowledge of the properties of the spread we use to reconstruct the graph. For most of the literature, those are the times of infection for all nodes for each cascade. This assumption may hold for online epidemics, as information is usually dated (for instance, posts or retweets on social networks have time stamps). For human networks, however, this assumption is often unrealistic: official diagnosis (and hence recording by any tracking entity such as the CDC) may come days, weeks, or in important examples such as HIV, *years* after the actual moment of infection. Moreover, this can be highly variable from person to person, hence the infector is often diagnosed after the infectee. Similar issues arise with biological networks: we only know the expression of a gene when a measure is taken, which can happen after a typically arbitrary delay.

We therefore develop a method for learning the graph of epidemics with noisy times of infection. We demonstrate that past approaches are unusable, due to the fact that even small levels of noise are typically enough to cause order-of-diagnosis to differ from order-of-infection. We develop new techniques to deal with this setting, and prove our algorithm can reliably learn the edges of a tree in the limited-noise setting, for *any* noise distribution. We also show we can learn the structure of any bounded degree graph from a very weak observation model, in a sample-optimal fashion. We finally provide an algorithm which learns the weights of any bounded-degree graph in the limited-noise setting.

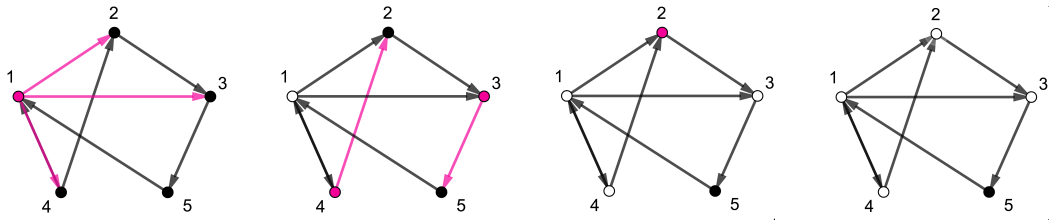
1.1 Model

We observe epidemics spreading on a graph, and aim to reconstruct the graph structure from noisy estimates of the times of infection. In this section, we specify the exact propagation model, the noisy observation model, and the two learning tasks this work tackles.

Propagation model: We consider a particular variant of the *independent cascade model*, close to the one-step model introduced by [15] and further studied by [20]. The epidemic spreads in *discrete time* on a *weighted directed graph* $G = (V, E)$, where parents can infect their children with probability

Table 1. Notations

$G = (V, E)$	Graph G, V set of nodes, E set of edges.
N	Number of nodes in the graph.
T_i^m	Random variable for the actual time of infection of node i during cascade m.
n_i^m	Noise of node i during cascade m.
$T_i'^m = T_i^m + n_i^m$	Random variable for the noisy time of infection of node i during cascade m.
I_i^m	Random boolean variable for infection status of the nodes during cascade m: $I_i'^m = True \Leftrightarrow$ node i was infected during cascade m.
p_{ij}	Weight of edge (i, j), corresponding to the probability that i infects j. If $(i, j) \in E$, $0 < p_{min} \leq p_{ij} \leq p_{max} < 1$.



(a) At t=0, node 1 is the source, in the infected state. It can possibly infect node 2, 3 and 4, all in the susceptible state.

(b) At t=1, nodes 3 and 4 are infected. Node 3 can infect node 5, in the susceptible state. Node 4 can infect node 2, in the susceptible state, but not node 1, since node 1 is now removed.

(c) At t=2, node 2 is infected. All its neighbors are in the removed state, so new node can be infected.

(d) At t=3, the cascade stops, even if node 5 remains in the susceptible state.

Fig. 1. A complete cascade.

equal to the weight of the edge between them, but children cannot infect their parents. We allow bidirectional edges: it is possible that both $(i, j) \in E$ and $(j, i) \in E$, possibly with different weights. For each edge $(i, j) \in E$, the corresponding weight p_{ij} is such that $0 < p_{min} \leq p_{ij} \leq p_{max} < 1$.

This process is an instance of a $Susceptible \rightarrow Infected \rightarrow Removed$ (SIR) process. Only Susceptible nodes can become Infected. Infected nodes transition to the Removed state exactly one time step after they have become Infected. Once a node is in the Removed state, it does not interact with the epidemic anymore (the propagation of the epidemic proceeds as though this node were no longer on the graph).

Each node starts out in the Susceptible state. As in [1], each cascade m starts at a positive time[1] on a unique Infected source node, picked uniformly among the nodes of the graph. Once a node becomes infected, it is removed from the graph, and if it has children, each is infected at the next time step independently according to a probability specified by the weight of the edge shared between the infected parent node and its susceptible child. The process ends when there are no

[1]Most of the literature considers the initial time of infection to be 0. This is because when we have access to the exact times of infection, we can make this assumption without loss of generality. In our case, it would imply we know exactly when an outbreak started, which is usually not the case.

newly infected nodes (either because no infection happened during the previous time step, in which case some nodes may never be infected, or because all the nodes of the graph are removed). One realization of this process from start to finish is called a **cascade**. The number of Removed nodes at the end of a cascade is called the **size of this cascade**. If two nodes are infected during the same cascade, we say that they are **co-infected** for this cascade. This process is illustrated in Figure 1.

Observation model: Let T_i^m be a random variable corresponding to the time of infection of node i during cascade m, and let t_i^m be its realization (if i stays in the susceptible state during cascade m, we have $t_i^m = \infty$). We introduce three observation models.

In the **no-noise setting**, we have access to the exact times of infection T_i^m.

In the **limited-noise setting**, we never get to observe the exact times of infection T_i^m, but only a noisy version $T_i^{'m} = T_i^m + n_i^m$ (with realization $t_i^{'m}$), where all the n_i^m are i.i.d., and represent the noise added to the T_i^m. We assume n_i^m follows a *known* distribution $\mathcal{D}$. The only restriction we put on $\mathcal{D}$ is that it cannot have infinite value (*i.e.,* $t_i^{'m} = \infty \Leftrightarrow t_i^m = \infty$, and we know for a fact when nodes have been infected or not).

In the **extreme-noise setting**, we take the previous setting to the extreme, and we assume that instead of having access to the noisy times of infection $T_i^{'m}$, we only have access to the infection status of the nodes I_i^m. We know $I_i^m = True$ if i was infected during cascade m, and $I_i^m = False$ otherwise. Note that $T_i^{'m} < \infty \Leftrightarrow I_i^m = True$, so we can always deduce the infection status from the noisy times of infection. However, we cannot guess the noisy times of infection from the infection status: the (noisy) times of infections contain strictly more information than the infection status.

For these three settings, we call a **sample** the vector of all observations for the cascade m. In the no-noise setting, this is the extended-integer vector $\{t_i^m\}_{i\in V}$. In the limited-noise setting, this is the extended-integer vector $\{t_i^{'m}\}_{i\in V}$. In the extreme-noise setting, this is the boolean vector corresponding to the realization of $\{I_i^m\}_{i\in V}$. We also use the notation $T' = \{T_i^{'m}\}_{i\in V}^{m=1\dots M}$ (respectively $t' = \{t_i^{'m}\}_{i\in V}^{m=1\dots M}$) for the matrix representing the random variable (respectively the realizations) of all the samples.

Learning tasks: We focus on two different learning tasks. When we learn the **structure** of a graph, it means that for any two nodes i and j, we can decide whether or not there exists an edge between these two nodes (whatever its direction). When we learn the **weights** of the graph, it means that for every two nodes i and j, we learn the exact value[2] of both p_{ij} and p_{ji} up to any given precision ϵ.

1.2 Why is it a hard problem?

1.2.1 Counting approaches. Most approaches in the no-noise setting relate to counting. In our setting, for instance, a natural (and consistent) estimator for p_{ij} is to count how often an infection occurred along an edge, and divide it by how often such an infection could have happened:

$$\hat{p_{ij}} = \frac{\text{Number of times } j \text{ becomes infected one time step after } i}{\text{Number of times } i \text{ was infected before } j}.$$

In the no-noise setting, j could only have been infected by a node signaling exactly one time step before j. However, in the limited-noise setting, j signaling its infection one time step after i could stem from a variety of scenarios:

- i could have indeed infected j: cases a), b) and c) of Figure 3 below.
- j could have infected i, but the noise flipped the order of signaling: cases d), e) and f).
- No infection happen between i and j, and the probability of infection depends mainly on another node k: cases g), e) and f). This could happen for *any other node k* in the graph.

[2]When $(i, j) \notin E$, we have $p_{ij} = 0$.

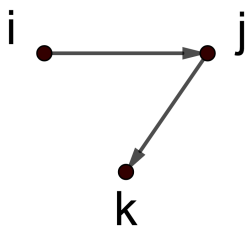


Fig. 2. Possible scenarios which could have led to $T_i^{'} = 2, T_j^{'} = 3$ and $T_k^{'} = 4$. In the no-noise setting, this implies $T_i = 2, T_j = 3, T_k = 4$, and there is only one possible infection pattern.

The natural estimator introduced earlier is therefore not consistent anymore; instead, it tends to a quantity which depends on p_{ij}, but also p_{ji}, and $p_{ik}, p_{ki}, p_{jk}, p_{kj}$ as well, for all the other nodes k in the graph. By counting the number of times j became infected one time step after i, we are not counting the number of infections along the edge (i, j) anymore, but instead a mixture of all the scenarios described above, which not only include the cases where j infected i, but also events in which the cascade spread through another node k, and the edge (i, j) was irrelevant to the process. Using this estimator, or any obvious (to us) extension of it, would not only imply learning the wrong weights for the edges, but also learning edges when there are no edges. Our first contribution is therefore to design a new set of estimators, from which we can deduce the value of p_{ij} (Sections 2.3 and 3.2).

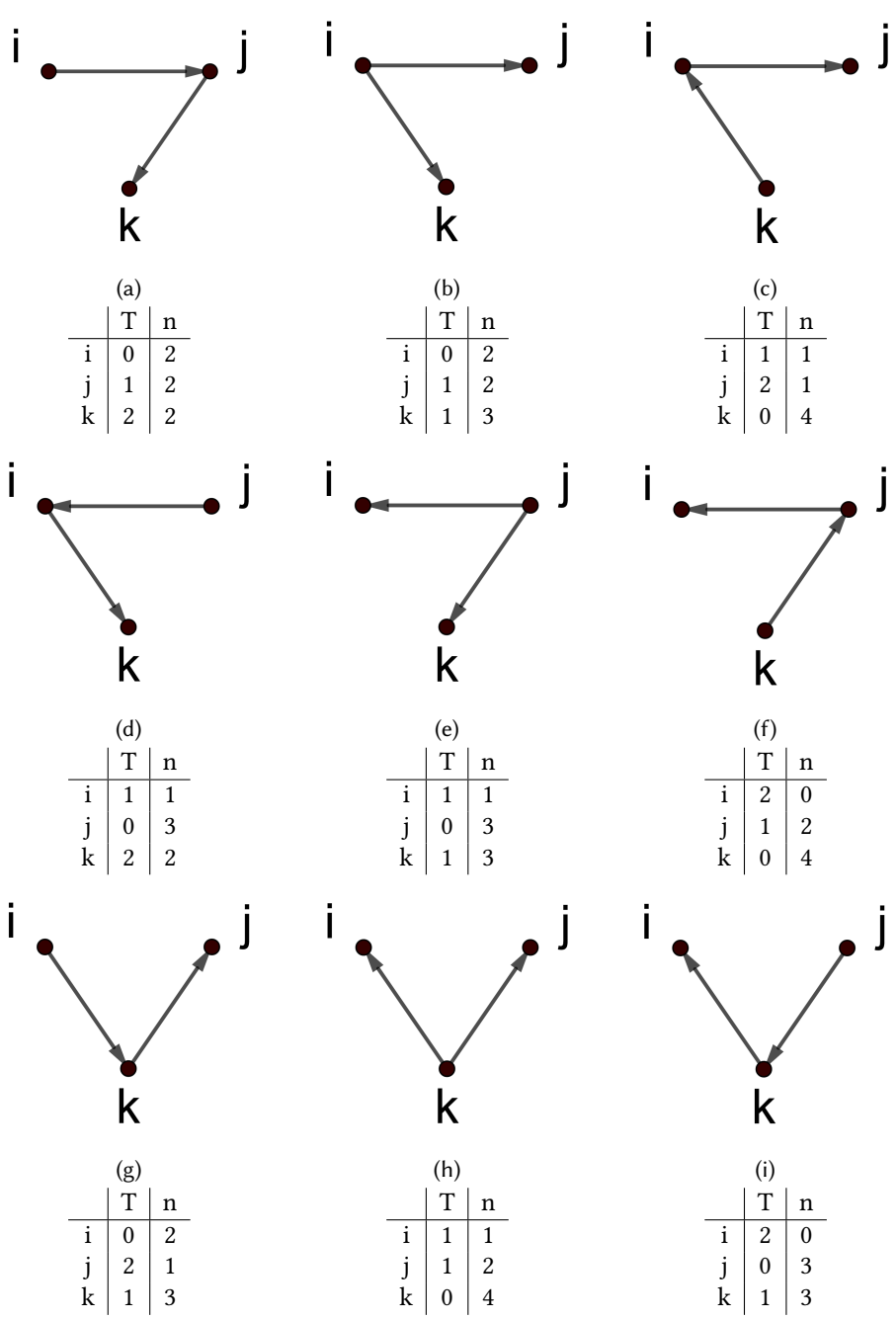
Adding noise in the time of infection not only reverses the cascade chronology, it also exponentially increased the number of possible infection patterns that *could* have happened. Bounding the realm of possibilities is therefore our second step towards solving the problem (Section 2.1).

1.2.2 Max-likelihood approaches. Another common approach is to use likelihood-based methods. For instance, in [30], the authors develop a max-likelihood-based approach to learn the edges of the graph. They prove the log-likelihood has desirable properties: it decouples into only one local problem for each node, and this local problem is convex (up to the change of variable $\theta_{ij} = -\log(1 - p_{ij})$):

$$\mathcal{L}(t, P_{ij}) = \log\left(\frac{1}{N} \cdot \prod_{1 \leq i \leq N} \underbrace{\left(\prod_{j; t_j < t_i - 1} (1 - p_{ji})\right)}_{\substack{\text{Probability that } i \text{ was not} \\ \text{infected before } t_i'.}} \cdot \underbrace{\left(1 - \prod_{j; t_j = t_i - 1} (1 - p_{ji})\right)}_{\substack{\text{Probability that } i \text{ was} \\ \text{infected at } t_i'.}}\right)$$

$$\mathcal{L}(t, P_{ij}) = \log\left(\frac{1}{N}\right) + \sum_{i=1}^{N} \sum_{j; t_j < t_i - 1} \log(1 - p_{ji}) + \sum_{j; t_j' = t_i' - 1} \log(1 - p_{ji})$$

In our setting, the log-likelihood has none of these properties. It is not convex, and it is unclear any method other than brute force could find its maximum. Moreover, it does not decouple anymore, and even computing the log-likelihood itself takes exponential time.



(a)

	T	n
i	0	2
j	1	2
k	2	2

(b)

	T	n
i	0	2
j	1	2
k	1	3

(c)

	T	n
i	1	1
j	2	1
k	0	4

(d)

	T	n
i	1	1
j	0	3
k	2	2

(e)

	T	n
i	1	1
j	0	3
k	1	3

(f)

	T	n
i	2	0
j	1	2
k	0	4

(g)

	T	n
i	0	2
j	2	1
k	1	3

(h)

	T	n
i	1	1
j	1	2
k	0	4

(i)

	T	n
i	2	0
j	0	3
k	1	3

Fig. 3. Possible scenarios which could have led to $T_i^{'} = 2$, $T_j^{'} = 3$ and $T_k^{'} = 4$. We have $T_l^{'} = T_l + n_l$. In the limited-noise setting, there are nine possible infection patterns (many more scenarios with the same infection pattern, but different noise values, are not shown).

$$\mathcal{L}(t', P_{ij}) = \log\left(\frac{1}{N}\cdot\underbrace{\sum_{(t_1,\dots,t_N)\leq(t'_1-1,\dots,t'_N-1)}}_{\text{This is the root of the difficulty.}}\cdot\prod_{1\leq i\leq N}\underbrace{\left(n(t'_i-t_i)\right)}_{\text{Noise at node } i \text{ is } t'_i-t_i.}\right.$$
$$\left.\cdot\underbrace{\left(\prod_{j;t_j<t_i-1}(1-p_{ji})\right)}_{\text{Probability that } i \text{ was not infected before } t'_i.}\cdot\underbrace{\left(1-\prod_{j;t_j=t_i-1}(1-p_{ji})\right)}_{\text{Probability that } i \text{ was infected at } t'_i.}\right)$$
$$= \log\left(\frac{1}{N}\right)+\log\left(\sum_{\substack{(t_1,\dots,t_N)\leq\\(t'_1-1,\dots,t'_N-1)}}\prod_{1\leq i\leq N}n(t'_i-t_i)\left(\prod_{j;t_j<t_i-1}(1-p_{ji})\right)\cdot\left(1-\prod_{j;t_j=t_i-1}(1-p_{ji})\right)\right)$$

When dealing with hidden variables, a common technique would be to use the Expectation-Maximization algorithm [8]. However, in our setting, the number of hidden states is $\prod_{i=1}^{N} t'_i$, which can be as large as $(N-1)^N$. This prohibits any realistic use of the Expectation-Maximization algorithm for networks with more than twelve nodes. Moreover, except for the recent contributions [24], very little is known about the theoretical convergence of the Expectation-Maximization algorithm.

1.3 Contributions

The contributions of this article are multiple:

- To the best of our knowledge, we are the first to tackle the problem of learning the edges of a graph from noisy times of infection, a simple but natural extension of a well-known problem.
- We provide the first efficient algorithm for learning the structure and weights of a bidirectional tree in this setting. We also establish a tree-specific lower bound which shows that our algorithm is sample-optimal for learning the structure of the tree (Section 2).
- We prove it is possible to learn the structure of any bounded-degree graph in the extreme setting for which we only have access to the infection status (*i.e.,* whether or not a node was infected). Moreover, we can do so with optimal sample complexity, according to the bound established in [30].
- We provide polynomial algorithms for learning the weights of bounded degree graphs.
- Finally, we extend the results from bounded-degree graphs to general graphs. This proves the problem is solvable under any noise distribution, although the exponential sample complexity and running time prohibits any use of this algorithm in practice (Section 3.2).

2 LEARNING BIDIRECTIONAL TREES

The bidirectional tree is the simplest example which illustrates some of the difficulties of the noisy setting. Indeed, for a directed tree, the true sequence of infections can be reconstructed, and we can use techniques from the no-noise setting. For a bidirectional tree, those techniques cannot be extended. However, the uniqueness of paths in the bidirectional tree still makes this problem considerably easier than the general setting. We therefore start by presenting a solution for the bidirectional tree. The key ideas here generalize to the neighborhood-based decomposition we introduce below, which forms our key conceptual approach for the general problem.

This section contains three contributions. First, we show how to learn the structure of a tree using only the infection status, *i.e.*, what we call the extreme-noise setting (Section 2.1). For each cascade, we only know which nodes were infected. We show this contains enough information to learn the structure of bidirectional tree. Second, we establish a lower bound for the no-noise setting, and show our algorithm has the same dependency in the number of nodes N as this lower bound. In other words, for the task of learning the structure of any tree, an optimal algorithm in the no-noise setting would need as many cascades as our algorithm needs in the extreme-noise setting (up to constants).

Finally, we show how we can leverage this learned structure to learn the weights of the tree, this time when we have noisy access to the times of infection, *i.e.,* the limited-noise setting (Section 2.3). We provide sample complexity for this task.

2.1 Tree structure

As illustrated in Section 1.2, the number of edges that *could* exist is much higher in the limited-noise setting than the number of actual edges in the tree. Our first key contribution is therefore to introduce a new estimator, $\hat{h}_{ij}$, which keeps track of the fraction of cascades for which i and j were both infected. This estimator can therefore be computed in the extreme-noise setting, with the information contained in the infection status alone. Using this estimator, we show that in the specific case where the graph is a tree, we learn the structure of this tree, $i.e.$ whether or not both $p_{ij} = p_{ij} = 0$.

Our algorithm for learning the edges of the tree relies on one central observation: $h_{i,j}$ achieves a kind of local maximum if there is an edge between i and j (Lemma 2.2). This observation relies heavily on the fact that there is uniqueness of paths on a tree. Let us now dive into the proof.

Definition 2.1. Let $\hat{h}_{i,j}$ be the fraction of cascades in which both i and j became infected. We have:

$$\hat{h}_{i,j} \to_{M\to\infty} \mathbb{P}(I_i^m \ \& \ I_j^m) := h_{i,j}.$$

We now show that the limit $h_{i,j}$ of the estimator $\hat{h}_{i,j}$ satisfies a local maximum property on the edges of the tree:

Lemma 2.2. *If i and j are not neighbors, let $(u_0, u_1, \dots, u_L)$ be the path between them, with $u_0 = i$, $u_L = j$, and $d > 1$. Then:*

$$\forall r \in \{0, \dots, L-1\},\ h_{i,j} < h_{u_r, u_{r+1}}.$$

Proof. We consider the case in which both i and j have been infected. There is a unique source of infection and a unique path between i and j. Therefore, all the nodes on the path from i to j must have been infected as well. In particular, both u_r and u_{r+1} were infected. This shows $I_i^m \ \& \ I_j^m \Rightarrow I_{u_r}^m \ \& \ I_{u_{r+1}}^m$, so $\mathbb{P}(I_i^m \ \& \ I_j^m) \leq \mathbb{P}(I_{u_r}^m \ \& \ I_{u_{r+1}}^m)$, and therefore $h_{i,j} \leq h_{u_r, u_{r+1}}$.

What's more, every time u_r and u_{r+1} became infected, at least one more infection along an edge must have occurred in order for j to become infected as well. This occurred with probability at

most $p_{max} < 1$. Therefore, $h_{i,j} = \mathbb{P}(I_i^m \ \& \ I_j^m) \leq p_{max} \cdot \mathbb{P}(I_{u_r}^m \ \& \ I_{u_{r+1}}^m) = p_{max} \cdot h_{u_r,u_{r+1}}$. We conclude $h_{i,j} < h_{u_r,u_{r+1}}$.

□

This simple lemma allows us to design Algorithm 1. Indeed, suppose we have access to all the limits $h_{i,j}$. By ordering them in decreasing order, we can deduce the structure of the tree by greedily adding every edge unless it forms a cycle[3].

Algorithm 1 Learn the undirected edges of the tree.

```
1: procedure LearnTree({h_{i,j}}_{i,j∈V})                ▷ h_{i,j} limit of ĥ_{i,j}.
2:     sorted_h ← Sort({h_{i,j}}_{i,j∈V}) by decreasing order
3:     edges_tree ← []
4:     for h_{i,j} ∈ sorted_h do
5:         if Adding corresponding edge (i, j) to edges_tree does not create a cycle then
6:             Add (i, j) to edges_tree
       return edges_tree
```

We show that if we have access to the limits $h_{i,j}$ of the estimators $\hat{h}_{i,j}$, the algorithm above correctly find the structure of the tree.

Lemma 2.3. *Algorithm 1 correctly finds all the $N-1$ pairs (i,j) such that there exists at least one directed edge between i and j.*

Proof. We show that in the for-loop at line 4, we add an edge to *edges_tree* if and only if this edge was a real edge in the original tree. We prove it by induction on the elements of the sorted list *pairs_h_edge*.

When no element has been selected, the proposition is trivially true.
Suppose now that t elements of *pairs_h_edge* have been examined so far. Let $(\sim,(i,j))$ be the $t+1^{th}$ element. Two cases arise:

(1) i and j are not neighbors. Let $(u_0,\dots,u_L)$ be the path between them, with $u_0 = i$ and $u_d = j$. In this case, using Lemma 2.2, $\forall r$, $h_{u_r,u_{r+1}} > h_{i,j}$. In other words, all the pairs (u_r,u_{r+1}) have already been considered by the algorithm. By induction, we have kept all of them in *edges_tree*. Therefore, adding the pair (i,j) would form a cycle. This pair is not kept in *edges_tree*, which is what we wanted since it is not an edge in the original tree.

(2) i and j are neighbors. Suppose that adding this pair forms a cycle. Then there is a sequence $(v_0 = i,\dots,v_{L'} = j)$ of nodes such that $h_{v_k,v_{k+1}}$ were all bigger than $h_{i,j}$, and the pairs (v_k,v_{k+1}) were kept by the algorithm for all k. However, by uniqueness of paths in a tree, there exists a pair (v_a,v_{a+1}) such that the path connecting v_a and v_{a+1} in the original tree goes through (i,j). Using Lemma 2.2, this means $h_{i,j} > h_{v_a,v_{a+1}}$, which is a contradiction. Therefore, adding this pair in *edges_tree* does not form a cycle. This pair is kept in *edges_tree*.

Therefore, this algorithm keeps all the edges, and only the edges of the tree, so it recovers the tree structure.

□

We next quantify how many cascades M are needed for Algorithm 1 to be correct if we replace the $\{h_{i,j}\}_{i,j\in V}$ by their estimates $\{\hat{h}_{i,j}\}_{i,j\in V}$. We note that we do not require $\hat{h}_{i,j}$ to be close to their

[3]This algorithm is very similar in spirit to Kruskal's algorithm for finding a maximum spanning tree [23].

limit, but only need the order of the $\hat{h}_{i,j}$ to be the same as the order of the $h_{i,j}$. We identify events which guarantee that the order is the same (Corollary 2.6):

Definition 2.4. Let $\mathcal{H}_3 := \{(i,j,k) \in \{1,\dots,N\}^3, p_{ij} + p_{ji} > 0 \ \& \ p_{jk} + p_{kj} > 0\}$ be the set of triplets of nodes such that at least one directed edge exists between the first and the second node, as well as between the second and the third node.

Proposition 2.5. *If:*

$$\forall (i,j,k) \in \mathcal{H}_3, \ \hat{h}_{i,j} > \hat{h}_{i,k} \text{ and } \hat{h}_{j,k} > \hat{h}_{i,k},$$

then for all paths $(u_0,\dots,u_L)$ in the tree, with $L > 1$, we have:

$$\forall r \in \{0,\dots,L-1\}, \ \hat{h}_{u_r,u_{r+1}} > \hat{h}_{u_0,u_L}.$$

Proof. For $r \in \{0,\dots,L-2\}$, we have by hypothesis $\hat{h}_{u_r,u_{r+1}} > \hat{h}_{u_r,u_{r+2}}$. Now, we recall that $\hat{h}_{u_0,u_L}$ is the number of cascades for which both u_0 and u_L were infected. By uniqueness of paths in the tree, every time both u_0 and u_L were infected, both u_r and u_{r+2} must have been infected as well. This shows that $\hat{h}_{u_0,u_L} \leq \hat{h}_{u_r,u_{r+2}}$. Notice that this is a deterministic property, not an asymptotic property. Therefore, $\hat{h}_{u_r,u_{r+1}} > \hat{h}_{u_r,u_{r+2}} \geq \hat{h}_{u_0,u_L}$.

For $r = L-1$, we follow an identical reasoning, but with $\hat{h}_{u_r,u_{r+1}} > \hat{h}_{u_{r-1},u_{r+1}}$.

Corollary 2.6. *If:*

$$\forall (i,j,k) \in \mathcal{H}_3, \ \hat{h}_{i,j} > \hat{h}_{i,k} \text{ and } \hat{h}_{j,k} > \hat{h}_{i,k},$$

then the correctness of Algorithm 1 is preserved when given $\hat{h}$ as input instead of h.

In other words, Algorithm 1 outputs a correct set of undirected edges with finite samples.

Proof. According to Proposition 2.5, for all paths $(u_0,\dots,u_L)$ in the tree, with $L > 1$, we have that $\forall r \in \{0,\dots,L-1\}$, $\hat{h}_{u_r,u_{r+1}} > \hat{h}_{u_0,u_L}$. As shown in the proof of Lemma 2.3, this is the only property of the input needed in order to yield the correct output.

Proposition 2.7. *With $M = \frac{N\left(\log\left(\frac{1}{\delta}\right)+2\log(N)\right)}{p_{min}(1-p_{max})}$ cascades, with probability at least $1-\delta$, we have:*

$$\forall (i,j,k) \in \mathcal{H}_3, \ \hat{h}_{i,j} > \hat{h}_{i,k} \text{ and } \hat{h}_{j,k} > \hat{h}_{i,k}.$$

Proof. Let us consider one triplet (i,j,k) in $\mathcal{H}_3$. We recall that $\hat{h}_{i,k}$ is the number of cascades for which both i and k were infected. Since the only path from i to k is through j, we always have that $\hat{h}_{i,j} \geq \hat{h}_{i,k}$ and $\hat{h}_{j,k} \geq \hat{h}_{i,k}$. We notice that to obtain $\hat{h}_{i,j} > \hat{h}_{i,k}$, we only need one cascade for which both i and j got infected, but not k. We lower bound the probability $P_{\text{triplet identified}}$ of this cascade happening. For each cascade m, we have:

$$\begin{aligned}
P_{\text{triplet identified}} &= \mathbb{P}(I_i^m \ \& \ I_j^m \ \& \ \textsc{Not}(I_k^m)) \\
&\geq \mathbb{P}(i \text{ was a source, } i \text{ infected } j, j \text{ did not infect } k) \\
&\quad + \mathbb{P}(j \text{ was a source, } j \text{ infected } i, j \text{ did not infect } k) \\
&\geq \frac{1}{N} \cdot p_{ij} \cdot (1-p_{jk}) + \frac{1}{N} \cdot p_{ji} \cdot (1-p_{jk}) \\
&\geq \frac{1}{N} p_{min}(1-p_{max}).
\end{aligned}$$

The probability that this event never occurs during the M cascades is upper bounded by:

$$\mathbb{P}(\hat{h}_{i,j} = \hat{h}_{i,k}) \leq \left(1 - \frac{1}{N} p_{min}(1-p_{max})\right)^M$$

$$\leq e^{-\frac{M}{N} p_{min}(1-p_{max})}$$
$$\leq \frac{\delta}{N^2}.$$

Now, there are $N-1$ edges in a tree, therefore $|\mathcal{H}_3| \leq (N-1)^2 < N^2$. By union bound:

$$\mathbb{P}\left(\bigcup_{(i,j,k)\in\mathcal{H}_3} \hat{h}_{i,j} = \hat{h}_{i,k}\right) \leq \sum_{(i,j,k)\in\mathcal{H}_3} \mathbb{P}(\hat{h}_{i,j} = \hat{h}_{i,k})$$
$$< N^2 \cdot \frac{\delta}{N^2}$$
$$\leq \delta.$$

Notice that $\mathcal{H}_3$ contains both (i,j,k) and (k,j,i). We have therefore proven that with probability at least $1-\delta$, when considering $M = \frac{N\left(\log\left(\frac{1}{\delta}\right)+2\log(N)\right)}{p_{min}(1-p_{max})}$ cascades, we have $\forall (i,j,k) \in \mathcal{H}_3,\ \hat{h}_{i,j} > \hat{h}_{i,k}$ and $\hat{h}_{j,k} > \hat{h}_{i,k}$.

Putting together Proposition 2.7 and Corollary 2.6, we obtain our first theorem for learning the undirected edges with finite samples:

Theorem 2.8. *With $M = \frac{N\left(\log\left(\frac{1}{\delta}\right)+2\log(N)\right)}{p_{min}(1-p_{max})}$ cascades, with probability at least $1-\delta$, we can learn the structure of a any bidirectional tree in the extreme-noise setting, i.e., when we only have access to the infection status of the nodes.*

2.2 Lower bound

In this section, we prove an information-theoretical lower bound for trees in the no-noise setting. With very minor adjustments, we adapt the lower bound of [30]. Since for a general tree, the colorred maximum degree can be up to $N-1$, we design a lower bound which is independent from the max-degree. Let G be a tree drawn uniformly from $\mathcal{G}$, the set of all possible trees on N nodes, and $\hat{G}$ be the reconstructed graph from the times of infection. $G \leftrightarrow T \leftrightarrow \hat{G}$ therefore forms a Markov chain. If we denote by $H(A)$ the entropy of the random variable A, we have:

$$\begin{aligned}
H(G) &= I(G;\hat{G}) + H(G|\hat{G}) \\
\text{(data processing inequality)} \quad &\leq I(G;T) + H(G|\hat{G}) \\
\text{(independent cascades)} \quad &\leq M \cdot I(T^1;T'^1) + H(G|\hat{G}) \\
(I(X,Y) \leq H(X)) \quad &\leq M \cdot \sum_{i=1}^{N} H(T_i^1) + H(G|\hat{G}) \\
\text{(Fano's inequality)} \quad &\leq M \cdot \sum_{i=1}^{N} H(T_i^1) + (1 + P_e \log(|\mathcal{G}|))
\end{aligned}$$

Since G is drawn uniformly from $\mathcal{G}$, $H(G) = \log(|\mathcal{G}|)$. There are N^{N-2} trees on N nodes, according to Cayley's formula [5], so $H(G) = (N-2)\cdot\log(N)$.

In conclusion:

$$M \geq \frac{(1-P_e)(N-2)\log(N) - 1}{N \cdot H(T_i^1)} = \Omega\left(\frac{\log(N)}{H(T_i^1)}\right)$$

Using the same kind of techniques as in [30], we can assume $H(T_i^1) \sim \frac{1}{N}$. Therefore:

Theorem 2.9. *In the no-noise setting, we need $M = \Omega(N\log(N))$ cascades to learn the tree structure.*

In our extreme-noise setting, when we have only access to the infection status of the nodes, we can learn the tree structure with the same sample complexity as the no-noise setting!

2.3 Tree weights

In this section, we assume we are in the limited-noise setting, and we have access to the times of infection. We also assume we have already learned the structure of the tree.

Once we have reduced the set of possible edges by learning the structure of the bidirectional tree, learning the weights of the edges is still non-trivial. Indeed, from $T_i^{'m}$ and $T_j^{'m}$, it is still impossible to know whether this sample is useful for estimating p_{ij} (case when i infected j), or whether we should use this sample for estimating p_{ji} instead (case when j infected i). What is more, we only get one sample per node and per cascade, so it is impossible to know what really happened during that cascade. However, knowing the distribution of the noise, it is possible to compute the probability that the noise maintained the order of infections. Using this information and the reduced set of known undirected edges, we can compute two sets of $N(N-1)$ estimators, from which it is possible to infer the weights of all edges in the tree.

We introduce these two sets of $N(N-1)$ estimators, or, in other words, two estimators for each directed edge. These estimators tend to multivariate polynomials of the weights of most edges of the tree. Thus in general these polynomials have exponentially many terms; however, when i and j are neighbors, it is possible to express them concisely using a quantity $\mathcal{P}_{\setminus j}(\to i)$, which we define formally below. This succinct representation is the key idea we exploit to solve the resulting system of equations. Once we know the structure of the bidirectional tree, we can consider the four estimators for each undirected edge (two estimators for each directed edge). They form a system of four equations and four unknowns, which we solve to obtain the weights of the edges.

Definition 2.10. We define two new notations:

- $\mathcal{P}_{\setminus j}(\to j)$ is the probability that j became infected before any node on the path from j to i, including i, became infected
- $p_{i\to j}$ is the probability that there exists a path $(u_0, \dots, u_L)$, with $u_0 = i$ and $u_L = j$, such that for all $r \in \{0, \dots, L-1\}$, u_r infected u_{r+1}.

To simplify the following notations, let us introduce s_k, which is the probability that the noise on j has delay at least k relative to the noise on i. Since the noise is i.i.d., this value is independent from i and j: $s_k = \mathbb{P}(n_j - n_i \geq k)$[4]. For instance, if i infected j during cascade m, the probability that the noise did not flip the order of infection (*i.e.* $T_i^{'m} < T_j^m$) is $\mathbb{P}(n_j \geq n_i) = s_0$. In the reverse case, the probability that the noise flipped the order of infection is $\mathbb{P}(T_j^m + n_j < T_i^{'m} + n_i) = \mathbb{P}(1 + n_j < n_i) = \mathbb{P}(n_i - n_j \geq 2) = s_2$.

We now introduce the estimators:

Definition 2.11. We introduce 2 sets of $N(N-1)$ estimators:

$$\hat{f}_{i<j} = \text{Fraction of infections for which } i \text{ and } j \text{ got infected, and } i \text{ reported before } j.$$

$$\hat{g}_{i,\setminus j} = \text{Fraction of infections for which } i \text{ got infected,}$$

[4]For instance, for geometric noise of parameter q, we have: $s_k = \sum_{t_j=\max(0,k)}^{\infty} \sum_{t_i=0}^{t_j-k} (1-q)^{t_i+t_j} = (1-q)^{\max(0,k)} \left(1 - \frac{(1-q)^{1-\min(0,k)}}{2-q}\right)$.

but j did not.

By the law of large numbers, as the number of cascades scales, $\hat{f}_{i<j}$ tends to $f_{i<j}$ and $\hat{g}_{i,\not\to}$ to $g_{i,\not\to}$, where

$$\begin{aligned} f_{i<j} &:= \mathbb{P}(T_i'^m < T_j'^m < \infty), \\ g_{i,\not\to} &:= \mathbb{P}(T_i'^m < \infty, T_j'^m = \infty). \end{aligned}$$

We now compute the exact values of these two quantities. Let us assume the (unique) path between i and j has length L. We call $(u_0, \dots, u_L)$ the set of nodes on the path from i to j, with $u_0 = i$ and $u_L = j$. We then have:

Lemma 2.12. *Recall $f_{i<j}$ and $g_{i,\not\to}$ are the expectation of the estimators defined above. We have:*

$$\begin{aligned} f_{i<j} = \mathcal{P}_{\not\to}(\to i) \cdot p_{i\to j} \cdot s_{-L+1} + \mathcal{P}_{\not\to}(\to j) \cdot p_{j\to i} \cdot s_{L+1} \\ + \sum_{l=1}^{L-1} \mathcal{P}_{\not\to,\not\to}(\to u_l) \cdot p_{u_l\to i} \cdot p_{u_l\to j} \cdot s_{2l-L+1}. \end{aligned}$$

$$\hat{g}_{i,\not\to} = \mathcal{P}_{\not\to}(\to i) \cdot (1 - p_{i\to j}) + \sum_{l=1}^{L-1} \mathcal{P}_{\not\to,\not\to}(\to u_l) \cdot p_{u_l\to i} \cdot (1 - p_{u_l\to j}).$$

What's more, when i and j are neighbors (which implies $d = 1$), the expressions simplify to:

$$\begin{aligned} f_{i<j} &= \mathcal{P}_{\not\to}(\to i) \cdot p_{ij} \cdot s_0 + \mathcal{P}_{\not\to}(\to j) \cdot p_{ji} \cdot s_2 \\ g_{i,\not\to} &= \mathcal{P}_{\not\to}(\to i) \cdot (1 - p_{ij}). \end{aligned}$$

Proof.

$$\begin{aligned} f_{i<j} &= \mathbb{P}(T_i'^m < T_j'^m < \infty) \\ &= \mathbb{P}(i \text{ got infected before any nodes on the path from } i \text{ to } j, \\ &\quad \text{then infected } j, \text{ and the noise did not flip the times of infection}) \\ &\quad + \mathbb{P}(j \text{ got infected before any nodes on the path from } i \text{ to } j, \\ &\quad \text{then infected } i, \text{ and the noise flipped the times of infection}) \\ &\quad + \mathbb{P}(\text{One node on the path from } i \text{ to } j \text{ got infected before } i \text{ and } j, \\ &\quad \text{then infected both } i \text{ and } j, \text{ but } i \text{ reported before } j) \\ &= \mathcal{P}_{\not\to}(\to i) \cdot p_{i\to j} \cdot s_{-L+1} \\ &\quad + \mathcal{P}_{\not\to}(\to j) \cdot p_{j\to i} \cdot s_{L+1} \\ &\quad + \sum_{l=1}^{L-1} \mathcal{P}_{\not\to,\not\to}(\to u_l) \cdot p_{u_l\to i} \cdot p_{u_l\to j} \cdot s_{2l-L+1}. \end{aligned}$$

This expression is involved in general. However, if i and j are neighbors, then there are no nodes u_l on the path between i and j, other than i and j themselves. What's more, $p_{i\to j} = p_{ij}$, and $L = 1$. Therefore:

$$f_{i<j} = \mathcal{P}_{\not\to}(\to i) \cdot p_{ij} \cdot s_0 + \mathcal{P}_{\not\to}(\to j) \cdot p_{ji} \cdot s_2.$$

Let us now focus on $g_{i,\not\to}$:

$$\begin{aligned} g_{i,\not\to} &= \mathbb{P}(T_i'^m < \infty, T_j'^m = \infty) \\ &= \mathbb{P}(i \text{ got infected before any nodes on the path from } i \text{ to } j, \end{aligned}$$

but j did not get infected)
$+ \mathbb{P}$(One node on the path from i to j got infected before i and j,
then infected i but not j)
$$= \mathcal{P}_{\not j}(\to i) \cdot (1 - p_{i\to j}) + \sum_{l=1}^{L-1} \mathcal{P}_{\not i,\not j}(\to u_l) \cdot p_{u_l \to i} \cdot (1 - p_{u_l \to j}).$$

As before, this expression is complex in general, but simplifies if i and j are neighbors, in which case:

$$g_{i,\not j} = \mathcal{P}_{\not j}(\to i) \cdot (1 - p_{ij}).$$

□

Using the simplified expression only for when i and j are neighbors, we obtain:

Proposition 2.13. *If we know (i, j) is an edge in the original tree, then the probability of infection along this edge is given by:*

$$p_{ij} = \frac{f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}{g_{i,\not j} \cdot (s_0^2 - s_2^2) + f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}.$$

Proof. According to Lemma 2.12, we had four second-order equations, with 4 unknowns: p_{ij}, p_{ji}, $\mathcal{P}_{\not j}(\to i)$ and $\mathcal{P}_{\not i}(\to j)$. We solve it, and obtain the wanted result. See Appendix A for details. □

Now that we have proven the problem is solvable, we establish the number of samples needed to learn the weights with the method above.

Lemma 2.14. *With $M = \frac{N^2}{\epsilon^2} \log\left(\frac{6}{\delta}\right) \frac{\left((s_0^2 - s_2^2 + s_0 + s_2) p_{max} + s_0 + s_2\right)^2}{(s_0^2 - s_2^2)^2}$ samples, with probability at least $1 - \delta$, we have:*

$$|\hat{p}_{ij} - p_{ij}| \leq \epsilon.$$

Proof. Using Hoeffding's inequality:

$$\mathbb{P}(|\hat{f}_{i<j} - f_{i<j}| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2},$$
$$\mathbb{P}(|\hat{f}_{j<i} - f_{j<i}| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2},$$
$$\mathbb{P}(|\hat{g}_{i,\not j} - g_{i,\not j}| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2}.$$

Choosing $M = \frac{1}{\epsilon_1^2} \log\left(\frac{6}{\delta}\right)$, we have that with probability at least $1 - \delta$, all the following hold:

$$|\hat{f}_{i<j} - f_{i<j}| \leq \epsilon_1,$$
$$|\hat{f}_{j<i} - f_{j<i}| \leq \epsilon_1,$$
$$|\hat{g}_{i,\not j} - g_{i,\not j}| \leq \epsilon_1.$$

Hence, with probability at least $1 - \delta$, we have (see Appendix A for details):

$$\hat{p}_{ij} - p_{ij} \leq \epsilon_1 \frac{(s_0^2 - s_2^2 + s_0 + s_2) p_{max}}{g_{i,\not j} \cdot (s_0^2 - s_2^2) + f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2} + \epsilon_1 \frac{s_0 + s_2}{g_{i,\not j} \cdot (s_0^2 - s_2^2) + f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2} + o(\epsilon_1).$$

We use the results from Lemma 2.12 to bound the denominator by $\frac{s_0^2-s_2^2}{N}$. In the end, we obtain:

$$|\hat{p}_{ij} - p_{ij}| \leq \epsilon_1 N \frac{(s_0^2 - s_2^2 + s_0 + s_2)p_{max} + s_0 + s_2}{s_0^2 - s_2^2} + o(\epsilon_1).$$

We choose $\epsilon_1 = \frac{\epsilon}{N} \frac{s_0^2-s_2^2}{(s_0^2-s_2^2+s_0+s_2)p_{max}+s_0+s_2}$.
Noticing $((s_0^2 - s_2^2 + s_0 + s_2)p_{max} + s_0 + s_2) \leq 3$:
With $M = \frac{N^2}{\epsilon^2} \log\left(\frac{12N^2}{\delta}\right) \frac{9}{(s_0^2-s_2^2)^2}$ samples, with probability at least $1-\delta$, we have $|\hat{p}_{ij} - p_{ij}| \leq \epsilon$. □

By a union bound on all the weights of the tree, knowing there are at most $2N(N-1) < 2N^2$ directed edges in a directed tree, we obtain the following sample complexity:

THEOREM 2.15. *With $M = \frac{N^2}{\epsilon^2} \log\left(\frac{12N^2}{\delta}\right) \frac{9}{(s_0^2-s_2^2)^2}$ cascades, with probability $1-\delta$, we can learn all the weights of the edges of a bidirectional tree within precision ϵ in the limited-noise setting, i.e., when we only have access to the noisy times of infection.*

Remarks:

- The sample complexity depends on the noise only through the quantities s_0 and s_2. This means that if we do not know the distribution of the noise, but we do know it is i.i.d., and know the two quantities s_0 and s_2, we can still compute all the weights of the tree with the sample bound given above.
- For certain natural classes of distributions, the key quantity $(s_0^2 - s_2^2)^2$ (which appears in the denominator of our bounds) decreases as variance grows, and hence the impact of the variance is captured here. We note that in general, there is no monotonic relationship between the variance of the noise and the quantity $(s_0^2 - s_2^2)^2$.[5]
- If we relax the i.i.d. assumption for the noise, there are cases for which our method might not succeed. Specifically, if we can compute $\mathbb{P}(n_j - n_i \geq 0)$ and $\mathbb{P}(n_j - n_i \geq 2)$ for all pairs of nodes (i, j), and there exists a pair such that $\mathbb{P}(n_j - n_i \geq 0) \leq \mathbb{P}(n_i - n_j \geq 2)$, our method fails. If we can guarantee $\mathbb{P}(n_j - n_i \geq 0) > \mathbb{P}(n_i - n_j \geq 2)$ for all pairs (i, j), then the i.i.d. assumption is not needed, and we can get a similar (albeit slightly more complex) expression as above for the sample complexity.

3 BOUNDED-DEGREE GRAPHS

In the previous section, the algorithm presented relies heavily on the uniqueness of paths. This property implies that we can deduce the edges from the nodes which are co-infected the most often. However, this is not true for a general bounded-degree graph. In Figure 4, we can see that the two nodes i and j would be co-infected frequently despite not sharing an edge. This makes the task of finding the structure much more challenging than for the bidirectional tree.

In this section, we show how the main ideas for learning the structure of the bidirectional tree can be extended for learning the structure of general bounded-degree graphs, in the extreme-noise setting, with optimal sample complexity. The framework for learning the weights of the edges in the limited-noise setting is - to the best of our knowledge - not extendable to general bounded-degree graphs; we therefore develop a new algorithm to learn the weights for general bounded-degree graphs.

[5]For instance, if the noise is Gaussian, increasing the variance will indeed decrease the quantity $(s_0^2 - s_2^2)^2$ and increase the sample complexity. However, when the noise is Bernoulli, taking two values v_1 and v_2, increasing $|v_1 - v_2|$ increases the variance, but decreases the value $(s_0^2 - s_2^2)^2$, so the sample complexity decreases.

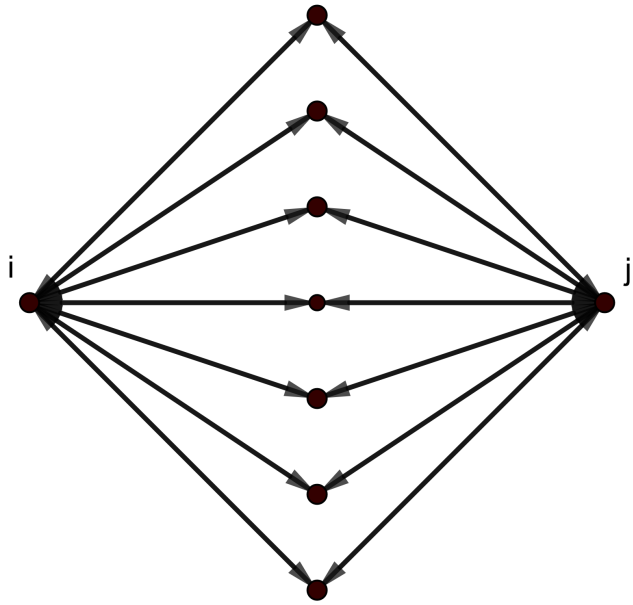


Fig. 4. Two nodes can be co-infected frequently without sharing an edge.

3.1 Bounded-degree structure

In the previous section, we introduced the estimator $\hat{h}_{i,j}$, which records the fraction of cascades for which both i and j are infected. From a local maximum property of this estimator, we deduced the structure of the tree, in a sample efficient fashion. Indeed, if there exists a path between i and k, and the first edge on this path is (i, j), then if i and k are infected, j must have been infected as well.

We want to build on this idea for a bounded-degree graph of maximum degree d. However, for such a graph, there may be multiple paths leading from i to j, and we cannot guarantee a single node will be infected each time. However, if i is a node, $\mathcal{N}_i$ is its neighborhood, and $k \notin \mathcal{N}_i$ is another node of the graph, we can guarantee that if both i and k are infected, there exists a node in $\mathcal{N}_i$ which is infected. Moreover, $\mathcal{N}_i$ is the set of smallest size for which at least one node is infected at the same time as i the most frequently. This leads us to a new set of estimators:

Definition 3.1. Let i be a node of the graph, and let S be a set such that $|S| \leq d$ and $i \notin S$. We define a new set of estimators:

$$\hat{h}_{i,S} = \text{fraction of cascades for which } i \text{ is infected,} \\ \text{and at least one node of S is infected.}$$

Let us assume that for each pair (i, S), we have access to the limit $h_{i,S}$ of $\hat{h}_{i,S}$. We now introduce an algorithm showing how to leverage these limits to learn the structure of any bounded-degree graph.

Algorithm 2 Learn the undirected edges of any graph of maximum degree d.

```
procedure LearnGraph(({h_{i,S}}_{i∈V}^{|S|≤d})
    edges ← []
    for i = 1 ... n_nodes do
        S_max_i ← set such that h_{i,S_max_i} is maximal, and such that Size(S_max_i) is minimal.
        for n_j in S_max_i do
            Add edge (i, n_j) to edges
    return edges
```

We show this algorithm is correct.

Lemma 3.2. *Algorithm 2 correctly finds the neighborhood of each node.*

Proof. Let us recall that $h_{i,S}$ is the probability that node i and at least one node of S are co-infected. To prove the correctness of this algorithm, it suffices to prove:

$$\forall i \in V, \forall S \; s.t. \; |S| \leq d, \quad h_{i,S} \geq h_{i,\mathcal{N}_i} \implies \mathcal{N}_i \subseteq S$$

Let pick a set S such that $\mathcal{N}_i \setminus S \neq \emptyset$, and let k be a node in $\mathcal{N}_i \setminus S$. We know $h_{i,S\cup\{k\}} \geq h_{i,S} + \mathbb{P}(i \text{ and } k \text{ are the only infected nodes})$. Since i and k are neighbors, $\mathbb{P}(i \text{ and } k \text{ are the only infected nodes}) > 0$, and therefore $h_{i,S\cup\{k\}} > h_{i,S}$. Following this line of reasoning, if $\mathcal{N}_i \setminus S \neq \emptyset$, S we can always increase the value of $h_{i,S}$ by adding a node of $\mathcal{N}_i$.

However, it is impossible to increase the value of $h_{i,\mathcal{N}_i}$, because if i and any other node of the graph are co-infected, we know one node of $\mathcal{N}_i$ is also infected. Therefore, the algorithm is correct. □

Unfortunately, we do not have direct access to $h_{i,S}$. We therefore study how many samples are needed to replace $h_{i,S}$ by its estimate $\hat{h}_{i,S}$ while preserving the correctness of the algorithm. Just like in Section 2.1, we notice that we do not need the $\{\hat{h}_{i,S}\}_{i\in V}^{|S|\leq d}$ to be close to their limit, we only need the indexes of their ordering to be the same. We notice that the neighborhood $\mathcal{N}_i$ of i is one of the sets which are the most often infected at the same time as i is ($\mathcal{N}_i \in \arg\max_{|R|\leq d} h_{i,R}$). Therefore, we must have that for every set S, $\hat{h}_{i,S} \leq \hat{h}_{i,\mathcal{N}_i}$. However, some other sets might achieve the same value if we do not observe enough cascades. This could happen in two cases:

- Not all the nodes of the neighborhood were infected, and therefore a subset $T_1 \subset \mathcal{N}_i$ of the neighborhood is such that $\hat{h}_{i,T_1} = \hat{h}_{i,\mathcal{N}_i}$. Since $|T_1| < |\mathcal{N}_i|$, the algorithm would return T_1 and not $\mathcal{N}_i$, which would be a failure case.
- Some other node k is always infected every time a specific node $j \in \mathcal{N}_i$ is infected. The set $T_2 = \mathcal{N}_i \setminus \{j\} \cup \{k\}$ will therefore be such that $\hat{h}_{i,T_2} = \hat{h}_{i,\mathcal{N}_i}$, and the algorithm would not know which maximum to pick. This is a failure case as well.

We identify events which guarantee that the failure cases above cannot arise. Let $E_{i,j}^m$ be the event that i and j were the only infected nodes during cascade m. $E_{i,j} = \bigcup_{1\leq m\leq M} E_{i,j}^m$ is therefore the event that there exists a cascade for which only i and j were infected. If such a cascade exists, the set $S_{max} = \arg\max_{|R|\leq d} \hat{h}_{i,R}$ *must* contain j. If the event $E_{i,j}$ happens for every node j in the neighborhood of i, we can therefore guarantee Algorithm 2 is correct. We characterize the sample complexity needed for this below.

Proposition 3.3. *Let i be a node, and let j be one of its neighbors. Let S be a set of size $|S| \leq d$, such that $i \notin S$ and $j \notin S$. With probability at least $1 - \frac{\delta}{d \cdot N^{d+2}}$, among $M = \frac{(d+2)\cdot N\log(N)+N\log\left(\frac{d}{\delta}\right)}{p_{min}(1-p_{max})^{2(d-1)}}$ cascades, there exists a cascade in which i and j are infected, but no nodes of S are infected.*

Proof. We notice that if, during cascade m, the only infected nodes are i and j, no nodes of S are infected. This is exactly the event $E^m_{i,j}$ defined above.

$$\mathbb{P}(E^m_{i,j}) \geq \frac{1}{N} \cdot p_{min} \cdot (1 - p_{max})^{2(d-1)}.$$

The probability that this never happens during M cascade is exactly the complement of the event $E_{i,j}$ defined above:

$$\begin{aligned}
\mathbb{P}(\text{Not}(E_{i,j})) &\leq \left(1 - \frac{1}{N} \cdot p_{min} \cdot (1 - p_{max})^{2(d-1)}\right)^M \\
&\leq e^{-M \cdot \frac{p_{min}}{N} \cdot (1-p_{max})^{2(d-1)}} \\
&\leq \frac{\delta}{d \cdot N^{d+2}}.
\end{aligned}$$

□

Lemma 3.4. *With probability at least $1 - \delta$, if we observe $M = \frac{(d+2)\cdot N\log(N)+N\log\left(\frac{d}{\delta}\right)}{p_{min}(1-p_{max})^{2(d-1)}}$ cascades, Algorithm 2 is correct, and has running time $O(M \cdot N^{d+2}) = O(d \cdot N^{d+3}\log(N))$.*

Proof. Let $\mathcal{S} = \{S \in \mathcal{P}(V), |S| \leq d\}$ be the set of sets of nodes of size at most d, let $\mathcal{S}_{\setminus i} = \{S \in \mathcal{S}, i \notin S\}$ be the set of sets of $\mathcal{S}$ which do not contain i, and let $\mathcal{N}_i = \{j, (i,j) \in E\}$ be the neighborhood of i. We pick a set $S \in \mathcal{S}$, such that $S \neq \mathcal{N}_i$.

We first prove that $\hat{h}_{i,S} \leq \hat{h}_{i,\mathcal{N}_i}$. Since the neighborhood of i separates i from the rest of the graph, and infected nodes are connected, we can conclude that every time i and another node of S are infected, one node in the neighborhood $\mathcal{N}_i$ of i is infected as well. Therefore, we cannot increase $\hat{h}_{i,S}$ without also increasing $\hat{h}_{i,\mathcal{N}_i}$. In particular, this means that even if $|S| > |\mathcal{N}_i|$ (for instance if $\mathcal{N}_i \subset S$), we have $\hat{h}_{i,S} \leq \hat{h}_{i,\mathcal{N}_i}$. We therefore know that $\mathcal{N}_i \in \arg\max_{R \in \mathcal{S}_{\setminus i}} \hat{h}_{i,R}$.

We now prove that with probability at least $1 - \delta$, $\mathcal{N}_i$ is the set of $\arg\max_{R \in \mathcal{S}_{\setminus i}} \hat{h}_{i,R}$ of minimal size. To do so, we notice that if there exists a cascade such that i and $j \in \mathcal{N}_i$ are infected, but no node of S is infected, then this implies $\hat{h}_{i,S} < \hat{h}_{i,\mathcal{N}_i}$. Indeed, as shown above, every time we increase $\hat{h}_{i,S}$, we also increase $\hat{h}_{i,\mathcal{N}_i}$, and we know there exists one cascade for which we increased $\hat{h}_{i,\mathcal{N}_i}$ without increasing $\hat{h}_{i,S}$. We now calculate the probability P_{failure} that such a cascade does not exist for all nodes i in the graph, all nodes j in their neighborhood, and all sets S which do not include i or j.

$$\begin{aligned}
P_{\text{failure}} \leq\ & \mathbb{P}(\exists i \in V, \exists j \in \mathcal{N}_i, \exists S \in \mathcal{S}_{\setminus i,j}, \text{every time } i \text{ and } j \text{ are} \\
& \text{infected, a node of } S \text{ is also infected}) \\
\leq\ & \sum_{i \in V} \sum_{j \in \mathcal{N}_i} \sum_{S \in \mathcal{S}_{\setminus i,j}} \mathbb{P}(\text{every time } i \text{ and } j \text{ are infected,} \\
& \text{a node of } S \text{ is also infected}).
\end{aligned}$$

We use Proposition 3.3 to bound this quantity:

$$\begin{aligned} P_{\text{failure}} &\leq \sum_{i \in V} \sum_{j \in \mathcal{N}_i} \sum_{S \in \mathcal{S}_{\setminus i,j}} \frac{\delta}{d \cdot N^{d+2}} \\ &\leq N \cdot d \cdot N^{d+1} \cdot \frac{\delta}{d \cdot N^{d+2}} \\ &\leq \delta \end{aligned}$$

We now know that with probability at least $1-\delta$:

$$\forall R \in \bigcup_{j \in \mathcal{N}_i} \mathcal{S}_{\setminus i,j}, \quad \hat{h}_{i,R} < \hat{h}_{i,\mathcal{N}_i}.$$

This implies that no set of $\bigcup_{j \in \mathcal{N}_i} \mathcal{S}_{\setminus i,j}$ can belong in $\arg\max_{R \in \mathcal{S}_{\setminus i}} \hat{h}_{i,R}$. However, we have:

$$\mathcal{S}_{\setminus i} \setminus \bigcup_{j \in \mathcal{N}_i} \mathcal{S}_{\setminus i,j} = \{S \in \mathcal{S}, \mathcal{N}_i \subseteq S\}$$

In particular, this means that $\mathcal{N}_i$ is the only set of $\mathcal{S}_{\setminus i} \setminus \bigcup_{j \in \mathcal{N}_i} \mathcal{S}_{\setminus i,j}$ of minimal size. This shows that with probability at least $1-\delta$, $\mathcal{N}_i$ is the set of $\arg\max_{R \in \mathcal{S}_{\setminus i}} \hat{h}_{i,R}$ of minimal size. This proves Algorithm 2 is correct, and that we can learn the structure of any graph of maximum degree d with $M = \frac{(d+2) \cdot N \log(N) + N \log\left(\frac{d}{\delta}\right)}{p_{min}(1-p_{max})^{2(d-1)}}$ cascades.

Since we do at most one operation by pair of (node, set) and by cascade, the running time is $O(N \cdot N^{d+1} \cdot M)$, which is what we wanted to prove.

□

This leads to our theorem for learning the structure of any bounded-degree graph.

Theorem 3.5. *With probability at least $1-\delta$, in the extreme-noise setting, we can learn the structure of any graph of maximum degree d with $M = O\left(\frac{d \cdot N \log\left(\frac{N}{\delta}\right)}{p_{min}(1-p_{max})^{2d}}\right)$ cascades in polynomial time.*

Let us now assume that $p_{max} \sim \frac{1}{d}$. This assumption is reasonable when you expect a constant number of infections by time step. For instance, it makes sense for real diseases, for which carriers have to meet to transmit it (we can only meet a constant number of people each day). It would not make sense for social networks, in which it is possible to reach many followers with each post.

Corollary 3.6. *With probability at least $1-\delta$, in the extreme-noise setting, if we assume $p_{max} \sim \frac{1}{d}$ and p_{min} constant, we can learn the structure of any graph of maximum degree d with $M = O\left(d \cdot N \log\left(\frac{N}{\delta}\right)\right)$. This sample complexity is optimal, and matches the lower bound established in the no-noise setting.*

Proof. We need at least $O(d \cdot N \log(N))$ samples to learn the structure of a bounded-degree graph with maximum degree d, according to the lower bound in [30].

3.2 Bounded-degree weights

For the remainder of this paper, we state results in the limited-noise setting.

If we consider cascades of size k, the exact probability of infection between two nodes is a multivariate polynomial of degree N on $N(N-1)$ variables (the variables here would be the weights

of the graph), with a sum of up to $2 \cdot \sum_{l=1}^{k} \frac{(N-2)!}{(N-k)!}$ terms. If the graph has more than five nodes, the resulting polynomial is of degree more than five.

For our algorithm, we therefore only use cascades of size 1 or 2. This is a waste of the data, since we simply discard cascades of larger size. However, we are not aware of techniques on how to utilize larger cascades in the limited-noise setting. Cascades of size 1 or 2 are simple enough that we can write explicitly their probability. This allows use to:

(1) Design estimators for which we can calculate the exact limit.
(2) Combine these estimators to transform a polynomial system of degree N to a polynomial system of degree 2.
(3) Solve this system exactly and obtain the probabilities of infection.

3.3 Estimators

We start by designing a few estimators:

Definition 3.7. We introduce two sets of $N(N-1)$ estimators and one set of N estimators. These estimators can be computed even if we only have access to the noisy times of infection.

$$\begin{aligned}
\hat{h}^2_{i,j} &= \text{Fraction of cascades for which only } i \text{ and } j \text{ are infected.} \\
\hat{f}^2_{i<j} &= \text{Fraction of cascades for which only } i \text{ and } j \text{ are infected,} \\
&\quad \text{and } t'_i < t'_j. \\
\hat{e}^1_i &= \text{Fraction of cascades for which only } i \text{ is infected.}
\end{aligned}$$

Using the definition of s_k from Section 2.3, we now compute the limits of those estimators.

Proposition 3.8. *As the number of cascades M goes to infinity, the estimators introduced above tend to the following limit:*

$$\hat{h}^2_{i,j} \to_{M\to\infty} \frac{1}{N}(p_{ij}+p_{ji}) \prod_{k\neq i,j}(1-p_{ik})(1-p_{jk})$$

$$\hat{f}^2_{i<j} \to_{M\to\infty} \frac{1}{N}(p_{ij}\cdot s_0 + p_{ji}\cdot s_2) \prod_{k\neq i,j}(1-p_{ik})(1-p_{jk})$$

$$\hat{e}^1_i \to_{M\to\infty} \frac{1}{N}(1-p_{ij}) \prod_{k\neq i,j}(1-p_{ik})$$

Proof. The proof is very similar to 2.12. See details in Appendix B.2. □

3.4 Solving the system

The limit of these estimators, as seen as a function of the probabilities of infection, is a complex polynomial on up to $2(N-1)$ variables. The crux of our algorithm is to combine those estimators in order to cancel out most of these variables, and create $N-1$ systems of two equations of degree 2 and two unknowns, which we then solve.

Proposition 3.9. *Let $\hat{V}_{ij} = \frac{\hat{f}^2_{i<j}}{\hat{h}^2_{i,j}+N\cdot\hat{e}^1_i\cdot\hat{e}^1_j}$. Then the limit of $\hat{V}_{ij}$ as the number of cascades M goes to infinity only depends on the variables p_{ij} and p_{ji} :*

$$\hat{V}_{ij} \to_{M\to\infty} \frac{p_{ij}\cdot s_0 + p_{ji}\cdot s_2}{1+p_{ij}\cdot p_{ji}}.$$

Proof. Since all the estimators converge towards a constant, we can use Slutsky's lemma to find the limit of $\hat{V}_{ij}$. Let V_{ij} be the limit of $\hat{V}_{ij}$ as M goes to infinity. We notice that the $\prod_{k\neq i,j}(1-p_{ik})(1-p_{jk})$ parts cancel each other out:

$$\begin{aligned} V_{ij} &= \frac{f^2_{i<j}}{h^2_{i,j} + N \cdot e^1_i \cdot e^1_j} \\ &= \frac{\frac{1}{N}(p_{ij}\cdot s_0 + p_{ji}\cdot s_2)\left[\prod_{k\neq i,j}(1-p_{ik})(1-p_{jk})\right]}{\left(\frac{1}{N}(p_{ij}+p_{ji}) + N\frac{(1-p_{ij})(1-p_{ji})}{N^2}\right)\left[\prod_{k\neq i,j}(1-p_{jk})(1-p_{jk})\right]} \\ &= \frac{(p_{ij}\cdot s_0 + p_{ji}\cdot s_2)}{1+p_{ij}\cdot p_{ji}}. \end{aligned}$$

□

We can therefore use this equality to deduce the weights of all the edges of the graph:

Theorem 3.10. *For any graph, for any noise distribution having finite values, we can learn the weights of all the edges of the graph. In particular, we can compute a quantity which converges to the true weight of each edge:*

$$\hat{p}_{ij} = \frac{2(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)}{\left(s_0^2 - s_2^2\right) + \sqrt{\left(s_0^2 - s_2^2\right)^2 - 4(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)(\hat{V}_{ij}s_2 - \hat{V}_{ji}s_0)}}.$$

Proof. We present a sketch of the proof here. The details can be found in Appendix B.1. We know $\hat{V}_{ij}$ tends to $V_{ij} = \frac{p_{ij}\cdot s_0 + p_{ji}\cdot s_2}{1+p_{ij}\cdot p_{ji}}$. Using both V_{ij} and V_{ji}, we can establish this second-degree equation:

$$V_{ji}s_2 - V_{ij}s_0 + \left(s_0^2 - s_2^2\right)p_{ij} + \left(V_{ij}s_2 - V_{ji}s_0\right)p_{ij}^2 = 0$$

We recall that by definition, $s_0 \geq s_2$. We also notice that if $p_{ij} = q_1$ and $p_{ji} = q_2$ is a pair of solutions of this system, then $p_{ij} = \frac{1}{q_2}$ and $p_{ji} = \frac{1}{q_1}$ forms the other pair of solution, which implies there is uniqueness of solutions in $[0, p_{max}]$. Since the real probabilities of infection satisfy this system, we also know the solution exists. Let $\Delta = \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. The only solution of this system in $[0, p_{max}]$ is then:

$$p_{ij} = \frac{2(V_{ji}s_2 - V_{ij}s_0)}{\left(s_0^2 - s_2^2\right) + \sqrt{\Delta}}.$$

□

3.5 Sample complexity

We establish the sample complexity needed to estimate p_{ij} with precision ϵ. To do so, we start by estimating V_{ij}. Note that we only consider the pair of nodes i and j if, among the M samples, there exists a cascade of size 2 in which i and j are the only infected nodes (*i.e.* $h^2_{i,j} > 0$). Otherwise, we set $\hat{p}_{ij} = \hat{p}_{ji} = 0$ as our estimate for p_{ij}.

Proposition 3.11. *With probability* $1 - \frac{\delta}{N^2}$, *with* $M = \frac{1}{\epsilon_V^2}\frac{16N^2}{p_{min}^2 s_2^2 (1-p_{max})^{4d}}\frac{2\log(3N)-\log(\delta)}{2}$ *samples, we can estimate* V_{ij} *with precision* ϵ_V.

Proof. We present a sketch of the proof; the details can be found in Appendix B.2. As in Proposition 2.14, we use Hoeffding's inequality:

$$\mathbb{P}(|\hat{f}^2_{i<j} - f^2_{i<j}| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2},$$
$$\mathbb{P}(|\hat{h}^2_{i,j} - h^2_{i,j}| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2},$$
$$\mathbb{P}(|\hat{e}^1_i - e^1_i| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2},$$
$$\mathbb{P}(|\hat{e}^1_j - e^1_j| > \epsilon_1) \leq 2e^{-2M\epsilon_1^2}.$$

We use this to bound above V_{ij}:

$$\hat{V}_{ij} \leq V_{ij}\left[1 + \frac{\epsilon_1}{f^2_{i<j}} + \epsilon_1 \frac{1 + N(e^1_i + e^1_j)}{h^2_{i,j} + Ne^1_i e^1_j} + o(\epsilon_1)\right]$$

By bounding below the denominators and bounding above the numerator, we finally obtain:

$$|\hat{V}_{ij} - V_{ij}| \leq \epsilon_1 \frac{4N}{p_{min}s_2(1-p_{max})^{2d}} + o(\epsilon_1).$$

Therefore, by union bound, and by choosing $\epsilon_1 \frac{4N}{p_{min}s_2(1-p_{max})^{2d}} = \epsilon_V$, and setting $2e^{-2M\epsilon_1^2} = \frac{\delta}{3N^2}$, we obtain:
With $M = \frac{1}{\epsilon_V^2} \frac{16N^2}{p^2_{min}s^2_2(1-p_{max})^{4d}} \frac{2\log(3N) - \log(\delta)}{2}$ samples, we can guarantee $|\hat{V}_{ij} - V_{ij}| \leq \epsilon_V$ with probability at least $1 - \frac{\delta}{N^2}$.

□

Once we have estimated V_{ij} with precision ϵ_V, estimating p_{ij} is unfortunately still not an easy task. Indeed, let $\Delta = \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. We have $p_{ij} = \frac{-\left(s_0^2 - s_2^2\right) + \sqrt{\Delta}}{2(V_{ij}s_2 - V_{ji}s_0)}$, which means that Δ has to be positive for this quantity to be defined. However, for general values of V_{ij} and V_{ji}, Δ can be negative. We therefore use the framework of constrained optimization to bound Δ away from 0.

Lemma 3.12. *Let* $\Delta = \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. *We have:*

$$\Delta \geq (s_0^2 - s_2^2)^2 \frac{(1 - p_{max})^2}{1 + p^2_{max}}.$$

Proof. We find the lower bound on Δ by reformulating the problem as a constrained optimization problem, and introducing the corresponding Lagrangian multipliers. The details can be found in Appendix B.1.

□

Now that we have established this bound on Δ, we can give the sample complexity needed to estimate p_{ij}.

Proposition 3.13. *Assuming we can estimate* V_{ij} *within precision* ϵ_V, *then we can estimate* p_{ij} *within precision* $\epsilon = \frac{6\epsilon_V(1+p^2_{max})}{(s_0^2 - s_2^2)^2(1-p_{max})^2}$.

Proof. This is a simple derivation which can be found in Appendix B.2.

□

We finally piece everything together, and use a union bound on all the p_{ij} to obtain the final sample complexity of our algorithm for learning the weights of general bounded-degree graphs.

Theorem 3.14. *In the limited-noise setting, with probability at least* $1-\delta$*, with* $M = O\left(\frac{e^{4p_{max}(d+1)}}{p_{min}^2 s_2^2 (s_0^2-s_2^2)^4}\frac{N^2}{\epsilon^2}\log\left(\frac{N}{\delta}\right)\right)$ *samples, we can learn the weights of any bounded-degree graph up to precision* ϵ*.*

Proof. We obtain the desired bound by combining Proposition 3.11 and Proposition 3.13. See details in Appendix B.2.

□

Once again, if we assume $p_{max} \sim \frac{1}{d}$ and p_{min} constant, we obtain the following sample complexity:

Corollary 3.15. *In the limited-noise setting, with probability at least* $1-\delta$*, if* $p_{max} \sim \frac{1}{d}$ *and* p_{min} *constant, we can learn the weights of any bounded-degree graph up to precision epsilon with* $M = O\left(\frac{1}{s_2^2 (s_0^2-s_2^2)^4}\frac{N^2}{\epsilon^2}\log\left(\frac{N}{\delta}\right)\right)$ *samples.*

4 GENERAL GRAPHS

In the limited-noise setting, we notice that nothing prevents us from using the algorithm for learning bounded-degree weights for general graph. This proves this problem is solvable for *any* graph, and *any* noise distribution. As explained in Section 1.2, this is not an obvious result. However, if we do not assume $p_{max} \sim \frac{1}{N}$, the sample complexity is now exponential.

Corollary 4.1. *In the limited-noise setting, with probability at least* $1-\delta$*, it is possible to learn all the weights of any graph within any given precision, for any noise distribution, with finite (but potentially exponential) sample complexity.*

5 DISCUSSION AND FUTURE WORK

In this paper, we presented the first results to learn the edges of a graph from noisy times of infection. We showed we learn the structure of any bidirectional tree or any bounded-degree graph (note that not all trees are bounded-degree graphs) with optimal sample complexity.

However, our results are not tight for learning the weights of bidirectional trees or bounded-degree graphs. In the no-noise setting, [30] proves this can be achieved with $O(d^2 N \log(N))$ for graphs of maximum degree d with correlation decay. Our results have sample complexity $O(N^2 \log(N))$ without any assumption of correlation decay. It is an open problem to understand whether it is possible to achieve a sample complexity of $O(d^2 N \log(N))$ in the limited-noise setting, and whether assuming correlation decay is necessary to obtain such a result.

Moreover, for learning the weights of a general bounded-degree graph, we only use cascades of size 1 or 2. If we are given an infinite number of cascades of size bigger than 2, our current algorithm cannot learn the weights of the graph. Future work could develop an algorithm without such a weakness.

All our results for learning the weights of the edges are in the limited-noise setting. Whether or not it is possible to learn the noise in the extreme-noise setting is an other question of interest for future work.

Finally, we have made no restriction on the distribution of the noise we add, other than it is finite. It would be interesting to study whether stronger restrictions on the noise (for instance Gaussian noise) would lead to stronger results. It would also be interesting to allow infinite noise, and develop algorithms which are robust to errors in the infection status of a node (our current algorithms can return wrong graph structure with only one adversarially chosen false positive).

REFERENCES

[1] Bruno Abrahao, Flavio Chierichetti, Robert Kleinberg, and Alessandro Panconesi. 2013. Trace complexity of network inference. *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '13* (2013), 491. https://doi.org/10.1145/2487575.2487664 arXiv:arXiv:1308.2954v1

[2] Ery Arias-castro, Emmanuel J Candès, and Arnaud Durand. 2011. Detection of an anomalous cluster in a network. *The Annals of Statistics* 39, 1 (2011), 278–304. https://doi.org/10.1214/10-AOS839 arXiv:arXiv:1001.3209v2

[3] Ery Arias-castro and S T Nov. [n. d.]. Detecting a Path of Correlations in a Network. ([n. d.]), 1–12. arXiv:arXiv:1511.01009v1

[4] Daniel Bernoulli and Sally Blower. 2004. An attempt at a new analysis of the mortality caused by smallpox and of the advantages of inoculation to prevent it. *Reviews in medical virology* 14 (2004), 275–288. https://doi.org/10.1002/rmv.443

[5] A. Cayley. 1897. A theorem on trees. In *Collected Mathematical Papers Vol. 13*. Cambridge University Press, 26–28.

[6] Justin Cheng, Lada A. Adamic, P. Alex Dow, Jon Kleinberg, and Jure Leskovec. 2014. Can Cascades be Predicted?. In *Proceedings of the 23rd international conference on World wide web (WWW' 14)*. https://doi.org/10.1145/2566486.2567997 arXiv:1403.4608

[7] Michela Del Vicario, Alessandro Bessi, Fabiana Zollo, Fabio Petroni, Antonio Scala, Guido Caldarelli, H. Eugene Stanley, and Walter Quattrociocchi. 2016. The spreading of misinformation online. *Proceedings of the National Academy of Sciences* (2016), 201517441. https://doi.org/10.1073/pnas.1517441113

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal ofthe Royal Statistical Society* 39, 1 (1977), 1–38.

[9] Kimon Drakopoulos, Asuman Ozdaglar, and John N. Tsitsiklis. 2014. An efficient curing policy for epidemics on graphs. *arXiv preprint arXiv:1407.2241* December (2014), 1–10. https://doi.org/10.1109/TNSE.2015.2393291 arXiv:arXiv:1407.2241v1

[10] Kimon Drakopoulos, Asuman Ozdaglar, and John N. Tsitsiklis. 2015. A lower bound on the performance of dynamic curing policies for epidemics on graphs. 978 (2015), 3560–3567. https://doi.org/10.1109/CDC.2015.7402770 arXiv:1510.06055

[11] Giulia Fanti, Peter Kairouz, Sewoong Oh, Kannan Ramchandran, and Pramod Viswanath. 2016. Rumor source obfuscation on irregular trees. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science (SIGMETRICS' 16)*. ACM, 153–164.

[12] Giulia Fanti, Peter Kairouz, Sewoong Oh, Kannan Ramchandran, and Pramod Viswanath. 2017. Hiding the Rumor Source. *IEEE Transactions on Information Theory* 63, 10 (2017), 6679–6713. https://doi.org/10.1109/TIT.2017.2696960 arXiv:1509.02849

[13] Giulia Fanti, Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. Spy vs. Spy: Rumor Source Obfuscation. *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS' 14)* (2015), 271–284. https://doi.org/10.1145/2745844.2745866 arXiv:1412.8439

[14] Mehrdad Farajtabar, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Khalil, Shuang Li, Le Song, and Hongyuan Zha. 2017. Fake News Mitigation via Point Process Based Intervention. In *Proceedings of the 34th International Conference on Machine Learning (ICML' 17)*. arXiv:1703.07823 http://arxiv.org/abs/1703.07823

[15] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. 2001. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Information Retrieval* 4, 2 (2001), 133–151. https://doi.org/10.1023/A:1011419012209 arXiv:arXiv:astro-ph/0005074v1

[16] Manuel Gomez-rodriguez, Jure Leskovec, and Andreas Krause. 2012. Inferring Networks of Diffusion and Influence. In *ACM Transactions on Knowledge Discovery from Data (TKDD' 12)*, Vol. 5. https://doi.org/10.1145/2086737.2086741

[17] Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. 2013. Structure and Dynamics of Information Pathways in Online Media. In *6th International Conference on Web Search and Data Mining (WSDM 2013)*.

[18] Jessica Hoffmann and Constantine Caramanis. 2018. The Cost of Uncertainty in Curing Epidemics. *Proceedings of the ACM on Measurement and Analysis of Computing Systems (SIGMETRICS' 18)* 2, 2 (2018), 11–13. https://doi.org/10.1145/3219617.3219622

[19] Tomoharu Iwata, Amar Shah, and Zoubin Ghahramani. 2013. Discovering Latent Influence in Online Social Activities via Shared Cascade Poisson Processes. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD' 13)*.

[20] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. 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*. https://doi.org/10.1145/956755.956769 arXiv:0806.2034v2

[21] Justin Khim and Po-Ling Loh. 2017. Permutation Tests for Infection Graphs. (2017), 1–28. arXiv:1705.07997 http://arxiv.org/abs/1705.07997

[22] Justin Khim and Po-Ling Loh. 2018. A theory of maximum likelihood for weighted infection graphs. (2018), 1–47. arXiv:arXiv:1806.05273v1 https://arxiv.org/pdf/1806.05273.pdf

[23] Joseph B. Kruskal. 1956. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proc. Amer. Math. Soc.* 7, 1 (1956), 48–50. http://www.jstor.org/stable/2033241

[24] Jeongyeol Kwon, Wei Qian, Constantine Caramanis, Yudong Chen, and Damek Davis. 2019. Global Convergence of the EM Algorithm for Mixtures of Two Component Linear Regression. XX (2019), 1–57. arXiv:arXiv:1810.05752v3

[25] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective Outbreak Detection in Networks. *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '07)* (2007), 420. https://doi.org/10.1145/1281192.1281239

[26] Eli A. Meirom, Chris Milling, Constantine Caramanis, Shie Mannor, Ariel Orda, and Sanjay Shakkottai. 2014. Localized epidemic detection in networks with overwhelming noise. (2014), 1–27. arXiv:1402.1263 http://arxiv.org/abs/1402.1263

[27] Chris Milling, Constantine Caramanis, Shie Mannor, and Sanjay Shakkottai. 2012. 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)*.

[28] Chris Milling, Constantine Caramanis, Shie Mannor, and Sanjay Shakkottai. 2015. Local detection of infections in heterogeneous networks. *Proceedings - IEEE INFOCOM* 26 (2015), 1517–1525. https://doi.org/10.1109/INFOCOM.2015.7218530

[29] Seth Myers, Chenguang Zhu, and Jure Leskovec. 2012. Information Diffusion and External Influence in Networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD' 12)*. 33–41.

[30] Praneeth Netrapalli and Sujay Sanghavi. 2012. 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)*. 211–222. https://doi.org/10.1145/2318857.2254783 arXiv:1202.1779

[31] M. E. J. Newman. 2014. *Networks: An Introduction*. Vol. 23. 73–75 pages. https://doi.org/10.1017/S0963180113000479

[32] Devavrat Shah and Tauhid Zaman. 2010. Detecting sources of computer viruses in networks: theory and experiment. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 38. ACM, 203–214.

[33] Devavrat Shah and Tauhid Zaman. 2010. Rumors in a Network : Who ' s the Culprit ? *IEEE Transactions on information theory* 57, 8 (2010), 1–43. https://doi.org/10.1109/TIT.2011.2158885 arXiv:0909.4370

[34] Devavrat Shah and Tauhid Zaman. 2012. Rumor centrality: a universal source detector. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 40. ACM, 199–210.

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

[36] Zhaoxu Wang, Wenxiang Dong, Wenyi Zhang, and Chee Wei Tan. 2014. Rumor source detection with multiple observations: Fundamental limits and algorithms. In *ACM SIGMETRICS Performance Evaluation Review*, Vol. 42. ACM, 1–13.

[37] Liang Wu and Huan Liu. 2018. 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*. https://doi.org/10.1145/3159652.3159677

[38] Ali Zarezade, Ali Khodadadi, Mehrdad Farajtabar, Hamid R Rabiee, and Hongyuan Zha. 2017. Correlated Cascades : Compete or Cooperate. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. 238–244.

[39] Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. 2015. 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). https://doi.org/10.1145/2783258.2783401 arXiv:1506.02594

A BIDIRECTIONAL TREE

We include here the full calculations for learning the weights of the bidirectional tree.

Proposition A.1. *If we know (i, j) is an edge in the original tree, then the probability of infection along this edge is given by:*

$$p_{ij} = \frac{f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}{g_{i,\setminus j} \cdot (s_0^2 - s_2^2) + f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}.$$

Proof. According to Lemma 2.12, we have:

$$\begin{aligned}
f_{i<j} &= \mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} \cdot s_0 + \mathcal{P}_{\setminus i}(\to j) \cdot p_{ji} \cdot s_2 \\
g_{i,\setminus j} &= \mathcal{P}_{\setminus j}(\to i) \cdot (1 - p_{ij}) \\
f_{j<i} &= \mathcal{P}_{\setminus i}(\to j) \cdot p_{ji} \cdot s_0 + \mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} \cdot s_2 \\
g_{j,\setminus i} &= \mathcal{P}_{\setminus i}(\to j) \cdot (1 - p_{ji}).
\end{aligned}$$

We have 4 second-order equations, with 4 unknowns: $p_{ij}, p_{ji}, \mathcal{P}_{\setminus j}(\to i)$ and $\mathcal{P}_{\setminus i}(\to j)$. We solve it:

$$\begin{aligned}
f_{i<j} + f_{j<i} &= (\mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} + \mathcal{P}_{\setminus i}(\to j) \cdot p_{ji}) \cdot (s_0 + s_2) \\
f_{i<j} - f_{j<i} &= (\mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} - \mathcal{P}_{\setminus i}(\to j) \cdot p_{ji}) \cdot (s_0 - s_2) \\
\implies \mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} &= \frac{1}{2}\left(\frac{f_{i<j} + f_{j<i}}{s_0 + s_2} + \frac{f_{i<j} - f_{j<i}}{s_0 - s_2}\right) \\
&= \frac{f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}{s_0^2 - s_2^2} \\
\mathcal{P}_{\setminus j}(\to i) &= g_{i,\setminus j} + \mathcal{P}_{\setminus j}(\to i) \cdot p_{ij} \\
&= g_{i,\setminus j} + \frac{f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}{s_0^2 - s_2^2} \\
\implies p_{ij} &= \frac{\mathcal{P}_{\setminus j}(\to i) \cdot p_{ij}}{\mathcal{P}_{\setminus j}(\to i)} \\
p_{ij} &= \frac{f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}{g_{i,\setminus j} \cdot (s_0^2 - s_2^2) + f_{i<j} \cdot s_0 - f_{j<i} \cdot s_2}.
\end{aligned}$$

□

Lemma A.2. *With $M = \frac{N^2}{\epsilon^2} \log\left(\frac{6}{\delta}\right) \frac{\left((s_0^2 - s_2^2 + s_0 + s_2) p_{max} + s_0 + s_2\right)^2}{(s_0^2 - s_2^2)^2}$ samples, with probability at least $1 - \delta$, we have:*

$$|\hat{p}_{ij} - p_{ij}| \leq \epsilon.$$

Proof. Using Hoeffding's inequality:

$$\begin{aligned}
\mathbb{P}(|\hat{f}_{i<j} - f_{i<j}| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}, \\
\mathbb{P}(|\hat{f}_{j<i} - f_{j<i}| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}, \\
\mathbb{P}(|\hat{g}_{i,\setminus j} - g_{i,\setminus j}| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}.
\end{aligned}$$

Choosing $M = \frac{1}{\epsilon_1^2} \log\left(\frac{6}{\delta}\right)$, we have that with probability at least $1 - \delta$, all the following hold:

$$\begin{aligned}
|\hat{f}_{i<j} - f_{i<j}| &\leq \epsilon_1, \\
|\hat{f}_{j<i} - f_{j<i}| &\leq \epsilon_1,
\end{aligned}$$

$$|\hat{g}_{i,\backslash i} - g_{i,\backslash i}| \leq \epsilon_1.$$

Hence, with probability at least $1-\delta$, we have:

$$\begin{aligned}
\hat{p}_{ij} &= \frac{\hat{f}_{i<j}\cdot s_0 - \hat{f}_{j<i}\cdot s_2}{\hat{g}_{i,\backslash i}\cdot(s_0^2-s_2^2) + \hat{f}_{i<j}\cdot s_0 - \hat{f}_{j<i}\cdot s_2} \\
&\leq \frac{(f_{i<j}+\epsilon_1)\cdot s_0 - (f_{j<i}-\epsilon_1)\cdot s_2}{(g_{i,\backslash i}-\epsilon_1)\cdot(s_0^2-s_2^2) + (f_{i<j}-\epsilon_1)\cdot s_0 - (f_{j<i}+\epsilon_1)\cdot s_2} \\
&= \frac{f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2 + \epsilon_1(s_0+s_2)}{g_{i,\backslash i}\cdot(s_0^2-s_2^2) + f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2 - \epsilon_1(s_0^2-s_2^2+s_0+s_2)} \\
&= p_{ij}\frac{1}{1-\frac{\epsilon_1(s_0^2-s_2^2+s_0+s_2)}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2}} \\
&\quad + \epsilon_1\frac{s_0+s_2}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} + o(\epsilon_1) \\
&= p_{ij}\left(1+\frac{\epsilon_1(s_0^2-s_2^2+s_0+s_2)}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2}\right) \\
&\quad + \epsilon_1\frac{s_0+s_2}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} + o(\epsilon_1) \\
\hat{p}_{ij} - p_{ij} &\leq \epsilon_1\frac{(s_0^2-s_2^2+s_0+s_2)p_{max}}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} \\
&\quad + \epsilon_1\frac{s_0+s_2}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} + o(\epsilon_1).
\end{aligned}$$

Using the results from Lemma 2.12, we have:

$$\begin{aligned}
f_{i<j} &= \mathcal{P}_{\backslash i}(\rightarrow i)\cdot p_{ij}\cdot s_0 + \mathcal{P}_{\backslash j}(\rightarrow j)\cdot p_{ji}\cdot s_2, \\
g_{i,\backslash i} &= \mathcal{P}_{\backslash i}(\rightarrow i)\cdot(1-p_{ij}).
\end{aligned}$$

We use it to simplify the denominator:

$$\begin{aligned}
\text{denominator} &= g_{i,\backslash i}\cdot(s_0^2-s_2^2) + f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2 \\
&= \left(\mathcal{P}_{\backslash i}(\rightarrow i)\cdot(1-p_{ij})\right)\cdot(s_0^2-s_2^2) \\
&\quad + \left(\mathcal{P}_{\backslash i}(\rightarrow i)\cdot p_{ij}\cdot s_0 + \mathcal{P}_{\backslash j}(\rightarrow j)\cdot p_{ji}\cdot s_2\right)\cdot s_0 \\
&\quad - \left(\mathcal{P}_{\backslash j}(\rightarrow j)\cdot p_{ji}\cdot s_0 + \mathcal{P}_{\backslash i}(\rightarrow i)\cdot p_{ij}\cdot s_2\right)s_2 \\
&= \mathcal{P}_{\backslash i}(\rightarrow i)\cdot(s_0^2-s_2^2) \\
&\geq \frac{s_0^2-s_2^2}{N}.
\end{aligned}$$

Plugging back above:

$$\begin{aligned}
\hat{p}_{ij} - p_{ij} &\leq \epsilon_1\frac{(s_0^2-s_2^2+s_0+s_2)p_{max}}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} \\
&\quad + \epsilon_1\frac{s_0+s_2}{g_{i,\backslash i}\cdot(s_0^2-s_2^2)+f_{i<j}\cdot s_0 - f_{j<i}\cdot s_2} + o(\epsilon_1)
\end{aligned}$$

$$\leq \epsilon_1 N \frac{(s_0^2 - s_2^2 + s_0 + s_2)p_{max}}{s_0^2 - s_2^2} + \epsilon_1 N \frac{s_0 + s_2}{s_0^2 - s_2^2} + o(\epsilon_1).$$

By symmetry, we obtain:

$$|\hat{p}_{ij} - p_{ij}| \leq \epsilon_1 N \frac{(s_0^2 - s_2^2 + s_0 + s_2)p_{max} + s_0 + s_2}{s_0^2 - s_2^2} + o(\epsilon_1).$$

By choosing $\epsilon_1 = \frac{\epsilon}{N} \frac{s_0^2 - s_2^2}{(s_0^2 - s_2^2 + s_0 + s_2)p_{max} + s_0 + s_2}$, we therefore have:

With $M = \frac{N^2}{\epsilon^2} \log\left(\frac{6}{\delta}\right) \frac{\left((s_0^2 - s_2^2 + s_0 + s_2)p_{max} + s_0 + s_2\right)^2}{(s_0^2 - s_2^2)^2}$ samples, with probability at least $1 - \delta$, we have $|\hat{p}_{ij} - p_{ij}| \leq \epsilon$.

□

B BOUNDED-DEGREE GRAPHS

B.1 Solving the system

Lemma B.1. *Let* $\Delta = \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. *We have:*

$$\Delta \geq (s_0^2 - s_2^2)^2 \frac{(1 - p_{max})^2}{1 + p_{max}^2}.$$

Proof. Finding a lower bound for Δ can be achieved through minimizing Δ, or maximizing $(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. We want to solve:

$$\begin{aligned}
\underset{V_{ij}, V_{ji}}{\text{maximize}} \quad & (V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0) \\
\text{subject to} \quad & V_{ij} = \frac{p_{ij}s_0 + p_{ji}s_2}{1 + p_{ij}p_{ji}}, \\
& V_{ji} = \frac{p_{ji}s_0 + p_{ij}s_2}{1 + p_{ij}p_{ji}}, \\
& p_{ij} \geq 0, \\
& p_{max} - p_{ij} \geq 0, \\
& p_{ji} \geq 0, \\
& p_{max} - p_{ji} \geq 0.
\end{aligned}$$

To do so, we introduce Lagrangian multipliers. By replacing V_{ij} and V_{ji} with their actual value, the optimization problem above only has affine constraints, so it satisfies the linearity constraint qualification for the Karush-Kuhn-Tucker conditions. In other words, all the partial derivatives of the Lagrangian are equal to 0 for an optimal point.

$$\begin{aligned}
\mathcal{L} &= \mathcal{L}(V_{ij}, V_{ji}, p_{ij}, p_{ji}, \lambda_1, \lambda_2, \mu_1, \mu_2, \mu_3, \mu_4) \\
&= (V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0) \\
&\quad - \lambda_1(v_{ij}(1 + p_{ij}p_{ji}) - p_{ij}s_0 + p_{ji}s_2) \\
&\quad - \lambda_2(v_{ji}(1 + p_{ij}p_{ji}) - p_{ji}s_0 + p_{ij}s_2) \\
&\quad - \mu_1 p_{ij} - \mu_2(p_{max} - p_{ij}) \\
&\quad - \mu_3 p_{ji} - \mu_4(p_{max} - p_{ji}).
\end{aligned}$$

We calculate the gradients of $\mathcal{L}$.

$$\frac{\partial \mathcal{L}}{\partial V_{ij}} = V_{ji}(s_0^2 + s_2^2) - 2V_{ij}s_0s_2 - \lambda_1(1 + p_{ij}p_{ji}),$$

$$\frac{\partial \mathcal{L}}{\partial V_{ji}} = V_{ij}(s_0^2 + s_2^2) - 2V_{ji}s_0s_2 - \lambda_2(1 + p_{ij}p_{ji}),$$
$$\frac{\partial \mathcal{L}}{\partial p_{ij}} = -\lambda_1(V_{ij}p_{ji} - s_0) - \lambda_2(V_{ji}p_{ji} - s_2) - \mu_1 + \mu_2,$$
$$\frac{\partial \mathcal{L}}{\partial p_{ji}} = -\lambda_1(V_{ij}p_{ij} - s_2) - \lambda_2(V_{ji}p_{ij} - s_0) - \mu_3 + \mu_4.$$

From now on, we find the set X^0 of points for which all the partial derivatives are null. We know the solution of the maximization problem is the point of X^0 which maximizes the objective function.

Let us assume an interior point solution exists. For this point, all the gradients of $\mathcal{L}$ are equal to 0. Since it is an interior point, we also have $\mu_1 = \mu_2 = \mu_3 = \mu_4 = 0$ by complementary slackness. Solving this system, we obtain:

$$\lambda_1 = (s_0^2 - s_2^2)\frac{p_{ji}s_0 - p_{ij}s_2}{(1 + p_{ij}p_{ji})^2},$$
$$\lambda_2 = (s_0^2 - s_2^2)\frac{p_{ij}s_0 - p_{ji}s_2}{(1 + p_{ij}p_{ji})^2}.$$

Plugging this in above, the condition $\frac{\partial \mathcal{L}}{\partial p_{ij}} = 0$ becomes $p_{ij}p_{ji}(1-p_{ji}) = 0$. However, this is impossible for an interior point, since $0 < p_{ij}, p_{ji} < p_{max} < 1$. Therefore, the extrema of Δ are attained when at least one constraint is active.

We notice that if the conditions $p_{ij} = 0$ or $p_{ji} = 0$ are active, then $(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0) = 0$. Let us suppose (without loss of generality, by symmetry of the problem) that we have $p_{ij} = p_{max}$. The objective function is then increasing in p_{ji}. Therefore, Δ is minimized when $p_{ij} = p_{ji} = p_{max}$, which implies $v_{ij} = V_{ji} = \frac{p_{max}(s_0+s_2)}{1+p_{max}^2}$. In this case:

$$\begin{aligned}
\Delta &\geq (s_0^2 - s_2^2)^2 - 4V_{ij}^2(s_0 - s_2)^2 \\
&\geq (s_0^2 - s_2^2)^2 - 4\left(\frac{p_{max}(s_0 + s_2)}{1 + p_{max}^2}\right)^2 (s_0 - s_2)^2 \\
&\geq (s_0^2 - s_2^2)^2 \left[1 - 4\frac{p_{max}}{1 + p_{max}^2}\right] \\
&\geq (s_0^2 - s_2^2)^2 \frac{(1 - p_{max})^2}{1 + p_{max}^2}.
\end{aligned}$$

This expression is always positive, which is what we wanted.

□

Theorem B.2. *For any graph, for any noise distribution having finite values, we can learn the weights of all the edges of the graph. In particular, we can compute a quantity which converges to the true weight of each edge:*

$$\hat{p}_{ij} = \frac{2(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)}{\left(s_0^2 - s_2^2\right) + \sqrt{\left(s_0^2 - s_2^2\right)^2 - 4(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)(\hat{V}_{ij}s_2 - \hat{V}_{ji}s_0)}}.$$

Proof.

$$\begin{aligned}
\hat{V}_{ij} &\to_{M\to\infty} V_{ij} \\
&:= \frac{p_{ij} \cdot s_0 + p_{ji} \cdot s_2}{1 + p_{ij} \cdot p_{ji}}
\end{aligned}$$

$$p_{ji} = \frac{V_{ij} - p_{ij} \cdot s_0}{s_2 - V_{ij} \cdot p_{ij}}$$

We can plug this in V_{ji}:

$$V_{ji} = \frac{p_{ji} \cdot s_0 + p_{ij} \cdot s_2}{1 + p_{ij} \cdot p_{ji}}$$

$$V_{ji}\left[1 + p_{ij} \cdot \frac{V_{ij} - p_{ij} \cdot s_0}{s_2 - V_{ij} \cdot p_{ij}}\right] = \frac{V_{ij} - p_{ij} \cdot s_0}{s_2 - V_{ij} \cdot p_{ij}} \cdot s_0 + p_{ij} \cdot s_2$$

After some shuffling around, we obtain the second-degree equation:

$$V_{ji}s_2 - V_{ij}s_0 + \left(s_0^2 - s_2^2\right) p_{ij} + \left(V_{ij}s_2 - V_{ji}s_0\right) p_{ij}^2 = 0$$

We recall that by definition, $s_0 \geq s_2$. We also notice that if $p_{ij} = q_1$ and $p_{ji} = q_2$ is a pair of solutions of this system, then $p_{ij} = \frac{1}{q_2}$ and $p_{ji} = \frac{1}{q_1}$ forms the other pair of solution, which implies there is uniqueness of solutions in $[0, p_{max}]$. Since the real probabilities of infection satisfy this system, we also know the solution exists. Let $\Delta = \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)$. The only solution of this system in $[0, p_{max}]$ is:

$$\begin{aligned}
p_{ij} &= \frac{-\left(s_0^2 - s_2^2\right) + \sqrt{\Delta}}{2(V_{ij}s_2 - V_{ji}s_0)} \\
&= \frac{\left(s_0^2 - s_2^2\right)^2 - \left(s_0^2 - s_2^2\right)^2 - 4(V_{ji}s_2 - V_{ij}s_0)(V_{ij}s_2 - V_{ji}s_0)}{2(V_{ji}s_0 - V_{ij}s_2)\left(\left(s_0^2 - s_2^2\right) + \sqrt{\Delta}\right)} \\
&= \frac{2(V_{ji}s_2 - V_{ij}s_0)}{\left(s_0^2 - s_2^2\right) + \sqrt{\Delta}}
\end{aligned}$$

□

B.2 Sample complexity

Proposition B.3. *As the number of cascades M goes to infinity, the estimators below tend to the following limit:*

$$\hat{h}^2_{i,j} \to_{M\to\infty} \frac{1}{N}(p_{ij} + p_{ji}) \prod_{k\neq i,j}(1 - p_{ik})(1 - p_{jk})$$

$$\hat{f}^2_{i<j} \to_{M\to\infty} \frac{1}{N}(p_{ij} \cdot s_0 + p_{ji} \cdot s_2) \prod_{k\neq i,j}(1 - p_{ik})(1 - p_{jk})$$

$$\hat{e}^1_i \to_{M\to\infty} \frac{1}{N}(1 - p_{ij}) \prod_{k\neq i,j}(1 - p_{ik})$$

Proof. Using the law of large numbers:

$$\begin{aligned}
\hat{f}^2_{i<j} &\to_{M\to\infty} \mathbb{E}[\hat{f}^2_{i<j}] \\
&= \mathbb{P}(i \text{ source, infects } j, \text{ no other infections, delay } 0) \\
&\quad + \mathbb{P}(j \text{ source, infects } i, \text{ no other infections, delay } 2) \\
&= \frac{1}{N} p_{ij} \prod_{k\neq i,j}(1 - p_{ik}) \prod_{k\neq i,j}(1 - p_{jk}) \cdot s_0 \\
&\quad + \frac{1}{N} p_{ji} \prod_{k\neq i,j}(1 - p_{jk}) \prod_{k\neq i,j}(1 - p_{ik}) \cdot s_2
\end{aligned}$$

$$= \frac{1}{N}(p_{ij} \cdot s_0 + p_{ji} \cdot s_2) \prod_{k \neq i,j} (1 - p_{ik})(1 - p_{jk}).$$

In the same vein, we have:

$$\begin{aligned}
\hat{h}^2_{i,j} &\to_{M \to \infty} \mathbb{E}[\hat{h}^2_{i,j}] \\
&= \mathbb{P}(i \text{ source, infects } j, \text{ no other infections}) \\
&\quad + \mathbb{P}(j \text{ source, infects } i, \text{ no other infections}) \\
&= \frac{1}{N}(p_{ij} + p_{ji}) \prod_{k \neq i,j} (1 - p_{ik})(1 - p_{jk}) \\
\hat{e}^1_i &\to_{M \to \infty} \mathbb{E}[\hat{e}^1_i] \\
&= \mathbb{P}(i \text{ source, no other infections}) \\
&= \frac{1}{N} \prod_{k \neq i} (1 - p_{ik}) \\
&= \frac{1}{N}(1 - p_{ij}) \prod_{k \neq i,j} (1 - p_{ik}).
\end{aligned}$$

□

Proposition B.4. *With probability* $1 - \frac{\delta}{N^2}$, *with* $M =$ *samples, we can estimate* V_{ij} *with precision* ϵ_V.

Proof. As in Proposition 2.14, we use Hoeffding's inequality:

$$\begin{aligned}
\mathbb{P}(|\hat{f}^2_{i<j} - f^2_{i<j}| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}, \\
\mathbb{P}(|\hat{h}^2_{i,j} - h^2_{i,j}| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}, \\
\mathbb{P}(|\hat{e}^1_i - e^1_i| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}, \\
\mathbb{P}(|\hat{e}^1_j - e^1_j| > \epsilon_1) &\leq 2e^{-2M\epsilon_1^2}.
\end{aligned}$$

We use this to bound above V_{ij}:

$$\begin{aligned}
\hat{V}_{ij} &= \frac{\hat{f}^2_{i<j}}{\hat{h}^2_{i,j} + N \cdot \hat{e}^1_i \cdot \hat{e}^1_j} \\
&\leq \frac{f^2_{i<j} + \epsilon_1}{h^2_{i,j} + \epsilon_1 + N \cdot (e^1_i + \epsilon_1) \cdot (e^1_j + \epsilon_1)} \\
&= V_{ij} \frac{1 + \frac{\epsilon_1}{f^2_{i<j}}}{1 + \frac{\epsilon_1 + N\epsilon_1(e^1_i + e^1_j)}{h^2_{i,j} + Ne^1_i e^1_j}} \\
&= V_{ij} \left[1 + \frac{\epsilon_1}{f^2_{i<j}} + \epsilon_1 \frac{1 + N(e^1_i + e^1_j)}{h^2_{i,j} + Ne^1_i e^1_j} + o(\epsilon_1) \right]
\end{aligned}$$

We bound below the denominators:

$$\begin{aligned}
f^2_{i<j} &\geq \frac{1}{N}(p_{ij}s_0 + p_{ji}s_2)(1 - p_{max})^{2d} \\
&\geq \frac{p_{min}s_2}{N}(1 - p_{max})^{2d}
\end{aligned}$$

$$h^2_{i,j} + Ne^1_i e^1_j \geq \frac{1}{N}(1 + p_{ij}p_{ji})(1 - p_{max})^{2d}$$
$$\geq \frac{1}{N}(1 - p_{max})^{2d}$$

We bound above the numerator:

$$N(e^1_i + e^1_i) = N\left(\frac{1}{N}\prod_{k\neq i}(1 - p_{ik}) + \frac{1}{N}\prod_{k\neq j}(1 - p_{jk})\right)$$
$$\leq N\left(\frac{1}{N} + \frac{1}{N}\right)$$
$$\leq 2.$$

Plugging in above:

$$\hat{V}_{ij} = V_{ij}\left[1 + \frac{\epsilon_1}{f^2_{i<j}} + \epsilon_1\frac{1 + N(e^1_i + e^1_j)}{h^2_{i,j} + Ne^1_i e^1_j} + o(\epsilon_1)\right]$$
$$\leq V_{ij}\left[1 + \frac{\epsilon_1}{\frac{p_{min}s_2}{N}(1 - p_{max})^{2d}} + \frac{\epsilon_1(1 + 2)}{\frac{1}{N}(1 - p_{max})^{2d}} + o(\epsilon_1)\right].$$

Using $V_{ij} \leq 1$, and by symmetry:

$$|\hat{V}_{ij} - V_{ij}| = \epsilon_1\frac{N}{(1 - p_{max})^{2d}}\left[\frac{1}{p_{min}s_2} + 3\right] + o(\epsilon_1)$$
$$=\leq \epsilon_1\frac{N}{(1 - p_{max})^{2d}}\left[\frac{1 + 3p_{min}s_2}{p_{min}s_2}\right] + o(\epsilon_1)$$
$$\leq \epsilon_1\frac{4N}{p_{min}s_2(1 - p_{max})^{2d}} + o(\epsilon_1).$$

Therefore, by union bound, and by choosing $\epsilon_1\frac{4N}{p_{min}s_2(1-p_{max})^{2d}} = \epsilon_V$, and setting $2e^{-2M\epsilon_1^2} = \frac{\delta}{3N^2}$, we obtain:
With $M = \frac{1}{\epsilon_V^2}\frac{16N^2}{p^2_{min}s^2_2(1-p_{max})^{4d}}\frac{2\log(3N)-\log(\delta)}{2}$ samples, we can guarantee $|\hat{V}_{ij} - V_{ij}| \leq \epsilon_V$ with probability at least $1 - \frac{\delta}{N^2}$. □

Proposition B.5. *Assuming we can estimate V_{ij} within precision ϵ_V, then we can estimate p_{ij} within precision $\epsilon = \frac{6\epsilon_V(1+p^2_{max})}{(s_0^2-s_2^2)^2(1-p_{max})^2}$.*

Proof. If we know $|\hat{V}_{ij} - V_{ij}| < \epsilon_V$ and $|\hat{V}_{ji} - V_{ji}| < \epsilon_V$:

$$\hat{p}_{ij} = \frac{2(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)}{\left(s_0^2 - s_2^2\right) + \sqrt{\left(s_0^2 - s_2^2\right)^2 - 4(\hat{V}_{ji}s_2 - \hat{V}_{ij}s_0)(\hat{V}_{ij}s_2 - \hat{V}_{ji}s_0)}}$$
$$\leq \frac{2(V_{ji}s_2 - V_{ij}s_0) + 2\epsilon_V(s_0 + s_2)}{\left(s_0^2 - s_2^2\right)^2 + \sqrt{\Delta - 4\epsilon_V(s_0 + s_2)^2(V_{ij} + V_{ji})}}.$$

We recall $s_0 + s_2 \leq 1$, $V_{ij} \leq 1$, $p_{ij} \leq 1$, and $1 + p^2_{max} \leq \Delta \leq 1$ (Lemma 3.12). Hence:

$$\hat{p}_{ij} \leq \frac{2(V_{ji}s_2 - V_{ij}s_0) + 2\epsilon_V}{\left(s_0^2 - s_2^2\right)^2 + \sqrt{\Delta}\sqrt{1 - 8\frac{\epsilon_V}{\Delta}}}$$

$$\leq p_{ij}\left(1+\frac{4\epsilon_V}{\sqrt{\Delta}\left(\left(s_0^2-s_2^2\right)^2+\sqrt{\Delta}\right)}\right)+\frac{2\epsilon_V}{\left(s_0^2-s_2^2\right)^2+\sqrt{\Delta}}+o(\epsilon_V)$$
$$\leq p_{ij}+\frac{6\epsilon_V}{\Delta}.$$

By symmetry, and using the bound on Δ stated above, we conclude that if we know V_{ij} and V_{ji} up to precision ϵ_V, we know p_{ij} up to precision $\epsilon = \frac{6\epsilon_V(1+p_{max}^2)}{(s_0^2-s_2^2)^2(1-p_{max})^2}$.

□

Theorem B.6. *In the limited-noise setting, with probability at least* $1-\delta$*, with* $M = O\left(\frac{e^{4p_{max}(d+1)}}{p_{min}^2 s_2^2(s_0^2-s_2^2)^4}\frac{N^2}{\epsilon^2}\log\left(\frac{N}{\delta}\right)\right)$ *samples, we can learn the weights of any bounded-degree graph up to precision epsilon.*

Proof. With probability at least $1-\frac{\delta}{N^2}$, using $M = \frac{1}{\epsilon_V^2}\frac{16N^2}{p_{min}^2 s_2^2(1-p_{max})^{4d}}\frac{2\log(3N)-\log(\delta)}{2}$ samples, we can guarantee $|\hat{V}_{ij}-V_{ij}|\leq\epsilon_V$ with probability at least $1-\frac{\delta}{N^2}$ samples, knowing $\frac{1}{\epsilon_V} = \frac{6(1+p_{max}^2)}{\epsilon(s_0^2-s_2^2)^2(1-p_{max})^2}$. This gives us a sample complexity of:

$$\begin{aligned}M &\geq \frac{18}{\epsilon^2(s_0^2-s_2^2)^4(1-p_{max})^{4(d+1)}}N^2\left[\frac{4}{p_{min}s_2}\right]^2\log\left(\frac{9N^2}{\delta}\right)\\ &\geq \frac{1152\cdot e^{4p_{max}(d+1)}}{p_{min}^2 s_2^2(s_0^2-s_2^2)^4}\frac{N^2}{\epsilon^2}\log\left(\frac{9N^2}{\delta}\right)\\ &= O\left(\frac{e^{4p_{max}(d+1)}}{p_{min}^2 s_2^2(s_0^2-s_2^2)^4}\frac{N^2}{\epsilon^2}\log\left(\frac{N}{\delta}\right)\right).\end{aligned}$$

□

C NUMERICAL EXPERIMENTS

In this section, we provide some numerical experiments on simulated data to investigate the sample complexity of the method introduced in Section 3.2. While Theorem 3.14 guarantees polynomial dependence on the key scaling parameters of the problem (except for the dependence on the degree, which in many settings may be counteracted by the small value of $p_{\max}$, as indicated by Corollary 2.6), the number of cascades required can still scale to numbers that would not be feasible in practice. Our experiments here show two things. First, they indicate that our algorithm in fact outperforms the theoretical guarantees of Theorem 3.14. Specifically illustrating this, we consider the following setting: a graph of max-degree 10 on 1000 nodes, with Bernoulli noise taking values 0 or 2 with probability $\frac{1}{2}$. Then, if we want to learn the weights within precision $\epsilon = 0.1$, our bound guarantees that 10^{20} cascades, are sufficient. Our simulations show, however, that this is likely to be achieved with about 10^8 cascades.

Next, we consider the impact of considering a different objective. Our results provide guarantees on the *maximum error*. It may, however, be enough for practical applications, to have control of the *average absolute error*. While our theoretical bounds do not give sample complexity bounds for guarantees on this objective, our simulations demonstrate that it is possible to achieve control of the average absolute error with far fewer samples. Figure 5 gives a plot of average absolute error versus sample complexity, for the same numbers mentioned above. We see, in particular, that 0.1 average absolute error is achieved in about $3*10^5$ cascades, thus representing a very significant decrease.

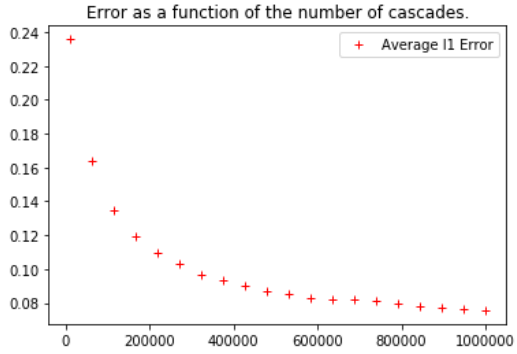


Fig. 5. Average absolute error for a random graph of max-degree d on 1000 nodes, with Bernoulli noise taking values 0 or 2 with probability $\frac{1}{2}$. The average is taken only on the existing edges to demonstrate the efficiency of the procedure.

We note that individually, the terms in our bound do not have much room for improvement in theory. As we have speculated, it may be possible to improve N^2 to $N \log N$. However, the message from these experiments is that in order to transition these theoretical results to a practical setting, it is in fact important to consider the specific dependence on each parameter. Moreover, it may be important to consider obtaining guarantees for objectives like average absolute error, for which far better results may be possible.

ACKNOWLEDGMENTS

The authors would like to thank Taylor Kessler Faulkner and Surbhi Goel for bouncing ideas, and their pointers on making this paper more enjoyable to read.

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

Received February 2019; revised March 2019; accepted April 2019