RI14004, 13 February 2014 Computer Science

IBM Research Report

A Bayesian Framework for Estimating Properties of Network Diffusions

Satya R. K. Pasumarthi
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
sarpasum@in.ibm.com

Varun R. Embar
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
varembar@in.ibm.com

Indrajit Bhattacharya
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
indrajitb@in.ibm.com

IBM Research Division
Almaden - Austin - Beijing - Delhi - Haifa - T.J. Watson - Tokyo - Zurich

A Bayesian Framework for Estimating Properties of Network Diffusions

Satya R. K. Pasumarthi
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
sarpasum@in.ibm.com

Varun R. Embar
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
varembar@in.ibm.com

Indrajit Bhattacharya
IBM Research Division
IBM India Research Lab
Bangalore 560045, India
indrajitb@in.ibm.com

13 Feb 2014

Abstract

The analysis of network connections, diffusion processes and cascades is of practical and academic interest across many disciplines. Many problems in this analysis involve evaluating properties of the diffusion network. However, these properties often involve variables that are not explicitly observed in real world diffusions, such as the network connection strengths and the diffusion paths of infections over the network. These hidden variables therefore need to be estimated for these properties to be evaluated. In this paper, we propose and study this novel problem in a Bayesian framework by capturing the posterior distribution of these hidden variables given the observed cascades, and computing the expectation of these properties under this posterior distribution. We identify and characterize interesting network diffusion properties whose expectations can be computed exactly and efficiently, either wholly or in part. For properties that are not 'nice' in this sense, we propose a Gibbs Sampling framework for Monte-Carlo integration. In detailed experiments using various network diffusion properties over multiple synthetic and real datasets, we demonstrate that the proposed approach is significantly more accurate than a frequentist plug-in baseline. We also propose a map-reduce implementation of our framework and demonstrate that this scales easily for large datasets.

1 Introduction

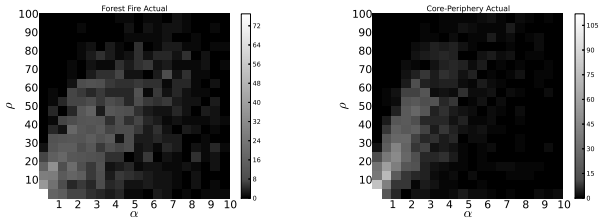
The study of networks and diffusions over them has a long history in epidemiology, sociology, econometrics and marketing. Interest in the problem has increased many fold over the last two decades in the context of information diffusion and social networks, first because of the growth of the internet, and then the social media revolution [2, 13]. The study typically involves three different objects of interest: a *network* that defines strengths of connection between entities, a stochastic *diffusion process* that defines how 'infections' diffuse over the network, and *cascades* tracing the diffusion of specific infections over the network. Many different problems have been studied in the context of these three objects of interest. A problem that has received a lot of attention is that of network inference [21, 7, 6, 8, 4, 9, 19, 22, 15], where the task is to infer the hidden network of connection strengths from the cascades, assuming a diffusion process.

However, inferring the network of diffusions is often an intermediate task in the analysis. The main objective is often to compute some *property* of the network and/or the cascades, such as centrality and reach of individual nodes, and optimal seeds for viral marketing [14, 12, 10], community structures [17, 1], the likelier diffusion mechanism [18], etc.

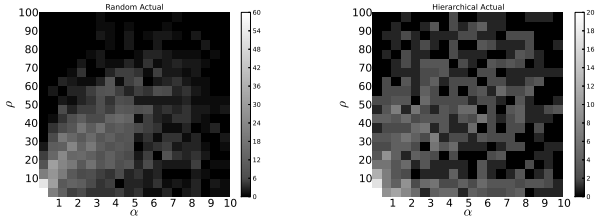
Goyal et. al. [10] propose the problem of finding 'tribe leaders', who are well connected to a large tribe of nodes in the network, and whose tribe nodes follow their actions frequently in the cascades. While finding and counting such tribe leaders is a computationally expensive property, consider a simplification of this definition. Imagine we wish to find (and count) influential leaders, where the influence of a leader is measured by his out-degree in the network, where an edge is counted in the degree only if it is strong and frequently used in the cascades. This influence score is much simpler to compute given completely observed networks and cascades, and yet is useful for marketers and epidemiologists.

In Fig. 1, we show the strength-frequency distribution of edges in four different synthetically-generated network diffusions, where edge strength (α) is the x-axis and transmission frequency ρ is y. These correspond to Forest Fire, Core-Periphery, Random and Hierarchical graphs respectively, each with 1024 nodes and ~ 2000 edges. In each case, we generated 20 splitting, independent cascades [22] on top these graphs with 2 randomly chosen seeds for each cascade. The distribution only considers actual edges used in the cascades. An alternative interpretation is that these show the summed influence score (defined above) of all users in a specific $\alpha - \rho$ region. We are not aware of any earlier investigation of such strength-frequency distributions for network diffusions. The plots clearly show that these distributions look very different depending on the underlying network connections and possibly also the diffusion mechanism. Thus, given network diffusion data from some network with unknown structure and diffusion mechanism, it is clearly of interest to construct and study such distributions.

In this paper, we investigate such joint properties of networks and cascades. The main difficulty in evaluating such properties for real-world network diffusions is that the connections strengths in the network are unknown. Addition-



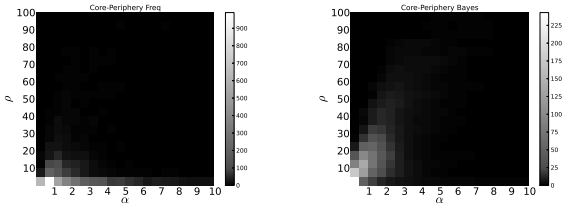
(a) Forest fire (FF) (b) Core-Periphery (CP)



(c) Random (Rnd) (d) Hierarchical (HI)

Figure 1: Edge distribution for cascades from different synthetic graphs

ally, the cascades only record the catchers of the infections and the infection times, but not the actual path traced by specific infections. For example, in social information flows, the friends and followers are known, but not the extent of influence between them, and most often users report information without revealing their sources. Therefore, to evaluate the properties, these hidden aspects need to be inferred from the observed cascades.



(a) Frequentist (b) Bayesian Expectation

Figure 2: CP: Inferred Edge distribution

One possible way for reconstructing a property is to take the 'frequentist plug-in approach', that finds a point estimate of the network given the cascade, and also of the diffusion paths followed in the cascades, and then uses only these point estimates for computing the property. The most popular point estimate used for network inference is the maximum likelihood estimate [6, 22]. This solution suffers from two different drawbacks. The first is the well known problem of overfitting for a frequentist approach. More importantly, for properties that

are not one-to-one functions of the network and the diffusion paths, the most likely value of the property need not correspond to the most likely network and diffusion paths. As an example, Fig. 2(a) shows the reconstructed edge distribution corresponding to the Core-Periphery diffusion data. It has failed to recovering the signature shape of the distribution.

In this paper, we motivate and propose a Bayesian solution to this problem, where both the network and the diffusion paths are modeled as random variables. Then the network diffusion properties are also random variables, and our problem becomes one of computing the expectation of the property under the posterior distribution of the hidden variables given the observed features of the cascade.

An obvious challenge for the Bayesian approach is the cost of computing expectations, which seem daunting for the network inference problem with its large number of coupled discrete and continuous hidden variables. However, our analysis shows that, for the popular independent cascade model, many interesting network diffusion properties are 'nice', in that their expectations can be computed exactly and efficiently, at least in part. For parts of the expectations that are not 'nice', we propose a Gibbs Sampling technique for efficient Monte Carlo approximation. Fig. 2(b) shows the reconstruction of the Core-Periphery edge distribution using our proposed approach. Clearly, it has been able to recover the distinctive form to a much better extent.

In detailed experiments using various network diffusion properties over multiple synthetic and real datasets, we demonstrate that the proposed approach is significantly more accurate than the MLE plug-in baseline. We show that using a map-reduce implementation the approach scales easily to very large datasets.

Our main contributions are as follows. (A) We propose a new problem which we believe has wide application and has not been studied in this generality before. (B) We propose a Bayesian solution framework, and characterize network diffusion properties that are efficiently computable within this framework. We show that the Bayesian framework is very effective for traditional network inference as well. (C) We perform detailed experiments that demonstrate the effectiveness of our solution for real and synthetic datasets, and also its scalability for large datasets.

2 Related Work

Different problems have been studied in the context of diffusion networks [2, 13]. The network inference problem [21, 7, 6, 8, 4, 9, 19, 22, 15] has been investigated in depth, starting with stationary discrete time models [7], to the more recent models that consider features [22] and time-varying networks [9]. The approaches have mostly been based on maximum-likelihood estimation.

Apart from inferring the complete network structure, there has been work on inferring summaries of the network, such as community structures [17, 1]. Other investigated properties are estimating influence of nodes [3], and subsequently selecting a subset of nodes that maximize influence [12]. Sadikov et. al. [20]

study various properties of cascades assuming completely missing infections.

Milling et. al. [18] study the problem of deciding which of two given networks caused a specific diffusion with its path properties observed. Efficient algorithms have been designed for identifying leaders and tribes [10] and mining propagation summaries [16] from cascades, assuming the underlying network and the diffusion paths to be known. These problems may be seen as computing joint properties of networks and cascades, with all variables observed.

In summary, we are not aware of any general framework for estimating joint properties of networks and diffusion processes in the context of hidden network and diffusions paths. We are also not aware of any Bayesian framework for network diffusion analysis.

3 Background & Problem Definition

In this section, we first review the network diffusion setting and the independent cascade model and define network diffusion properties and the problem of computing expectations of such properties.

Network Diffusion and Independent Cascade Model : We assume a *network* $G = (V, E)$ with nodes V and edges E. For $(u, v) \in E$, let $\alpha_{uv} \in \mathcal{R}_+$ denote the connection strength between nodes u and v. We have a set C of *cascades* corresponding to spreading infections over the network $G = (V, E)$. Each cascade $c \in C$ consists of a set of time-stamped infections: $c = \{(u_i, z_i, t_i)\}$, where $u_i \in V$, $z_i \in 1 \dots i-1$, $t_i \in \mathcal{R}_+$ and $t_i < t_j$ for $i < j$. The i^{th} infection records that node u_i got infected at time t_i by its parent infection z_i. Let π_i denote the set of 'potential parents' for the i^{th} infection, so that $z_i \in \pi_i$. Observe that using knowledge of the infecting parent z_i for all infections in the cascade, it is possible to uniquely reconstruct the path of the diffusing infection over the network.

The joint distribution $p(C|\alpha)$ on the cascades C given the network strengths is typically defined using a *generative process* that captures the dynamics of spreading infections. While many diffusion models have been proposed, we follow the popular *Continuous Time Independent Cascade Model* [6]. Under this model, cascades are generated in an *iid* fashion. Each cascade starts with an initial set of seed nodes getting infected. Then at any time, each currently uninfected node has non-zero probability of getting infected by its currently infected neighbors in the network. A node gets infected when its first potential parent infects it. We consider the setting where nodes can get infected multiple times in the same cascade, and the *splitting model* for this [22], where all infections between the current and the previous infections of a node are considered as its potential parents.

The main building block of the model is the probability density function $f(t_i|u_i, u_j, t_j, \alpha_{ji})$, which models the conditional likelihood of node u_i getting infected at time t_i by node u_j which got infected at time t_j for $t_j < t_i$. The likelihood of a cascade c with observed parent information z looks as follows

[6, 22]:

$$p(c|\alpha) \quad = \prod_i H(t_i|t_{z_i};\alpha_{z_i i}) \prod_{j\in\pi_i} S(t_i|t_j;\alpha_{z_j i}) \tag{1}$$

where $S(t) = 1 - F(t)$ is the survival function, and $H(t) = f(t)/S(t)$ is the hazard function corresponding to CDF $F(t) = \int_0^t f(t)dt$. The likelihood of the set of cascades C is given by the products of the likelihoods of the individual cascades: $p(C|\alpha) = \prod_{c\in C} p(c|\alpha)$.

For most real-world network diffusions, many of the variables above are unobserved. We will assume that the observed trace $C^o = \{c^o\}$ of the cascade C only contains the infected node u_i and the infection time t_i: $c^o = \{(u_i, t_i)\}$. Specifically, the identify of the infecting parent z_i is not observed. The posterior distribution $p(z|\{c^o\}, \alpha)$ over infection parents, conditioned on observed cascades $\{c^o\}$ and α, has the following form:

$$p(z \mid \{c^o\}, \alpha) = \prod_i \frac{H(t_i|t_{z_i};\alpha_{u_{z_i}u_i})}{\sum_{j\in\pi_i} H(t_i|t_j;\alpha_{u_j u_i})} \tag{2}$$

Observe that this decouples into terms involving individual infection parents z_i. This will be a key property for efficient computation of network diffusion properties in Sec. 5.

The network connection strengths α_{uv} are also typically unobserved. Additionally, we will assume that set of network edges E is also not known. Therefore, we will consider α to be a $|V| \times |V|$ matrix of unknown variables. The goal of the popular network inference problem is to reconstruct this α matrix using $\{c^o\}$ [6, 22]. The state-of-the-art approach is to obtain a maximum likelihood estimate:

$$\hat{\alpha} = \arg\max_\alpha \log p(\{c^o\}|\alpha) = \arg\max_\alpha \log \sum_z p(C|\alpha) \tag{3}$$

The Exponential, Power-law and Rayleigh distributions have been proposed for $f(t_i|u_i, u_j, t_j, \alpha_{ji})$ [6, 22]. For the Exponential distribution,

$$f(t_i|t_j;\alpha_{ji}) = \alpha_{ji} e^{-\alpha_{ji}(t_i - t_j)}$$
$$H(t_i|t_j) = \alpha_{ji}; \quad S(ti|t_j) = e^{-\alpha_{ji}(t_i - t_j)}$$

and for the Rayleigh distribution,

$$f(t_i|t_j;\alpha_{ji}) = \alpha_{ji}(t_i - t_j) e^{-\frac{1}{2}\alpha_{ji}(t_i - t_j)^2}$$
$$H(t_i|t_j) = \alpha_{ji}(t_i - t_j); \quad S(t_i|t_j) = e^{-\frac{1}{2}\alpha_{ji}(t_i - t_j)^2}$$

Network Diffusion Properties and Expectations: Given this background, we now define our problem. We are interested in computing properties $f(C, G)$ of the cascades C and the network G. The properties may be binary or real-valued, scalars, vectors or even matrices. Consider as examples strength-frequency distribution of edges, or influence of leader nodes. We will see more examples in Sec. 5.

The main difficulty is that for most real-world network diffusions α and z are unobserved, so that the functions are not directly computable. We investigate a fully Bayesian solution to the problem, where we imagine both α and z to be random variables, so that the property $f(\alpha, z)$ is also a random variable. Further assuming a joint distribution $p(C, \alpha)$ to be defined on the cascade C and the network connections strengths α, we consider the posterior distribution $p(z, \alpha|\{c^o\})$ over the hidden variables z and α conditioned on the observed trace $c^o = \{(u_i, t_i)\}$ of the cascades. Then we consider its expectation $\bar{f}(C, \alpha)$ of $f(C, \alpha)$ under this posterior distribution:

$$\bar{f}(C, \alpha) = E_{p(z,\alpha|\{c^o\})}[f(C, \alpha)] \tag{4}$$

For properties that do not involve z, we consider the expectation under the marginal posterior distribution $p(\alpha|\{c^o\}) = \sum_z p(z, \alpha|\{c^o\})$. We similarly define expectations of properties that do not involve α.

Recall that existing approaches only model the conditional distribution $p(C|\alpha)$ assuming α to be given. In the rest of this paper, our goal is two fold: (a) augment this conditional to model the joint distribution $p(C, \alpha)$ using a Bayesian framework, (b) investigate tractability of this expectation for interesting network diffusion properties. We look at the first aspect in Sec. 4 and the second in Sec. 5.

4 A Bayesian Framework

In this section, we define a Bayesian framework for network diffusion analysis that will enable us to compute expectations of network diffusion properties. For a Bayesian analysis, we need to model α as a random variable, with a prior distribution and a posterior distribution. Assuming a *iid* prior $p(\alpha) = \prod_{uv} p(\alpha_{uv})$, the joint distribution would simply be $p(C, \alpha) = p(C|\alpha) \prod_{uv} p(\alpha_{uv})$, so that the posterior distribution $p(\alpha|\{c^o\}, z)$ looks as follows:

$$p(\alpha \mid \{c^o\}, z) = \prod_{uv} \frac{\bar{H}_{uv} \bar{S}_{uv} p(\alpha_{uv})}{\int_{\alpha_{uv}} \bar{H}_{uv} \bar{S}_{uv} p(\alpha_{uv}) d\alpha_{uv}} \tag{5}$$

where $\bar{H}_{uv} = \prod_{i \in A_{uv}} H(t_i|t_{z_i}; \alpha_{uv})$ and
$\bar{S}_{uv} = \prod_{i,j \in P_{uv}} S(t_i|t_j; \alpha_{uv}) \prod_{j \in T_{uv}} S(T|t_j; \alpha_{uv})$, where $A_{uv} = \{i : u_i = v, u_{z_i} = u\}$ denotes actual infections of v by u, $P_{uv} = \{i, j : u_i = u, u_j = v; j \in \pi_i\}$ denotes potential infections of u by v, $T_{uv} = \{j : u_j = u, l_v < t_j\}$ denotes survivals of v from u, l_v is the time of last infection of node v, T the final time stamp in the cascades. Observe that this decouples into terms involving individual network strengths α_{uv}. Efficient computation of network properties in Sec. 5 hinges critically on this, as on the decoupling in Eqn. 2.

Another requirement for us is analytical integration of network properties with respect to α_{uv}. For this, it is convenient to consider *conjugate priors*. Both Rayleigh and Exponential are special cases of the Weibull distribution (corresponding to shape parameters 1 and 2) [3]. For likelihoods involving the

Weibull distribution with given shape parameter, the conjugate distribution is the *Gamma distribution*:

$$Gamma(\alpha_{uv}; a, b) = \frac{b^a}{\Gamma(a)} \alpha_{uv}^{a-1} \exp\{-b\alpha_{uv}\} \tag{6}$$

Substitution into Eqn. 5 gives us the following:

$$p(\alpha|\{c^o\}, z) = \prod_{uv} Gamma(a + \rho(u,v), b + \Delta_{uv}) \tag{7}$$

where $\rho_{uv} = |A_{uv}|$, $\Delta_{uv} = \sum_{i,j_\in P_{uv}} \delta_{ij} + \sum_{j \in T_{uv}} (\mathbf{T} - t_j)$, and $\delta_{ij} = (t_i - t_j)$ for the Exponential distribution and $\frac{1}{2}(t_i - t_j)^2$ for the Rayleigh distribution.

We observe that this posterior is very suitable for the network inference problem. Consider $a < 1$. Then for no transmissions across as edge, $\rho(u,v) = 0$, and the posterior is the same as the prior distribution $Gamma(a, b)$, which is peaked sharply around 0. This implies that in the absence of any transmission evidence in the cascade, there is very little belief in the existence of an edge. Once an observation is made and we have $\rho(u,v) \geq 1$, the posterior distribution is unimodal and peaked at $(a + \rho(u,v))/(b + \Delta_{uv})$. This lies between 0 and the MLE, which is $\rho(u,v)/\Delta_{uv}$. When we have large volumes of data so that $\rho(u,v) \gg a$ and $\Delta_{uv} \gg b$, the mean of the posterior approaches the MLE. While the parameterization $a < 1$ models prior belief in sparse network connections, it is also possible to make the Gamma prior noninformative if necessary, using $a, b \ll 1$ [5].

5 Network Diffusion Properties

In this section, we consider the multiple types of network diffusion properties, and analyze the tractability of computing their expectations under the posterior distribution $p(z, \alpha|\{c^0\})$. Consider, as a motivation, the network diffusion property in the introduction that counts leaders of tribes. Computing such properties is hard even when all the network diffusion variables are observed, and we will see that computing the expectations with unobserved variables is not tractable. However, we will investigate simplifications of these properties that are interesting and useful, and at the same time their expectations can be computed efficiently.

We will consider two different categories of network diffusion properties that involve the network and the cascade: network centric and cascade centric properties. In a network-centric property, the focus is on entities in the network, such as nodes, or edges, which satisfy some property in the network, as well as some property in the cascade. The 'counting leaders' property is an example in this category, with nodes in the network being the focus. A cascade-centric property, on the other hand, is about entities in the cascade, such as individual infections, which satisfy certain cascade property and additionally some network property. Before discussing more about such properties in Sec 5.2 and Sec 5.3, we first investigate conditions under which expectations of network diffusion properties are efficiently computable.

5.1 Niceness of Properties

Given the large size of real-world network diffusion data, in all of the following discussion, we will say that a computation is efficient if it is linear in the size of the network and the lengths of the cascades. Computing the expectation involves marginalizing out two variables: an integration over possible network strengths α, and a summation over possible network paths defined by the infection parent variables z. We first analyze these two marginalizations separately, before looking at computing the complete expectation.

Integrating over α: First, we characterize properties for which the integration over α can be performed efficiently. We call such properties *nice-α*. Intuitively, a *nice-α* property decomposes into terms that involve the parent variables z, and individual connection strengths α_{uv}. Additionally, the functions involving α_{uv} should be amenable to analytical integration with $p(\alpha_{uv}|z, \{c^o\})$ which is in the Gamma form.

Definition A property $f(\alpha, \mathbf{z})$ is *nice-α* if it can be written as $f(\alpha, \mathbf{z}) = g(\mathbf{z}) \prod_{u,v} h_{uv}(\alpha_{uv}, z)$ or as
$f(\alpha, \mathbf{z}) = g(\mathbf{z}) \sum_{u,v} h_{uv}(\alpha_{uv}, z)$ where
$\int h_{uv}(\alpha_{uv}, z)\ p(\alpha_{uv}|z, \{c^o\}) d\alpha_{uv}$ can be performed analytically $\forall\ u, v$.

Theorem 5.1 *Let $f(\alpha, \boldsymbol{z})$ be nice-α. Then computing the z-marginal $f_z(z) = \int_\alpha f(\alpha, \boldsymbol{z}) p(\alpha|z, \{c^o\}) d\alpha$ is $O(|\boldsymbol{V}|^2)$.*

The notion of nice-α can be extended to properties that are depend only on α and not on z. Such properties $f(\alpha)$ need to be of the form $\prod_{u,v} h_{uv}(\alpha_{uv})$ or $\sum_{u,v} h_{uv}(\alpha_{uv})$, where $\int h_{uv}(\alpha_{uv}) p(\alpha_{uv}|z, \{c^o\}) d\alpha_{uv}$ can be performed analytically for *all* z. Note that the z-marginal $f_z(z)$ is still a function of z through $p(\alpha|z, \{c^o\})$ Also, properties that are independent of α are trivially *nice-α*. Finally, this complexity corresponds to the scenario when no edge information is available to begin with. Given a set E of potential edges, the complexity above would be $O(|E|)$.

Summing over z: Now we characterize properties for which the summation over infection parents z can be performed efficiently. We call such properties *nice-z*. Recall from Eqn. 2 that the posterior distribution $p(z|\alpha, \{c^o\})$ decomposes into terms involving individual z_i variables. Intuitively, the summation over z can be performed efficiently if the property $f(\alpha, \mathbf{z})$ also decomposes over z.

Definition A property $f(\alpha, \mathbf{z})$ is *nice-$\boldsymbol{z}$* if it can be written either as $f(\alpha, \mathbf{z}) = g(\alpha) \prod_i h_i(z_i, \alpha)$ or as $f(\alpha, \mathbf{z}) = g(\alpha) \sum_i h_i(z_i, \alpha)$

Theorem 5.2 *Let $f(\alpha, \boldsymbol{z})$ be nice-$\boldsymbol{z}$. Then the α-marginal $f_\alpha(\alpha) = \sum_z f(\alpha, \boldsymbol{z}) p(\boldsymbol{z}|\alpha, \{c^o\})$ can be computed in $O(\pi|C|)$ time, where $\pi = \max_i \pi_i$ is the maximum number of potential parents over all infections.*

As for nice-α, the notion of *nice-$\boldsymbol{z}$* can be extended to properties that involve only z and ignore α. Note that for such properties, the α-marginal $f_\alpha(\alpha)$ still

depends on α through the posterior distribution $p(z|\alpha, \{c^o\})$. Also, a function which is independent of $\mathbf{z}$ is trivially *nice-*$\boldsymbol{z}$. Finally, $\pi \ll |C|$ and the complexity above can be written as $O(|C|)$.

Marginalizing both α and z: For computing the complete expectation in Eqn. 4, both marginalizations above need to be performed. We now investigate strategies for dong this. Interestingly, it turns out that the complete expectation can be computed efficiently and exactly for some network diffusion properties, which we call *nice-*$\boldsymbol{z}, \alpha$.

Definition A property $f(\alpha, \mathbf{z})$ is *nice-*$\boldsymbol{z}, \alpha$ if it can be written as

$$f(\alpha, \mathbf{z}) = \prod_{u,v} g_{uv}(\alpha_{uv}) \prod_{i=1}^{|\mathbf{D}|} \frac{h_i(z_i)}{\alpha_{u_{z_i} u_i}} \tag{8}$$

where $\int g_{uv}(\alpha_{uv}) p(\alpha_{uv}|z, \{c^o\}) d\alpha_{uv}$ can be performed analytically $\forall u, v$.

Lemma 5.3 *A property that is nice-*$\boldsymbol{z}, \alpha$ *is both nice-*α *according to Defn. 5.1 and nice-*$\boldsymbol{z}$ *according to 5.1.*

In addition to being nice-α and nice-z, it is necessary that *nice-*$\boldsymbol{z}, \alpha$ properties decouple the α and z variables, not just in the property, but also in the posterior distribution $p(\alpha, z|\{c^o\})$. This is achieved by introducing the $\alpha_{u_{z_i} u_i}$ terms in the property definition. These cancel out the corresponding terms in $p(\alpha, z|\{c^o\})$, which are responsible for the coupling.

Theorem 5.4 *Let $f(\alpha, \boldsymbol{z})$ be nice-*$\boldsymbol{z}, \alpha$*. Then the expectation $\bar{f}(\alpha, z)$ can be computed in $O(\pi|\boldsymbol{D}|)) + O(|\boldsymbol{V}|^2)$ time, up to a multiplicative constant.*

The multiplicative constant in question here is the inverse of the data likelihood $p(\{c^o\})$ of the observed variables $\{c^o\}$ in the cascades. This implies that we may not be able to compute the exact value of any *nice-*$\boldsymbol{z}, \alpha$ efficiently, but we can compare the values of two different *nice-*$\boldsymbol{z}, \alpha$ properties.

In general, there will be properties for which any one or both marginalizations cannot be performed analytically or efficiently. In such cases, we resort to Monte Carlo techniques. Here, we will assume that it is possible to draw *iid* samples $(\alpha^{(s)}, z^{(s)})$ from the joint distribution $p(\alpha, z|\{c^o\})$, and similarly $(\alpha^{(s)}) \sim p(\alpha|\{c^o\})$ and $(z^{(s)}) \sim p(z|\{c^o\})$ from the marginal distributions. In Sec 6, we describe a Gibbs Sampling algorithm for drawing such samples.

First consider properties which are *nice-*α but for which the subsequent marginalization $\sum_z f_z(z) p(z|\{c^o\})$ over z cannot be performed efficiently. For such properties, we first obtain the z-marginal $f_z(z)$ efficiently, and then use Monte Carlo summation for z:

$$\bar{f}(\alpha, z) \approx \frac{1}{S} \sum_s f_z(z^{(s)}), \text{ where } z^{(s)} \sim p(z|\{c^o\}), \ s = 1 \ldots S$$

On the other hand, consider properties which are *nice-z* but for which the subsequent marginalization $\int f_\alpha(\alpha) p(\alpha|\{c^o\}) d\alpha$ over α cannot be performed

analytically. For such properties, we first obtain the α-marginal $f_\alpha(\alpha)$ efficiently, and then use Monte Carlo integration for α:

$$\bar{f}(\alpha, z) \approx \frac{1}{S} \sum_s f_\alpha(\alpha^{(s)}), \text{ where } \alpha^{(s)} \sim p(\alpha|\{c^o\}), \ s = 1 \dots S$$

Finally, for properties where neither of the two marginalizations can be performed efficiently, we use Monte Carlo integration for both α and z:

$$\bar{f}(\alpha, z) \approx \frac{1}{S} \sum_s f(\alpha^{(s)}, z^{(s)}),$$
$$\text{where } (\alpha^{(s)}, z^{(s)}) \sim p(\alpha, z|\{c^o\}), \ s = 1 \dots S$$

Having characterized the notion of niceness for network diffusion properties in terms of computing the expectation, we now return to our motivating properties, and analyze them in this light.

5.2 Network-centric Properties

We first discuss network-centric properties, which involve computing scores for specific entities in the network, such as nodes, edges, etc. These scores are functions of the connection strengths α in the network and also of the cascades. Recall that the network and the cascades are connected through the node id's u_i in the individual infections.

The basic building blocks, for network scores of network entities, is the *direct* connection strength α_{uv} between nodes u and v. Using this, we can define $\alpha^{(2)}_{uv} = \sum_w \alpha_{uw}\alpha_{wv}$ or its approximation $\max_w \min(\alpha_{uw}, \alpha_{wv})$ as the *second-order* connection strength between u and v in the network. Generalizing further, $\alpha^{(r)}_{uv} = \sum_w \alpha^{(r-1)}_{uw}\alpha_{wv}$ is the r^{th}-order connection strength between them, and $\alpha^*_{uv} = \sum_{r=1}^{R} \alpha^{(r)}_{uv}$.

The other building block, for cascade scores of network entities, is the *direct* transmission frequency $\rho_{uv} = \sum_{ij} I(u_i = v, z_i = j, u_j = u)$ between u and v in the cascades. This can be generalized the same way as α_{uv} to define $\rho^{(2)}_{uv} = \sum_w \rho_{uw}\rho_{wv}$ or its approximation $\max_w \min(\rho_{uw}, \rho_{wv})$ as the *second-order* transmission frequency between u and v in the cascades. The interpretation is that u frequently infects some node w, who in turn frequently infects v in the cascades. This can also be generalized to similarly define $\rho^{(r)}_{uv}$ as the r^{th}-order transmission frequency, and finally ρ^*_{uv}.

Node-centric Properties: We now formally define our first motivating network diffusion property, that of finding influential nodes considering both network strengths α^*_{uv} and transmission frequencies ρ^*_{uv}.

Node influence score : Intuitively, a node's influence score $f_u(\alpha, z)$ is high if it has many 'followers' v with high α^*_{uv} and high ρ^*_{uv}. One way to capture this is to define

$$f_u(\alpha, \mathbf{z}; a, r) = \sum_v I(\alpha^*_{uv} > a) I(\rho^*_{uv}(\mathbf{z}) \geq r) \tag{9}$$

Alternatively, we could couple together α^*_{uv} and ρ^*_{uv}: $f_u(\alpha, \mathbf{z}) = \sum_v \alpha^*_{uv} \circ \rho^*_{uv}$ or $f_u(\alpha, \mathbf{z}) = \sum_v \alpha^{*\rho^*_{uv}}_{uv}$ Unfortunately, all of these forms are nice neither in α nor in z even when $R = 2$, or in other words we consider first and second order infections. So the only way to estimate them is to sample over both α and z. But it turns out that the definition for $R = 1$ is more tractable.

Node influence score for direct infections: This is the special case of node influence score where we only consider directly connected nodes in the network who are also directly infected in the cascades.

$$f_u(\alpha, \mathbf{z}; a, r) = \sum_v I(\alpha_{uv} > a) I(\rho_{uv}(\mathbf{z}) \geq r) \tag{10}$$

While this does not provide as much information about the influence of a node, this a reasonable surrogate. It turns out that this property is nice-α, so that the expectation can be partly calculated efficiently and exactly. The reason for not being nice-z is that while $\rho_{uv}(\mathbf{z})$ is itself nice-z, discretization of $\rho_{uv}(\mathbf{z})$ through $I(\rho_{uv}(\mathbf{z}) \geq r)$ leads to coupling across z_i variables. This implies that the alternatives $f_u(\alpha, \mathbf{z}; a) = \sum_v I(\alpha_{uv} > a)\rho_{uv}(\mathbf{z})$ and $f_u(\alpha, \mathbf{z}) = \sum_v \alpha_{uv}^{\rho_{uv}(\mathbf{z})}$ are both nice-α and nice-z, though not *nice-*$\boldsymbol{z}, \alpha$, which provides two different routes for partly approximating their expectations.

We may restrict the node influence score above to consider only the network connections and ignore the cascade:

$$f(\alpha; a)_u = \sum_v I(\alpha_{uv} > a) \tag{11}$$

Interestingly this is also nice-α and (trivially) nice-z, but not *nice-*$\boldsymbol{z}, \alpha$, like the definition above. Alternatively, we could consider only the transmission frequencies:

$$f(\mathbf{z}, r)_u = \sum_v I(\rho_{uv}(\mathbf{z}) \geq r) \tag{12}$$

This is (trivially) nice-α but not nice-z because of the discretization.

Edge-centric Properties: Edge-centric properties compute scores for edge (u, v) in the network. As before, we will focus on scores involving connection strengths α_{uv} and transmission frequencies ρ_{uv}.

Edge Distribution : Given a range (r_1, r_2) for the transmission frequency, and a range (a_1, a_2) for the connection strength, this counts the number of edges (u, v) in the network whose connection strengths α_{uv} and transmission frequencies ρ_{uv} lie in this range.

$$f(\alpha, \mathbf{z}) = \sum_{u,v} I(a_1 < \alpha_{uv} < a_2) I(r_1 \leq \rho_{uv}(\mathbf{z}) < r_2)$$

The resultant distribution of the edges over the α, ρ space can help in understanding how effective viral marketing strategies can be for this network. Additionally, the edge distribution can be viewed as the distribution of the summed

(or averaged) direct node influence scores. Recall that the plots in the introduction corresponded to this property.

Marginals or projections of this distribution along the ρ and α dimensions can also be useful.

$$f(\mathbf{z}) = \sum_{u,v} I(r_1 \leq \rho_{uv}(\mathbf{z}) < r_2); \quad f(\alpha) = \sum_{u,v} I(a_1 < \alpha_{uv} < a_2)$$

All of these properties are nice-α, but not nice-z.

Observe that removing the binning for the α-projection recovers the well studied network inference problem.

$$f(\alpha, \mathbf{z}) = \alpha \tag{13}$$

However, taking the expectation gives the Bayesian formulation of the network inference problem, where we are seeking the expected network connection strengths given the cascades. This is again nice-α, but not nice-z.

We have seen that all these network-centric properties can at best partially nice. We conclude this discussion by presenting an interesting property that is *nice-*$\boldsymbol{z}, \alpha$. Imagine that we are interested in finding strong edges that are not frequent, and weak edges that are frequent. For this, the following score is useful:

$$f_{uv}(\alpha, \mathbf{z}) = \alpha_{uv}^{-\rho_{uv}(\mathbf{z})} \tag{14}$$

It can be shown that this function satisfies Defn. 5.1, and therefore the complete expectation can be computed exactly and efficiently.

5.3 Cascade-centric Properties

For cascade centric properties, the focus is on entities in the cascade, such as individual infections, for which we compute some score based on the network as well as the cascade. We illustrate such properties using individual infections.

Infections due to Strongest Neighbor: The strongest neighbor of a node v in the network is the one with the maximum connection strength α_{uv}. Now, we can count the number of infections i, for which the infecting parent u_{z_i} is the strongest neighbor for u_i in the network.

$$f(\alpha, \mathbf{z}) = \sum_i I(u_{z_i} = \arg\max_v \alpha_{vu_i}) \tag{15}$$

We can similarly count number of infections by the n^{th}-strongest neighbor, for $n > 1$. Such an analysis is helpful for designing viral marketing strategies for a network. This property is nice-z, but not nice-α.

As an even simpler example of a network property, we can consider the checking parents nodes for individual infections. *Infection parent identification*: This indicates if node u is the parent of infection i.

$$f(\alpha, \mathbf{z})_{iu} = 1 \text{ if } z_i = u; \quad = 0 \text{ otherwise} \tag{16}$$

This is equivalent to recovering the diffusion tree for a cascade. This second infection-centric property is nice-z and also trivially nice-α.

It is worth observing that complete likelihood $p(\{c^o\}, \mathbf{z} \mid \alpha)$ of a cascade C given network strengths α can be seen as a cascade-centric property, where the entity of interest is the entire cascade.

$$f(\alpha, \mathbf{z}) = p(\{c^o\}, \mathbf{z} \mid \alpha) = \prod_{u,v} e^{-\alpha_{uv}\Delta_{uv}} \prod_i \alpha_{u_{z_i} u_i} \tag{17}$$

The likelihood function can be viewed similarly as a cascade-centric property. Unlike complete likelihood, this considers the likelihood of only the observed infection variables, and the parent variables z are summed out.

$$f(\alpha) = p(\{c^o\} \mid \alpha) = \prod_{u,v} e^{-\alpha_{uv}\Delta_{uv}} \prod_i \sum_{z_i \in \pi_i} \alpha_{u_{z_i} u_i} \tag{18}$$

Both of these properties are nice-z (likelihood trivially so), but not nice-α. The expectation of this property can be interpreted as considering the entire posterior distribution over α, learnt from the training cascades, to explain the test cascades. In contrast, the frequentist strategy uses only the maximum likelihood point estimate.

6 Inference

We have seen in Sec. 5 that computing the expectation for network diffusion properties that are not completely nice requires drawing samples from the posterior distribution $p(\alpha, z|\{c^o\})$ over network strengths α and infection parents z conditioned on the observed cascades $\{c^o\}$. In this section, we propose a Gibbs Sampling framework for this. In this framework, we iterate over all latent variables, sampling a new value for it from its conditional distribution, given the current values of all other variables. Under ergodicity conditions, asymptotically the samples are from the joint posterior distribution over all latent variables. For our problem, we need to draw samples from $p(z_i|\{c^o\}, \alpha, z_{-i})$ and from $p(\alpha_{uv}|\{c^o\}, z, \alpha_{-uv})$, where z_{-i} and α_{-uv} denote variables other than z_i and α_{uv}. (All expressions below are for the Exponential Distribution. Expressions for the Rayleigh are similar.)

First, the posterior distribution $p(\mathbf{z}, \alpha \mid \{c^o\})$ over both z and α looks as follows:

$$p(\mathbf{z}, \alpha \mid \{c^o\}) \propto \prod_{uv} \alpha_{uv}^{\rho_{uv}(z)+a-1} e^{-\alpha_{uv}(\Delta_{uv}^t + b)}$$

Given this, the conditional distribution for the i^{th} infection parent z_i turns out to have a very simple form:

$$p(z_i = j \mid z_{-i}, \alpha, \{c^o\}) \sim \alpha_{ji} \tag{19}$$

The conditional distribution for individual network strengths α_{uv} also has a simple *Gamma* density form:

$$p(\alpha_{uv} \mid \{c^o\}, \mathbf{z}, \alpha_{-uv}) \sim Gamma(\rho_{uv} + a, \Delta_{uv} + b) \tag{20}$$

For network properties that are nice-α, only samples of z are required. In such cases, an alternative is to perform *collapsed Gibbs Sampling*, by analytically integrating out α:

$$\begin{aligned} p(\mathbf{z}|\{c^o\}) &\propto \int_{\alpha} p(\mathbf{z}, \{c^o\} \mid \alpha) p(\alpha) d\alpha \\ &\propto \prod_{uv} \frac{\Gamma(\rho_{uv}(z) + a)}{(\Delta_{uv} + b)^{(\rho_{uv}(z)+a)}} \end{aligned} \tag{21}$$

Given this conditional, the conditional distribution for individual infection parents z_i looks as follows:

$$p(z_i = j \mid \mathbf{z}_{-i}, \{c^o\}) \propto \frac{(\rho^{-i}_{u_j u_i}(z) + a)}{\Delta_{u_j u_i} + b} \tag{22}$$

The collapsed Gibbs Sampling algorithm simply involves repeatedly sampling the parents of the individual infections from a Multinomial distribution, given the parents of all other infections. To the best of our knowledge, this is the first Gibbs Sampling algorithm for network analysis.

Recently, the independent cascade model has been extended to handle features of individual infections [22], which is useful to capture contents of social media posts when inferring influences. Our approach can be extended in an straight forward manner to incorporate features in this way. The analysis in Sec. 5 remains unchanged since the decoupling in Eqns. 2 and 7 still hold. The Gibbs Sampling updates acquire an additional feature term. We omit further details due to space constraints.

Map-Reduce Implementation To sample a parent for an infection of node v, we only require ρ_{*v} and Δ_{*v} in case of collapsed sampler, and α_{*v} in the case of uncollapsed sampler. To sample α_{*v}, we again require only ρ_{*v} and Δ_{*v}. Moreover, after sampling we update only ρ_{*v} and α_{*v}. As a result, we can run the sampler for each node v in parallel if we know the set of possible parents of each infection. The reducer, where the sampler is run, exploits this parallelism. When computing Δ_{uv}s , we can process each cascade in parallel and add these values for each u, v pair, to get the final values across cascades. The Mapper, which computes Δ_{uv} and the set of possible parents for each infection exploits this parallelism.

Each mapper computes Δ_{uv} for the set of cascades given to it and emits (v:Δ_{uv}) pair. It also generates a list of possible parents for each infection and emits (v:[Infection,{possible parent Infections}]) pair.

Each reducer performs sampling for a subset on nodes. For each node v, it combines the Δ_{uv}s from different mappers to compute the final Δ_{uv}. It then

creates a list of infections of node v with possible parent set. It performs the sampling for these infections and generates the samples. The samples from various reduces are be combined to generate the final samples.

7 Experiments

In this section, we report experimental evaluations of various network diffusion properties defined in Sec. 5 using our Bayesian approach on synthetic and real world datasets. We report how accurately we are able to estimate the properties and also how well our algorithms scale for large datasets.

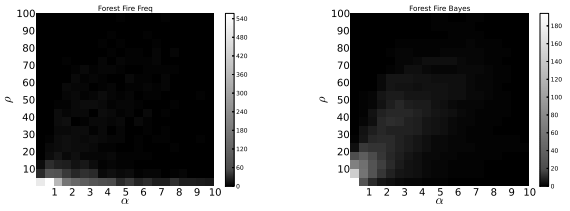
Baseline : We note at the outset that this general problem has not been studied before, so that there is no baseline we can compare against as such. However, one potential strategy is to first recover a point estimate $\hat{\alpha}$ (e.g. MLE) of the network strengths α using a state-of-the-art approach, consider the most likely infection parents $\hat{z} = \arg\max_z p(z|\alpha, \{c^o\})$ given $\hat{\alpha}$, and then evaluate the property $f(\hat{\alpha}, \hat{z})$ at $\hat{\alpha}, \hat{z}$. While this suffers from deficiencies outlined in Sec. 3, this is the best existing approach for our problem. As the state-of-the-art network inference approach for the continuous time independent cascade model, we used the featureless version of MONET [22]. We do not use NETRATE [6], since it does not support multiple infections of a node in a cascade. We have used the Exponential distribution for all experiments. In the rest of this section, we will refer to this approach as the frequentist plug-in approach (**FP**), and to our proposed approach of computing expectations as the Bayesian Expectation approach (**BE**).

Synthetic data experiments: We first conducted experiments on multiple synthetic datasets. First, they allowed us to evaluate accuracy against a gold-standard, which unfortunately is unavailable for most real-world network diffusion datasets. Secondly, they helped us understand how well our proposed approach works for different kinds of graphs. Following earlier experiments on network inference [7, 6], we created synthetic graphs with 1024 nodes using the Forest Fire (FF), and the Random (Rnd), Hierarchical (HI) and Core-Periphery (CP) Graph models, the last three being instances of Kronecker Graph models. We the same parameter values ([0.5, 0.5; 0.5, 0.5] for Rnd, [0.9,0.1;0.1,0.9] for HI, [0.9,0.5;0.5,0.9] for CP) as Gomez-Rodriguez et. al. [6]. To generate weights α_{uv} for each edge (u, v), we sampled uniformly from $(0.01, 10)$ [3]. We then generated 20 *splitting* cascades on top these graphs with 2 randomly chosen seeds for each cascade. Finally, we had 2046 edges and 48,947 infections for the Random graph, 1496 and 38046 for the Hierarchical, 2042 and 58062 for the Core-Periphery and 2023 and 55274 for the Forest Fire graph.

Recall that one of the reasons behind the synthetic data experiments is to be able to evaluate accuracy. For the infection parents z, we considered the true parents as the gold-standard. However, for the real-valued network connections α_{uv}, the true values are very difficult to recover for any algorithm given finite length cascades. For example, it is impossible to recover the strength for any edge that has no transmission in the cascade. Therefore, we considered

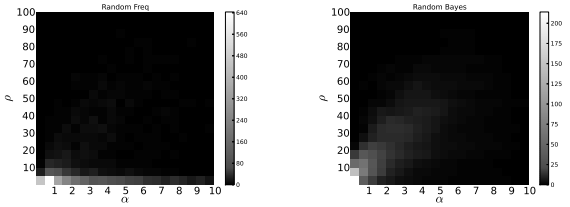
as our gold-standard the best achievable α_{uv} given the true infection parents in the cascades: $\alpha^* = \arg\max_\alpha f(\{c^o\}, z^*; \alpha)$. To evaluate accuracy of a computed property, we used absolute error between the gold-standard $f(\alpha^*, z^*)$ and the estimated value of the property for scalars, and root mean squares of the individual errors for vectors and matrices.

Network-centric Properties: In this category, we first evaluate the edge-distribution (Eqn. 5.2) as an example of a **property on edges**. Evaluating this property is challenging, because of the threshold parameters a and r. We discretized the α and the ρ ranges, and within each region of the α, ρ space, computed the actual, BE and FP values of these properties, and the errors for BE and FP to determine which is better.



(a) Frequentist (b) Bayesian Expectation

Figure 3: Inferred Edge distribution for Forest Fire

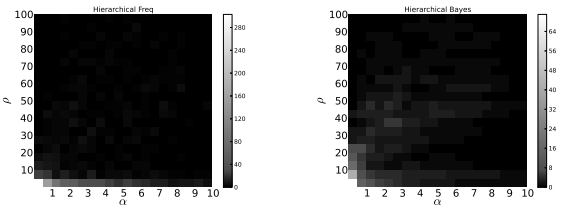


(a) Frequentist (b) Bayesian Expectation

Figure 4: Inferred Edge distribution for Random

The actual plots for the four networks were introduced in Fig. 1. The FE and BE reconstructions for the Core-Periphery graph were also introduced earlier in Fig. 2. The reconstructions for the other three graphs are shown in Figs. 3, 4, and 5. It can be seen quite clearly that while BE is able to reconstruct the actual distributions to a reasonable extent for all 4 graphs, FP does quite poorly. In fact, the FP reconstruction looks similar for all 4 cases, and fails to pick up the signatures for the different graphs.

We also calculated the actual errors for the two approaches over the α, ρ space. Since it is difficult to visualize the plots in 2D, we next evaluate the projections on the α-dimension and z-dimension (Eqn. 5.2) for the edge distribution in more detail for the 4 graphs.



(a) Frequentist (b) Bayesian Expectation

Figure 5: Inferred Edge distribution for Hierarchical

Table 1: α-proj. for edge distribution: Abs. error

NW	CP		HI		Rnd		FF	
α	BE	FE	BE	FE	BE	FE	BE	FE
0 1	591	6514	156	1501	470	4004	432	4008
1 2	169	1587	5	287	34	1085	30	1039
2 3	13	618	18	166	6	497	8	379
3 4	9	340	1	87	6	265	15	196
4 5	8	159	2	57	32	139	9	101
5 6	6	138	1	51	14	127	11	61
6 7	3	75	1	34	7	86	13	62
7 8	1	82	**14**	**9**	9	70	4	64
8 9	2	48	8	17	13	45	1	38
9 10	1	46	1	16	4	44	4	36

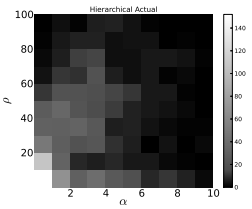
Table 2: ρ-proj. for edge distribution: abs. error

NW	CP		HI		Rnd		FF	
ρ	BE	FE	BE	FE	BE	FE	BE	FE
0 10	524	4373	69	1032	344	3339	287	2903
10 20	**348**	**228**	14	15	**108**	**8**	89	21
20 30	20	82	3	0	40	87	20	51
30 40	59	91	1	14	28	64	29	73
40 50	47	100	3	19	13	44	10	43
50 60	43	59	5	16	6	16	8	34
60 70	19	27	0	3	1	13	9	12
70 80	1	1	1	0	2	6	2	3
80 90	4	4	2	0	0	4	2	12
90 100	3	5	2	1	1	2	2	8

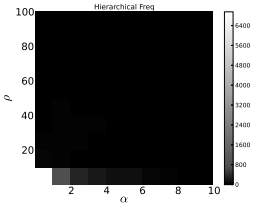
In Tab. 1, we record the errors for BE α-projection and the FP α-projection for different α-intervals. We can see that for the α-projection, the FP errors are an order of magnitude bigger for all intervals, except for $\alpha \in (7, 8)$ for Hierarchical. Similarly, in Tab. 2, we record the errors for BE ρ-projection and the FP ρ-projection for different ρ-intervals. In this case as well, FP error is significantly lower for the (10-20) interval for CP and Rnd.

Finally, we come to properties on nodes. We evaluated direct node influence (Eqn. 10), and indirect node influence for 2^{nd}-order neighbors (Eqn. 9) for the 4 graphs. Again, we partitioned the α, ρ-space into regions. However, reporting detailed results is even harder here, since we have actual, BE and FP scores for each node for each α, ρ-region. One option is to sum (or average) over the influence score over all nodes. However, recall that one interpretation of the edge-distribution is the distribution of the sum of direct influence scores over all nodes. So the edge-distribution evaluation above additionally serves as an evaluation of the direct node influence scores at an aggregate level.

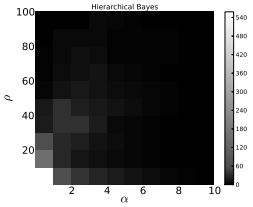
Due to space constraints, in Fig. 6, we show the (averaged) node indirect



(a) Actual



(b) Frequentist



(c) Bayesian Expectation

Figure 6: Indirect Edge distr. for Hierarchical

influence distribution only for the Hierarchical graph. Again, we see that BE is able to pick up the signature of the distribution to a reasonable extent, whereas FP has failed completely.

Table 3: Node influence scores: Agg. error

NW	CP		HI		Rnd		FF	
	BE	FP	BE	FP	BE	FP	BE	FP
Dir	29	124	22	46	30	104	30	98
InDir	62	417	15	98	27	288	27	273

In Tab. 3, we report the aggregated errors over all nodes and over all (α, ρ) regions, for both direct and indirect influence scores. We have scaled the values down by the total number of nodes, which is 1024. We can see again see that the BE errors are significant smaller than the FP errors across the board.

Cascade-centric Properties: Under cascade-centric properties, we evaluate infections due to n^{th} strongest neighbor (Eqn. 15) for $n = 1, 2, 3$.

Table 4: Infections by n^{th}-strongest nbr: abs. error

NW	CP		HI		Rnd		FF	
n	BE	FP	BE	FP	BE	FP	BE	FP
1	3711	22090	631	13537	578	21613	127	23476
2	2374	6171	691	2506	1352	3533	1284	**712**
3	191	**59**	194	3924	152	4825	388	6828

Tab. 4 records the absolute error of counting infections by the n^{th}-strongest neighbor for $n = 1, 2, 3$. Notice that FP has very high errors for $n = 1$. There are just two instances where FP works better than BE: for $n = 2$ in Forest Fire and for $n = 1$ in Core-Periphery, where the values are comparable. In all other

cases, FP has significantly higher errors than BE.

Likelihood, Network Inference and Parent Identification: We have seen that BE outperforms FP for various network diffusion properties. Such performance difference is attributable to two different kind of issues. Many to one functions. Bayesian approach of using the full posterior distribution versus frequentist plugging in. To evaluate the second aspect we look at the basic inference problems for network analysis, and generalization ability on held-out data.

Table 5: Log-likelihoods on synthetic data

NW	CP	HI	Rnd	FF
	BE, FP	BE, FP	BE, FP	BE, FP
Test	1.0e4, 0.6e4	6.5e3, 2.4e3	1.1e4, -1.5e4	1.2e4, 926
Train	2.8e4, 3.6e4	2.0e4, 2.2e4	2.3e4, 2.9e4	2.8e4, 3.3e4

In Tab. 5, we record the train and test likehoods for the 4 synthetic datasets. We see that BE consistently has higher test likelihood, while the train likelihood is higher for FP, suggesting overfitting.

Table 6: Network Inf. (NI) & Parent Id. (PI)

NW	CP		HI		Rnd		FF	
	BE	FP	BE	FP	BE	FP	BE	FP
NI	0.116	2.553	0.884	3.210	0.147	17.483	0.329	736.821
PI	0.533	0.406	0.861	0.783	0.757	0.646	0.770	0.674

In Tab. 6, we record the errors in recovery of α for BE and FP. Observe that the errors are consistently lower for BE across the 4 datasets. In fact, the FP errors are very high for the Random and Forest Fire datasets. In Tab. 6, we also see that parent identification accuracy of BE is consistently around 10% more than that of FP. Though loglikelihood, network inference and parent identification can be also seen as network diffusion properties within our framework, these three experiments serve more to demonstrate the strength of the Bayesian approach in general for network diffusion analysis independently of properties.

Iterations vs Error : Before moving on to experiments on real-world data, we make a note about Gibbs Sampling iterations. Gibbs Sampling algorithms often take thousands of iterations to converge, which can be a serious problem for large real-world datasets. For all our experiments, accuracy increases very sharply in the initial iterations, and is close to the best value within 100-200 iterations.

Experiments on real-world data: We now report experiments on real-world data, where the graph structures could be more complex than the synthetic settings. What is more likely is that underlying diffusion process is different from the Independent Cascade model, which our models assume, and which we had used for generating the synthetic cascades. We have performed experiments on two real-world network diffusion datasets from the information diffusion and social media domains. The nature of insights from the two datasets is similar. Due to space constraints, we only report our findings on one of them.

The **Meme Tracker** dataset[1] records the duffusion of "memes" or catch-

[1] http://snap.stanford.edu/infopath//data.html

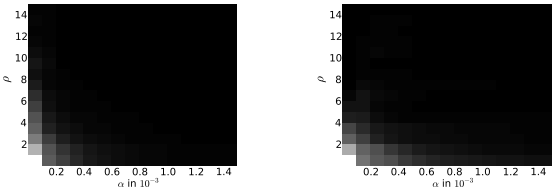
phrases across 5000 most active blogs and news sites between March 2011 and February 2012. The flow of each meme corresponds to one cascade. Related memes are grouped into one topics. For our experiments, we selected 5 topics, 2 of which have been used in earlier experiments involving non-stationary networks [9], and 3 others that seem stationary. Basketball has 1460 Nodes, 15417 Infections in 158 cascades, Alcohol 1993 nodes and 17321 infections in 167 cascades, Technology 2701 nodes and 35037 infections in 323 cascades, NBA 2481 nodes and 22736 infections in 229 cascades, and Occupy 1921 nodes and 21109 infections in 200 cascades. In each topic, we consider all sufficiently long cascades (length > 75). We split the cascades randomly (80-20 split) to generate the training and test cascades, and then prune infections of users in test cascades, who are not present in the training cascades.

Since no gold-standard is available for even α or z for this dataset, the only quantitative comparison between BE and FP that we were able to perform was using loglikelihood on held-out test data, using the knowledge of α learnt from training data. Recall that likelihood can be considered as nice-z property in our framework. However, it is the best scenario for the baseline since likelihood is a one-to-one function of α for this problem.

Table 7: Loglikelihood for Meme Tracker

	Bball	Alcohol	Tech.	NBA	Occupy
BE	-3.57e5	-5.91e5	-3.69e5	-6.1e5	-6.03e5
FP	-10.61e5	-18.41e5	-53.52e5	-12.87e5	-8.28e5

In Tab. 7, we report the loglikelihood values for the 5 selected topics. We can see that the BE values are significantly better than the FP values. Based on this, we feel that BE will outperform FP to a larger extent for other properties on real-world datasets.



(a) Tech (b) Alcohol

Figure 7: Edge distribution for Meme Tracker topics

Though we were unable to compare against a gold standard, we computed the network-centric and cascade-centric properties for Meme Tracker. In Fig. 7, we plot the edge distribution for two of the topics. We can see that the nature of the plots is different from all of the synthetic datasets. The mass is more concentrated towards weaker, infrequent edges. We suspect that this is because of the way users were sampled for this dataset.

Table 8: Millions of Infections vs time (secs)

# Infections	Time (12 nodes)	Time (1 node)
15	552	3635
31	888	6873
43	1311	10277
63	1948	14783

Scaling experiments : We also experimented with larger volumes of the Meme Tracker data using our map-reduce implementation. We created increasingly larger dataset sizes by randomly sampling cascades and checking the execution time for 100 iterations of Gibbs Sampling. We performed experiments on a Intel Xeon server with 100GB RAM, which supports 12 mapper/ reducer tasks in parallel.

In Tab. 8, we record execution time with increasing data size using 12 nodes and compare against the time taken on a single node. We can see that the map-reduce implementation allows us to scale our analysis by providing a (roughly) linear speed-up in terms of number or nodes.

In summary, the experiments clearly demonstrate that computing expectations under the posterior distribution leads to significantly better reconstruction of a wide variety of network diffusion properties. The proposed Bayesian framework that combines exact efficient computation with Gibbs Sampling based approximations outperforms state-of-the-art algorithms even for the well-studied network inference and parent identification problems, and in generalizing to held-out test data. The map-reduce implementation is promising in terms of scaling up the analysis to study properties of large network diffusion datasets.

8 Conclusions

In this paper, we have investigated the novel problem of computing expectations of properties of network diffusions involving hidden variables. We have proposed a Bayesian framework for computing such expectations, and proposed and characterized network diffusion properties that can be handled efficiently in this framework. In experiments over synthetic and real world datasets, we have shown that we are able to reconstruct network properties significantly more accurately than a frequentist baseline.

References

[1] N. Barbieri, F. Bonchi, and G. Manco. Influence-based network-oblivious community detection. In *ICDM*, 2013.

[2] F. Bonchi. Influence propagation in social networks: A data mining perspective. *IEEE Intelligent Informatics Bulletin*, 12(1), 2011.

[3] N. Du, L. Song, M. Gomez-Rodriguez, and H. Zha. Scalable influence estimation in continuous-time diffusion networks. In *NIPS*, 2013.

[4] N. Du, L. Song, A. Smola, and M. Yuan. Learning networks of heterogeneous influence. In *NIPS*, 2012.

[5] A. Elfessi and D. Reineke. A bayesian look at classical estimation: The exponential distribution. *Journal of Statistics Education*, 9(1), 2001.

[6] M. Gomez-Rodriguez, D. Balduzzi, and B. Schlkopf. Uncovering the temporal dynamics of diffusion networks. In *ICML*, 2011.

[7] M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *KDD*, 2010.

[8] M. Gomez-Rodriguez, J. Leskovec, and B. Schlkopf. Modeling information propagation with survival theory. In *ICML*, 2013.

[9] M. Gomez-Rodriguez, J. Leskovec, and B. Schlkopf. Structure and dynamics of information pathways in online media. In *WSDM*, 2013.

[10] A. Goyal, F. Bonchi, and L. Lakshmanan. Discovering leaders from community actions. In *CIKM*, 2008.

[11] A. Goyal, F. Bonchi, and L. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.

[12] A. Goyal, F. Bonchi, and L. Lakshmanan. A data-based approach to social influence maximization. *PVLDB*, 2011.

[13] A. Guille, H. Hacid, C. Favre, and D. A. Zighed. Information diffusion in online social networks: A survey. *SIGMOD Rec.*, 42(2), July 2013.

[14] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[15] K. Kutzkov, A. Bifet, F. Bonchi, and A. Gionis. Strip: Stream learning of influence probabilities. In *KDD*, 2013.

[16] L. Macchia, F. Bonchi, F. Gullo, and L. Chiarandini. Mining summaries of propagations. In *ICDM*, 2013.

[17] Y. Mehmood, N. Barbieri, F. Bonchi, and A. Ukkonen. Csi: Community-level social influence analysis. In *ECML PKDD*, 2013.

[18] C. Milling, C. Caramanis, M. S., and S. Shakkottai. Network forensics: Random infection vs. spreading epidemic. In *ACM Sigmetrics*, 2012.

[19] P. Netrapalli and S. Sanghavi. Finding the graph of epidemic cascades. In *ACM SIGMETRICS*, 2012.

[20] E. Sadikov, M. Medina, J. Leskovec, and H. Garcia-Molina. Correcting for missing data in information cascades. In *WSDM*, 2011.

[21] K. Saito, M. Kimura, K. Ohara, and H. Motoda. Learning continuous-time information diffusion model for social behavioral data analysis. In *Adv. in Mach. Learning*, 2009.

[22] L. Wang, S. Ermon, and J. Hopcroft. Feature-enhanced probabilistic models for diffusion network inference. In *ECML-PKDD*, 2012.