Stability and Robustness in Influence Maximization

XINRAN HE, University of Southern California
DAVID KEMPE, University of Southern California

In the well-studied Influence Maximization problem, the goal is to identify a set of k nodes in a social network whose joint influence on the network is maximized. A large body of recent work has justified research on Influence Maximization models and algorithms with their potential to create societal or economic value. However, in order to live up to this potential, the algorithms must be robust to large amounts of noise, for they require quantitative estimates of the influence which individuals exert on each other; ground truth for such quantities is inaccessible, and even decent estimates are very difficult to obtain.

We begin to address this concern formally. First, we exhibit simple inputs on which even very small estimation errors may mislead every algorithm into highly suboptimal solutions. Motivated by this observation, we propose the Perturbation Interval Model as a framework to characterize the stability of Influence Maximization against noise in the inferred diffusion network. Analyzing the susceptibility of specific instances to estimation errors leads to a clean algorithmic question, which we term the Influence Difference Maximization problem. However, the objective function of Influence Difference Maximization is NP-hard to approximate within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$.

Given the infeasibility of diagnosing instability algorithmically, we focus on finding influential users robustly across multiple diffusion settings. We define a Robust Influence Maximization framework wherein an algorithm is presented with a set of influence functions. The algorithm's goal is to identify a set of k nodes who are simultaneously influential for all influence functions, compared to the (function-specific) optimum solutions. We show strong approximation hardness results for this problem unless the algorithm gets to select at least a logarithmic factor more seeds than the optimum solution. However, when enough extra seeds may be selected, we show that techniques of Krause et al. can be used to approximate the optimum robust influence to within a factor of $1 - 1/e$. We evaluate this bicriteria approximation algorithm against natural heuristics on several real-world data sets. Our experiments indicate that the worst-case hardness does not necessarily translate into bad performance on real-world data sets; all algorithms perform fairly well.

CCS Concepts: • **Information systems → Social networks**; *Social advertising*; • **Theory of computation → Social networks**;

Additional Key Words and Phrases: Social networks, Influence maximization, Robust optimization, Stability analysis, Information diffusion, Submodular optimization

ACM Reference Format:
Xinran He and David Kempe. 2018. Stability and Robustness in Influence Maximization. *ACM Trans. Knowl. Discov. Data.* 1, 1, Article 1 (January 2018), 34 pages. https://doi.org/10.1145/3233227

1 INTRODUCTION

Computational social science is the study of social and economic phenomena based on electronic data, algorithmic approaches and computational models. It has emerged as an important application

Authors' addresses: Xinran He, University of Southern California, xinranhe1990@gmail.com; David Kempe, University of Southern California, david.m.kempe@gmail.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2018 Association for Computing Machinery.
1556-4681/2018/1-ART1 $15.00
https://doi.org/10.1145/3233227

of data mining and learning, while also invigorating research in the social sciences. Computational social science is frequently envisioned as a foundation for a discipline one could term "computational social engineering," wherein algorithmic approaches are used to change or mitigate individuals' behavior.

Among the many concrete problems that have been studied in this context, perhaps the most popular is Influence Maximization. It is based on the observation that behavioral change in individuals is frequently effected by influence from their social contacts. Thus, by identifying a small set of "seed nodes," one may influence a large fraction of the social network. The desired behavior may be of social value, such as refraining from smoking or drug use, using superior crops, or following hygienic practices. Alternatively, the behavior may provide financial value, as in the case of viral marketing, where a company wants to rely on word-of-mouth recommendations to increase the sale of its products.

1.1 Prevalence of Uncertainty and Noise

Contrary to the "hard" sciences, the study of social networks — whether using traditional or computational approaches — suffers from massive amounts of noise inherent in the data and models. The reasons range from the fundamental to the practical:

- At a fundamental level, it is not even clear what a "social tie" is. Different individuals or researchers operationalize the intuition behind "friendship", "acquaintance", "regular" advice seeking, etc. in different ways (see, e.g., [10]). Based on different definitions, the same real-world individuals and behavior may give rise to different mathematical models of the same "social network."
- Mathematical models of processes on social networks (such as opinion adoption or tie formation) are at best approximations of reality, and frequently mere guesses or mathematically convenient inventions. Furthermore, the models are rarely validated against real-world data, in large part due to some of the following concerns.
- Human behavior is typically influenced by many environmental variables, many of them hard or impossible to measure. Even with the rapid growth of available social data, it is unlikely that data sets will become sufficiently rich to disentangle the dependence of human behavior on the myriad variables that may shape it.
- Observational data on social behavior is virtually always incomplete. For example, even if API restrictions and privacy were not concerns (which they definitely are at this time) and a "complete" data set of Twitter *and* Facebook *and* e-mail communication were collected, it would still lack in-person and phone interactions.
- The process of inferring model parameters relies on a choice of model and hyperparameters, many of which are difficult to make. Furthermore, while for many models, parameter inference is computationally efficient, this is not universally the case.

1.2 Dealing with Uncertainty

Since none of these issues are likely to be resolved anytime soon, both the models for social network processes and their inferred parameters must be treated with caution. This is true both when one wants to draw scientific insight for its own sake, and when one wants to use the inferred models to make computational social engineering decisions. Indeed, the correctness guarantees for algorithms are predicated on the assumption of correctness of the model and the inferred parameters. When this assumption fails — which is inevitable — the utility of the algorithms' output is compromised. Thus, to make good on the claims of real-world relevance of computational social science, *it is imperative that the research community focus on robustness as a primary design goal.* Recently, Yadav

et al. [54] carried out a field study to test the performance of Influence Maximization algorithms in real-world homeless-youth social networks. They showed that the performance of Influence Maximization algorithms differs significantly across mathematical diffusion models and complex real-world diffusion phenomena.

In this article, we study the Influence Maximization problem in the presence of noise and uncertainty about the diffusion models and data. To be more specific, we are guided by the following three overarching questions:

(1) How do the noise and uncertainty impact the performance of Influence Maximization algorithms?
(2) How can we diagnose the instability *algorithmically*?
(3) How can we design Influence Maximization algorithms that are *robust* to noise and uncertainty?

First, we exhibit simple inputs on which even very small estimation errors may mislead every algorithm into highly suboptimal solutions. To quantify the impact of noise and uncertainty on Influence Maximization algorithms, we propose (in Section 4) the Perturbation Interval Model as a framework to characterize and measure the stability of Influence Maximization approaches in the presence of noise. We empirically examine the impact of noise on Influence Maximization algorithms under multiple real-world social networks with different diffusion models and settings. To diagnose the instability of a specific Influence Maximization instance, we formulate and analyze (in Section 5) a clean algorithmic question which we term the Influence Difference Maximization problem. A theoretical analysis of the problem shows that the objective function of Influence Difference Maximization is NP-hard to approximate to within a factor of $O(n^{1-\epsilon})$ for any $\epsilon > 0$.

Given the infeasibility of diagnosing instability algorithmically, we instead focus on answering question (3): designing Influence Maximization algorithms that are *robust* to noise and uncertainty. We define (in Section 6) a Robust Influence Maximization framework wherein an algorithm is presented with a set of influence functions, typically derived from different influence models or different parameter settings for the same model. The different parameter settings could be derived from observed cascades on different topics, under different conditions, or at different times. The algorithm's goal is to identify a set of k nodes who are simultaneously influential for all influence functions, compared to the (function-specific) optimum solutions.

We show strong approximation hardness results for this problem unless the algorithm gets to select at least a logarithmic factor more seeds than the optimum solution. However, when enough extra seeds may be selected, we show that techniques of Krause et al. can be used to approximate the optimum robust influence to within a factor of $1 - 1/e$. We evaluate this bicriteria approximation algorithm against natural heuristics on several real-world data sets. Our experiments indicate that the worst-case hardness does not necessarily translate into bad performance on real-world data sets; all algorithms perform fairly well.

2 RELATED WORK

Based on the early work of Domingos and Richardson [18], Kempe et al. [34] formally defined the problem of finding a set of influential individuals as a discrete optimization problem, proposing a greedy algorithm with a $1 - 1/e$ approximation guarantee for the Independent Cascade [22, 23] and Linear Threshold [30] models. A long sequence of subsequent work focused on more efficient algorithms for Influence Maximization (both with and without approximation guarantees) and on broadening the class of models for which guarantees can be obtained [8, 14, 34, 36, 43, 50, 51, 53]. See the book by Chen et al. [12] and the survey in [34] for more detailed overviews.

As a precursor to maximizing influence, one needs to infer the influence function σ from observed data. The most common approach is to estimate the parameters of a particular diffusion model [1, 24, 25, 27, 45, 47, 48]. Theoretical bounds on the required sample complexity for many diffusion models have been established, including [1, 45, 47] for the Discrete-Time Independent Cascade (DIC) model, [27] for the Continuous-Time Independent Cascade (CIC) model, and [45] for the Linear Threshold model. However, it remains difficult to decide which diffusion models fit the observation best. Moreover, the diffusion models only serve as a rough approximation to the real-world diffusion process. In order to sidestep the issue of diffusion models, Du et al. [19] proposed to directly learn the influence function σ from the observations, without assuming any particular diffusion model. They only assume that the influence function is a weighted average of coverage functions. While their approach provides polynomial sample complexity, they require a strong technical condition on finding an accurate approximation to the reachability distribution. Hence, their work remains orthogonal to the issue of Robust Influence Maximization.

Several papers take first steps toward Influence Maximization under uncertainty. Goyal, Bonchi and Lakshmanan [29] and Adiga et al. [2] study *random* (rather than adversarial) noise models, in which either the edge activation probabilities $\theta_{u,v}$ are perturbed with random noise [29], or the presence/absence of edges is flipped with a known probability [2]. Neither of the models truly extends the underlying diffusion models, as the uncertainty can simply be absorbed into the probabilistic activation process. A detailed reduction is discussed in Section 4.2.

Another approach to dealing with uncertainty is to carry out multiple influence campaigns, and to use the observations to obtain better estimates of the model parameters. Chen et al. [15] model the problem as a combinatorial multi-armed bandit problem and use the UCB1 algorithm with regret bounds. Lei et al. [38] instead incorporate beta distribution priors over the activation probabilities into the DIC model. They propose several strategies to update the posterior distributions and give heuristics for seed selection in each trial so as to balance exploration and exploitation. Our approach is complementary: the study of stability tries to answer the question whether explorations are necessary to achieve good performance. Even in an exploration-based setting, there will always be residual uncertainty to be handled via robust algorithms, in particular when exploration budgets are limited.

A more general line of recent research focuses on the problem of learning and optimizing submodular functions (and beyond) under noise and uncertainty [5, 6, 11, 31]. Balkanski et al. study submodular function maximization from samples. They show that there is no algorithm that maximizes a general submodular function from only a polynomial number of samples [6, 31]. In a subsequent paper, they showed that submodular functions with bounded curvature can be optimized from samples and propose an optimal algorithm in terms of its approximation guarantee [5]. However, the submodular objective functions of Influence Maximization do not have bounded curvature. Recently, Balkanski, Immorlica and Singer [4] show that constant-factor approximation algorithms can be obtained for maximizing influence functions from samples under the Independent Cascade model when the network follows a Stochastic Block model. The main idea of their algorithm is to utilize the community structure of the social network. In addition to estimating the first-order marginal contribution of a node, they estimate the second-order marginal contribution to estimate the overlap in influence. The goal is to avoid selecting two nodes in the same community. Their approximation guarantee depends on strong assumptions about the network structure. Moreover, they assume a model in which an oracle will provide the exact value of the influence function without noise. Chen et al. study the general question of non-convex function optimization under uncertainty in a mini-max sense very similar to our setting [11]. We discuss the relation with their work in the next paragraph.

Chen et al. [13] and Lowalekar et al. [42] study the Robust Influence Maximization problem under the Perturbation Interval model which we introduce in the Section 4. The main result of Chen et al. [13] is an analysis of the heuristic of choosing the best solution among three candidates: make each edge's parameter as small as possible, as large as possible, or equal to the middle of its interval. They prove *solution-dependent* approximation guarantees for this heuristic.

The objective of Lowalekar et al. [42] is to minimize the maximum regret, i.e., the difference between the inferred influence function and the true one, instead of maximizing the minimum ratio between the inferred influence function and the true one. They propose a heuristic based on constraint generation ideas to solve the Robust Influence Maximization problem. The basic idea of the algorithm is to iterate between the following two steps. The first step finds the influence function with maximum regret and adds it to the set of possible influence functions Σ; the second step finds a set of seeds that maximizes influence robustly among all influence functions in Σ. Both steps are solved via mixed-integer programming, and the algorithm terminates when the maximum regret in the first step is smaller than a predefined threshold. The heuristic does not come with approximation guarantees; instead, [42] proposes a solution-dependent measure of robustness of a given seed set. As part of their work, [42] prove a result similar to our Lemma 6.2, showing that the worst-case instances all have the largest or smallest possible values for all parameters.

Chen et al. [11] study the general problem of robustly optimizing for a set of non-convex functions in the mini-max sense; hence, Robust Influence Maximization is a special case. They propose an algorithm which reduces robust optimization to a Bayesian optimization problem with an iterative weighting scheme of all the objective functions, an approach that has some similarities to the constraint generation approach of Lowalekar et al. [42]. A priori, their algorithm generates a distribution over many solutions; the robust approximation guarantee is for the expectation over the returned distribution. By sampling a logarithmic number of solutions and taking their union, their algorithm also implies a similar bicriteria result to ours. Although the work of Chen et al. [11] and our algorithm achieve similar approximation guarantee, the two algorithms take quite different algorithmic approaches.

3 PRELIMINARIES

The social network is modeled by a directed graph $G = (V, E)$ on n nodes. All parameters for non-existing edges are assumed to be 0. We first describe models of influence diffusion, and then formally define the problem of Influence Maximization.

3.1 Influence Diffusion Models

For concreteness, we focus on three diffusion models: the discrete-time Independent Cascade model (DIC) [34], the discrete-time Linear Threshold model (DLT) [34] and the continuous-time Independent Cascade model (CIC) [25]. Our framework can be generalized to most other diffusion models as well. While our hardness results are limited to Independent Cascade models (DIC and CIC) and a generalization to the Linear Threshold model is not immediate, our Robust Influence Maximization algorithm works for any diffusion model whose influence function is submodular.

Under all three models, nodes are either *active* or *inactive*. The set of active nodes at time t is denoted by A_t. Active nodes can influence other nodes and thereby cause them to become active, too.

Under the DIC model, the diffusion process unfolds in discrete time steps as follows: when a node u becomes active in step t, it attempts to activate all currently inactive neighbors in step $t + 1$. For each neighbor v, it succeeds with a known probability $\theta_{u,v}$; the $\theta_{u,v}$ are the parameters of the model. If node u succeeds, v becomes active. Once u has made all its attempts, it does not get to

make further activation attempts at later times; of course, the node v may well be activated at time $t+1$ or later by some node other than u.

Under the DLT model, each node v has a threshold η_v drawn independently and uniformly from the interval $[0,1]$. The diffusion under the DLT model unfolds in discrete time steps: A node v becomes active at step t if the total incoming weight from its active neighbors exceeds its threshold: $\sum_{u \in N(v) \cap A_{t-1}} \theta_{u,v} \geq \eta_v$ where the weight $\theta_{u,v}$ is the influence strength parameter associated with edge $e = (u,v)$.

Different from the DIC and DLT models, the CIC model describes a continuous-time process. Associated with each edge (u,v) is a delay distribution with parameter $\theta_{u,v}$. When a node u becomes newly active at time t_u, for every neighbor v that is still inactive, a delay time $d_{u,v}$ is drawn from the delay distribution. $d_{u,v}$ is the duration it takes u to activate v, which could be infinite (if u does not succeed in activating v). Commonly assumed delay distributions include the Exponential distribution or Rayleigh distribution. If multiple nodes $u_1, \ldots, u_\ell$ attempt to activate v, then v is activated at the earliest time $\min_i t_{u_i} + d_{u_i,v}$. Nodes are considered activated by the process if they are activated within a specified observation window $[0, T_s]$.

3.2 Influence Maximization Problem

A specific instance of an Influence Maximization problem is described by the class of its influence model (such as DIC, DLT, CIC, or others not discussed here in detail) and the setting of the model's parameters; the parameters are the influence probabilities under the DIC model, the edge weights under the DLT model, and the parameters of the edge delay distributions under the CIC model, respectively. Together, they completely specify the dynamic process; and thus a mapping σ from initially active sets S to the expected number[1] $\sigma(S)$ of nodes active at the end of the process. The function $\sigma(S)$ is referred to as the Influence Function. We can now formalize the Influence Maximization problem as follows:

Definition 3.1 (Influence Maximization). Maximize the objective $\sigma(S)$ subject to the constraint $|S| \leq k$.

ALGORITHM 1: Hill-Climbing Greedy Algorithm

1: Initialize: $S_0 \leftarrow \emptyset$
2: **for** $i = 1, \ldots, k$ **do**
3: Let u be the element maximizing the marginal gain $\sigma(S_{i-1} \cup \{u\}) - \sigma(S_{i-1})$.
4: Let $S_i \leftarrow S_{i-1} \cup \{u\}$.
5: **end for**
6: Return S_k

For most of the diffusion models studied in the literature, including all three models studied in this paper, it has been shown that $\sigma(S)$ is a monotone and submodular[2] function of S. These properties imply that a hill-climbing greedy approximation algorithm shown in Algorithm 1 guarantees a $1 - 1/e$ approximation [46].

[1]The model and virtually all results in the literature extend straightforwardly when the individual nodes are assigned non-negative importance scores.
[2]Recall that a set function f is monotone iff $f(S) \leq f(T)$ whenever $S \subseteq T$, and is submodular iff $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$ whenever $S \subseteq T$.

4 MODELING UNCERTAINTY IN INFLUENCE MAXIMIZATION

The concerns discussed in Section 1.1 combine to lead to significant uncertainty about the function σ: different models give rise to very different functional forms of σ, and missing observations or approximations in inference lead to uncertainty about the models' parameters.

To model this uncertainty, we assume that the algorithm is presented with a set Σ of influence functions, and assured that one of these functions actually describes the influence process, but not told *which* one. The set Σ could be finite or infinite. A finite Σ could result from a finite set of different information diffusion models that are being considered, or from a finite number of different contexts under which the individuals were observed (e.g., multiple topic-specific networks inferred from cascades on different topics), or from a finite number of different inference algorithms or algorithm settings being used to infer the model parameters from observations.

4.1 Perturbation Interval model

A particularly natural way of deriving a set Σ of influence functions arises when each model parameter is only known to lie within some given interval. For each of the edges (u, v), we are given an interval $I_{u,v} = [\ell_{u,v}, r_{u,v}] \subseteq [0, 1]$ with $\theta_{u,v} \in I_{u,v}$. For the DLT model, to ensure that the resulting influence functions are always submodular, we require that $\sum_{u \in N(v)} \theta_{u,v} \leq 1$ for all nodes v. We write $\boldsymbol{\Theta} = \times_{(u,v)\in E} I_{u,v}$ for the set of all allowable parameter settings. It is guaranteed that the ground truth parameter values satisfy $\boldsymbol{\theta'} \in \boldsymbol{\Theta}$; subject to this requirement, the ground truth parameters can be chosen arbitrarily; in particular, this includes choosing them adversarially. We term this model the *Perturbation Interval Model* or in short PIM.

For a fixed diffusion model such as DIC or DLT, the parameter values $\boldsymbol{\theta}$ determine an instance of the Influence Maximization problem. We will usually be explicit about indicating the dependence of the objective function on the parameter settings. We write $\sigma_{\boldsymbol{\theta}}$ for the objective function obtained with parameter values $\boldsymbol{\theta}$, and only omit the parameters when they are clear from the context. Therefore, under the PIM, $\Sigma = \{\sigma_{\boldsymbol{\theta}} | \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$; in particular, the set Σ is uncountable. For a given setting of parameters $\boldsymbol{\theta}$, we will denote by $S^*_{\sigma_{\boldsymbol{\theta}}} \in \operatorname{argmax}_S \sigma_{\boldsymbol{\theta}}(S)$ a solution maximizing the expected influence.

4.2 Stochastic vs. Adversarial Models

The classic Influence Maximization problem assumes that Σ contains only the true influence function to be optimized. Under uncertainty, there is more than one candidate in the set Σ. In this work, we assume an adversarial model. That is, the ground truth function to be optimized is chosen by an adversary from the set Σ. The process of influence maximization under uncertainty can be considered as a two-player game. The first player observes the candidate set Σ and picks a seed set S, trying to maximize the influence. Then, the second player picks an influence function σ from Σ to minimize the algorithm's performance. The algorithm's performance is (in our work) typically the ratio between the influence $\sigma(S)$ of the selected seed set and the influence $\sigma(S^*_\sigma)$ of the optimal seed set S^*_σ under the chosen function σ. Given its prominent role in our model, the decision to treat the choice of influence function as adversarial rather than stochastic deserves some discussion.

First, adversarial guarantees are stronger than stochastic guarantees, and will lead to more robust solutions in practice. Perhaps more importantly, inferring a Bayesian prior over influence functions in Σ will run into exactly the type of problem we are trying to address in the first place: data are sparse and noisy, and if we infer an incorrect prior, it may lead to very suboptimal results. Doing so would next require us to establish robustness over the values of the *hyperparameters* of the Bayesian prior over functions.

Specifically for the Perturbation Interval model, one may be tempted to treat the parameters as drawn according to some distribution over their possible range. This approach was essentially taken in [2, 29]. Goyal et al. [29] assume that for each edge (u, v), the value of $\theta_{u,v}$ is perturbed with uniformly random noise from a known interval. Adiga et al. [2] assume that each edge (u, v) that was observed to be present is actually absent with some probability ϵ, while each edge that was not observed is actually present with probability ϵ; in other words, each edge's presence is independently flipped with probability ϵ.

The standard Independent Cascade model subsumes both models straightforwardly. Suppose that a decision is to be made about whether u activates v. In the model of Goyal et al., we can first draw the actual (perturbed) value of $\theta'_{u,v}$ from its known distribution; subsequently, u activates v with probability $\theta'_{u,v}$; in total, u activates v with probability $\mathbb{E}\left[\theta'_{u,v}\right]$. Thus, we obtain an instance of the IC model in which all edge probabilities $\theta_{u,v}$ are replaced by $\mathbb{E}\left[\theta'_{u,v}\right]$. In the special case when the noise has mean 0, this expectation is exactly equal to $\theta_{u,v}$, which explains why Goyal et al. observed the noise to not affect the outcome at all.

In the model of Adiga et al., we first determine whether the edge is actually present; when it was observed present, this happens with probability $1 - \epsilon$; otherwise with probability ϵ. Subsequently, the activation succeeds with probability p. ([2] assumed uniform probabilities.) Thus, the model is an instance of the IC model in which the activation probabilities on all observed edges are $p(1 - \epsilon)$, while those on unobserved edges are $p\epsilon$. This reduction explains the theoretical results obtained by Adiga et al.

More fundamentally, practically all "natural" random processes that independently affect edges of the graph can be "absorbed into" the activation probabilities themselves; as a result, random noise does not at all play the result of actual noise.

To model the type of issues one would expect to arise in real-world settings, at the very least, noise must be correlated between edges. For instance, certain subpopulations may be inherently harder to observe or have sparser data to learn from. In the extreme case, certain subpopulations may not be observable at all due to their online privacy settings or offline evasive behavior. Correlated random noise would result in a more complex description of the noise model, and thus make it harder to actually learn and verify the noise model. In particular, as discussed above, this would apply given that the noise model itself must be learned from noisy data.

5 STABILITY OF INFLUENCE MAXIMIZATION

In this section, we focus on answering the first two questions introduced in Section 1.2: understanding the impact of uncertainty on Influence Maximization algorithms and diagnosing the stability of algorithms. We study the problem under the DIC and DLT models, with perturbations of the edge parameters following the Perturbation Interval model. We start our discussion with an artificial instance with two cliques to demonstrate the existence of instability. We then formally define the Influence Difference Maximization (IDM) problem in order to measure/characterize the stability of an Influence Maximization instance.

5.1 Can Instability Occur?

Suppose that we have inferred all parameters $\theta_{u,v}$, but are concerned that they may be slightly off: in reality, the influence probabilities are $\theta'_{u,v} \approx \theta_{u,v}$. Are there instances in which a seed set S that is very influential with respect to the $\theta_{u,v}$ may be much less influential with respect to the $\theta'_{u,v}$? It is natural to suspect that this might not occur: when the objective function σ varies sufficiently smoothly with the input parameters (e.g., for linear objectives), small changes in the parameters

only lead to small changes in the objective value; therefore, optimizing with respect to a perturbed input still leads to a near-optimal solution.

However, the objective σ of Influence Maximization does not depend on the parameters in a smooth way. To illustrate the issues at play, consider the following instance of the DIC model. The social network consists of two disjoint bidirected cliques K_n, and $\theta_{u,v} = \hat{\theta}$ for all u, v in the same clique; in other words, for each directed edge, the same activation probability $\hat{\theta}$ is observed. The algorithm gets to select exactly $k = 1$ node. Notice that because all nodes look the same, any algorithm essentially chooses an arbitrary node, which may as well be from Clique 1.

Let $\hat{\theta} = 1/n$ be the sharp threshold for the emergence of a giant component in the Erdős-Rényi Random Graph $G(n, p)$. It is well known [7, 21] that the largest connected component of $G(n, p)$ has size $O(\log n)$ for any $p \leq \hat{\theta} - \Omega(1/n)$, and size $\Omega(n)$ for any $p \geq \hat{\theta} + \Omega(1/n)$. Thus, if unbeknownst to the algorithm, all true activation probabilities in Clique 1 are $p \leq \hat{\theta} - \Omega(1/n)$, while all true activation probabilities in Clique 2 are $p' \geq \hat{\theta} + \Omega(1/n)$, the algorithm only activates $O(\log n)$ nodes in expectation, while it could have reached $\Omega(n)$ nodes by choosing Clique 2. Hence, small adversarial perturbations to the input parameters can lead to highly suboptimal solutions from any algorithm. The example reveals a close connection between the stability of a DIC instance and the question whether a uniform activation probability p lies close to the edge percolation threshold of the underlying graph. Characterizing the percolation threshold of families of graphs has been a notoriously hard problem. Successful characterizations have only been obtained for very few specific classes (such as d-dimensional grids [35] and d-regular expander graphs [3]). Therefore, it is unlikely that a clean characterization of stable and unstable instances can be obtained. The connection to percolation also reveals that the instability was not an artifact of having high node degrees. By the result of Alon et al. [3], the same behavior will be obtained if both components are d-regular expander graphs, since such graphs also have a sharp percolation threshold.

5.2 Influence Difference Maximization

The example of the two cliques shows that there exist *unstable instances*, in which an optimal solution to the observed parameters is highly suboptimal when the observed parameters are slightly perturbed compared to the true parameters. Of course, not every instance of Influence Maximization is unstable: for instance, when the probability $\hat{\theta}$ in the Two-Clique instance is bounded away from the critical threshold of $G(n, p)$, the objective function varies much more smoothly with $\hat{\theta}$. This motivates the following algorithmic question, which is the main focus of this section: *Given an instance of Influence Maximization, can we diagnose efficiently whether it is stable or unstable?*

Under the Perturbation Interval model, an instance $(I_{u,v})_{u,v}$ of Influence Maximization is the interval of the parameters which are subject to perturbations. We say that the instance is *stable* if $|\sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S)|$ is small for all pairs of objective functions $\sigma_{\boldsymbol{\theta}}, \sigma_{\boldsymbol{\theta}'}$ induced by legal probability settings[3], and for all seed sets S of size k. Here, "small" is defined relative to the objective function value $\sigma_{\boldsymbol{\theta}}(S^*_{\sigma_{\boldsymbol{\theta}}})$ of the optimum set.

When $|\sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S)|$ is actually small compared to $\sigma_{\boldsymbol{\theta}}(S^*_{\sigma_{\boldsymbol{\theta}}})$ for all sets S, a user can have confidence that his optimization result will provide decent performance guarantees even if his input was perturbed. The converse is of course not necessarily true: even in unstable instances, a solution that was optimal for the observed input *may* still be very good for the true input parameters.

Trying to determine whether there are a function $\sigma_{\boldsymbol{\theta}'}$ and a set S for which $|\sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S)|$ is large motivates the following optimization problem: Maximize $|\sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S)|$ over all pairs of

[3]For the DIC model, a probability setting $\boldsymbol{\theta}$ is legal when $\theta_{u,v} \in [0, 1]$ for all u, v. For the DLT model, there is an additional constraint that $\sum_{u \in N(v)} \theta_{u,v} \leq 1$ for all $v \in V$.

feasible functions $\sigma_{\boldsymbol{\theta}}, \sigma_{\boldsymbol{\theta}'}$ and all sets S. That is, we are interested in the quantity

$$\max_{S} \max_{\boldsymbol{\theta}, \boldsymbol{\theta}' \in \boldsymbol{\Theta}} |\sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S)|, \tag{1}$$

where $\boldsymbol{\theta}$ denotes the observed parameter values. For two parameter settings $\boldsymbol{\theta}, \boldsymbol{\theta}'$ with $\boldsymbol{\theta} \geq \boldsymbol{\theta}'$ coordinate-wise, it is not difficult to show using a simple coupling argument that $\sigma_{\boldsymbol{\theta}}(S) \geq \sigma_{\boldsymbol{\theta}'}(S)$ for all S. Therefore, for any fixed set S, the maximum is attained by making $\boldsymbol{\theta}$ as large as possible and $\boldsymbol{\theta}'$ as small as possible. Hence, solving the following problem is sufficient to solve (1).

Definition 5.1. Given an influence model and two parameter settings $\boldsymbol{\theta}, \boldsymbol{\theta}'$ with $\boldsymbol{\theta} \geq \boldsymbol{\theta}'$ coordinate-wise, define

$$\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S) \;=\; \sigma_{\boldsymbol{\theta}}(S) - \sigma_{\boldsymbol{\theta}'}(S).$$

Given the set size k, the Influence Difference Maximization (IDM) problem is defined as follows:

$$\begin{array}{ll} \text{Maximize} & \delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S) \\ \text{subject to} & |S| = k. \end{array}$$

In this generality, the Influence Difference Maximization problem subsumes the Influence Maximization problem, by setting $\theta'_{u,v} \equiv 0$ (and thus also $\sigma_{\boldsymbol{\theta}'} \equiv 0$).

While Influence Difference Maximization subsumes Influence Maximization, whose objective function is monotone and submodular, the objective function of Influence Difference Maximization is in general neither. To see non-monotonicity, notice that $\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(\emptyset) = \delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(V) = 0$, while generally $\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S) > 0$ for some sets S.

The function is also not in general submodular. The following example shows non-submodularity for both the DIC and DLT Models. The graph has four nodes $V = \{u, v, x, y\}$ and three edges $(u, v), (v, x), (x, y)$. The edges (v, x) and (x, y) are known to have an activation probability of 1, while the edge (u, v) has an adversarially chosen activation probability in the interval $[0, 1]$. With $S = \{u\}$ and $T = \{u, x\}$, we obtain that $\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S + v) - \delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S) = |\emptyset| - |\{v, x, y\}| = -3$, while $\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(T + v) - \delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(T) = |\emptyset| - |\{v\}| = -1$, which violates submodularity.

We establish a very strong hardness result in the form of the following theorem.

Theorem 5.2. *Under the DIC model, the Influence Difference Maximization objective function $\delta_{\boldsymbol{\theta}, \boldsymbol{\theta}'}(S)$ cannot be approximated better than $n^{1-\epsilon}$ for any $\epsilon > 0$ in polynomial time unless NP $\subseteq$ ZPP.*

Proof. We establish the approximation hardness of Influence Difference Maximization without any constraint on the cardinality of the seed set S. From this version, the hardness of the constrained problem is inferred easily as follows: if any better approximation could be obtained for the constrained problem, one could simply enumerate over all possible values of k from 1 to n, and retain the best solution, which would yield the same approximation guarantee for the unconstrained problem.

We give an approximation-preserving reduction from the Maximum Independent Set problem to the Influence Difference Maximization problem. It is well known that Maximum Independent Set cannot be approximated better than $O(n^{1-\epsilon})$ for any $\epsilon > 0$ unless NP $\subseteq$ ZPP [32].

Let $G = (V, E)$ be an instance of the Maximum Independent Set problem, with $|V| = n$. We construct from G a *directed* bipartite graph G' with vertex set $V' \cup V''$. For each node $v_i \in V$, there are nodes $v'_i \in V'$ and $v''_i \in V''$. The edge set is $E' \cup E''$, where $E' = \{(v'_i, v''_j) \,|\, (v_i, v_j) \in E\}$, and $E'' = \{(v'_i, v''_i) \,|\, v_i \in V\}$. All edges of E' are known to have an activation probability of 1, while all edges of E'' have an uncertain activation probability from the interval $[0, 1]$.

The difference is maximized by making all probabilities as large as possible for one function (meaning that all edges in $E' \cup E''$ are present deterministically), while making them as small as possible for the other (meaning that exactly the edges in E' are present).

First, let S be an independent set in G. Consider the set $S' = \{v_i' \mid v_i \in S\}$. Each node v_i'' with $v_i \in S$ is reachable from the corresponding v_i' in G', but not in $(V' \cup V'', E')$, because S is independent. Hence, the objective function value obtained in Influence Difference Maximization is at least $|S|$.

Conversely, consider an optimal solution S' to the Influence Difference Maximization problem. Without loss of generality, we may assume that $S' \subseteq V'$: any node $v_j'' \in V''$ can be removed from S' without lowering the objective value. Assume that $S := \{v_i \in V \mid v_i' \in S'\}$ is not independent, and that $(v_i, v_j) \in E$ for $v_i, v_j \in S$. Then, removing v_j' from S' cannot lower the Influence Difference Maximization objective value of S': all of v_j''s neighbors in V'' contribute 0, as they are reachable using E' already; furthermore, v_j'' also does not contribute, as it is reachable using E' from v_i'. Thus, any node with a neighbor in S can be removed from S', meaning that S is without loss of generality independent in G.

At this point, all nodes $v_j'' \in V''$ with $(v_i', v_j'') \in E'$ for some i contribute 0 to the Influence Difference Maximization objective function, as do the nodes $v_i' \in S'$. Therefore, the objective value of S' is exactly the number of nodes $v_i'' \in V''$ with $v_i' \in S'$, which is exactly $|S'| = |S|$. ■

5.3 Experiments

While we saw in Section 5.1 that examples highly susceptible (with errors of magnitude $\Omega(n)$) to small perturbations exist, the goal of this section is to evaluate experimentally how widespread this behavior is for realistic social networks.

5.3.1 Experimental Setting. We carry out experiments under the DIC model, for six classes of graphs — four synthetic and two real-world. In each case, the model/data give us a simple graph or multigraph. Multigraphs are converted to simple graphs by collapsing parallel edges to a single edge with weight c_e equal to the number of parallel edges; for simple graphs, all weights are $c_e = 1$. The observed probabilities for edges are $\theta_e = c_e \cdot p$; across experiments, we vary the base probability p to take on the values $\{0.01, 0.02, 0.05, 0.1\}$. The resulting parameter vector is denoted by $\boldsymbol{\theta}$.

The uncertainty interval for e is $I_e = [(1-\Delta)\theta_e, (1+\Delta)\theta_e]$; here, Δ is an uncertainty parameter for the estimation, which takes on the values $\{1\%, 5\%, 10\%, 20\%, 50\%\}$ in our experiments. The parameter vectors $\boldsymbol{\theta}^+$ and $\boldsymbol{\theta}^-$ describe the settings in which all parameters are as large (as small, respectively) as possible.

5.3.2 Network Data. We ran experiments on four synthetic networks and two real social networks. Synthetic networks provide a controlled environment in which to compare observed behavior to expectations, while real social networks may give us indications about the prevalence of vulnerability to perturbations in real networks that have been studied in the past.

Synthetic Networks. We generated synthetic networks according to three widely used network models. The network models are: (1) the 2-dimensional grid, (2) random regular graphs, (3) the Kronecker Graph model [40] with Core-Peripheral structure (*Kronecker-CP*), (4) the Kronecker Graph model with Hierarchical Community structure (*Kronecker-HC*).

The Kronecker Graph model matches several properties of real-world networks, including power-law degree distribution, small diameter and community structure [40].

It has been shown that networks with different structures can be generated from the Kronecker Graph Model, by choosing different seed matrices [40]. We use three different 2-by-2 seed matrices with different entries, listed in Table 1. The seed matrices generate Erdős-Rényi networks, core-peripheral networks, and networks with hierarchical community structures, respectively [40].

We generated 2-dimensional grid graphs and random regular graphs with 400 nodes. The generated Kronecker Graph model graphs with both structures have 512 nodes. For all synthetic networks, we selected $k = 20$ seed nodes.

Table 1. Kronecker Graph Parameters

Name	Parameter
Erdős-Rényi	$\begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$
Core-Peripheral	$\begin{pmatrix} 0.962 & 0.107 \\ 0.107 & 0.962 \end{pmatrix}$
Hierarchical Community	$\begin{pmatrix} 0.962 & 0.535 \\ 0.535 & 0.107 \end{pmatrix}$

Real Networks. We considered two real networks to evaluate the susceptibility of practical networks: the co-authorship network *STOCFOCS* of theoretical CS papers and the retweet network *Haiti*. In all experiments, we worked with uniform edge weights p.

STOCFOCS Network. The *STOCFOCS* co-authorship network is a multigraph extracted from published papers in the conferences STOC and FOCS from 1964–2001. Each node in the network is a researcher with at least one publication in one of the conferences. For each multi-author paper, we add a complete undirected graph among the authors. Parallel edges are compressed into a single edge with corresponding weight. The resulting graph has 1768 nodes and 10024 edges.

Twitter Dataset. The second real-world cascade dataset we consider is one crawled from Twitter. It comprises a complete collection of tweets between October 2009 and January 2010. Instead of using the raw data collection, several datasets are preprocessed from the complete collection of tweets for different experimental purposes.

We extracted a set of topic-specific networks from the complete collection of tweets. We treated each hashtag as a separate cascade, and extracted the top 100/250 users with the most tweets containing these hashtags into two datasets (Twitter100 and Twitter250). The hashtags were manually grouped into five categories of about 70–80 hashtags each, corresponding to major events/topics during the data collection period. The five groups are: Haiti earthquake (Haiti), Iran election (Iran), Technology, US politics, and the Copenhagen climate change summit (Climate). Examples of hashtags in each group are shown in Table 2. Whenever user B retweeted a post of user A with a hashtag belonging to category i, we inserted an edge from A to B in graph i. The extracted networks in Twitter100/250 have 100 and 250 nodes, respectively.

Our decision to treat each hashtag as a separate cascade is supposed to capture that most hashtags "spread" across Twitter when one user sees another use it, and starts posting with it himself. The grouping of similar hashtags captures that a user who may influence another to use a hashtag, say, #teaparty, would likely also influence the other user to a similar extent to use, say, #liberty. Limiting the data sets to the most active users was necessary because most users had exhibited very limited activity.

Table 2. Examples of hashtags in each category

Category	Hashtags
Iran	#iranelection, #iran, #16azar, #tehran
Haiti	#haiti, #haitiquake, #supphaiti, #cchaiti
Technology	#iphone, #mac, #microsoft, #tech
US politics	#obama, #conservative, #teaparty, #liberty
Climate	#copenhagen, #cop15, #climatechange

For the experiments in this section, we use the extracted 250-node network with the topic "Haiti;" we refer to this dataset as *Haiti*. The other topic specific-networks are used for experiments in Section 6.

5.3.3 Algorithms. Our experiments necessitate the solution of two algorithmic problems: Finding a set of size k of maximum influence, and finding a set of size k maximizing the *influence difference*. The former is a well-studied problem, with a monotone submodular objective function. We simply used the widely known $1 - 1/e$ approximation algorithm introduced in Section 3.2, which is best possible unless P=NP.

For the goal of Influence Difference Maximization, we established in Theorem 5.2 that the objective function is hard to approximate better than a factor $O(n^{1-\epsilon})$ for any $\epsilon > 0$. For experimental purposes, we used the *Random Greedy* algorithm of Buchbinder et al. [9].

The running time of the Random Greedy Algorithm is $O(kC|V|)$, where C is the time required to estimate $g(S \cup \{u\}) - g(S)$. In our case, the objective function is #P-hard to evaluate exactly [16, 52], but arbitrarily close approximations can be obtained by Monte Carlo simulation. Since each simulation takes time $O(|V|)$, when running $M = 2000$ iterations of the Monte Carlo simulation in each iteration, the overall running time of the algorithm is $O(kM|V|^2)$.

A common technique for speeding up the greedy algorithm for maximizing a submodular function is the CELF heuristic of Leskovec et al. [41]. When the objective function is submodular, the standard greedy algorithm and CELF obtain the same result. However, when it is not (as is the case here), the results may be different. We ran the Random Greedy algorithm both with and without the CELF heuristic. The single exception is the largest input, the STOCFOCS network. (Here, the greedy algorithm without CELF did not finish in a reasonable amount of time.) For all networks other than STOCFOCS, the results using CELF are not significantly different from the reported results without the CELF optimization. For STOCFOCS, we therefore only report the result including the CELF heuristic.

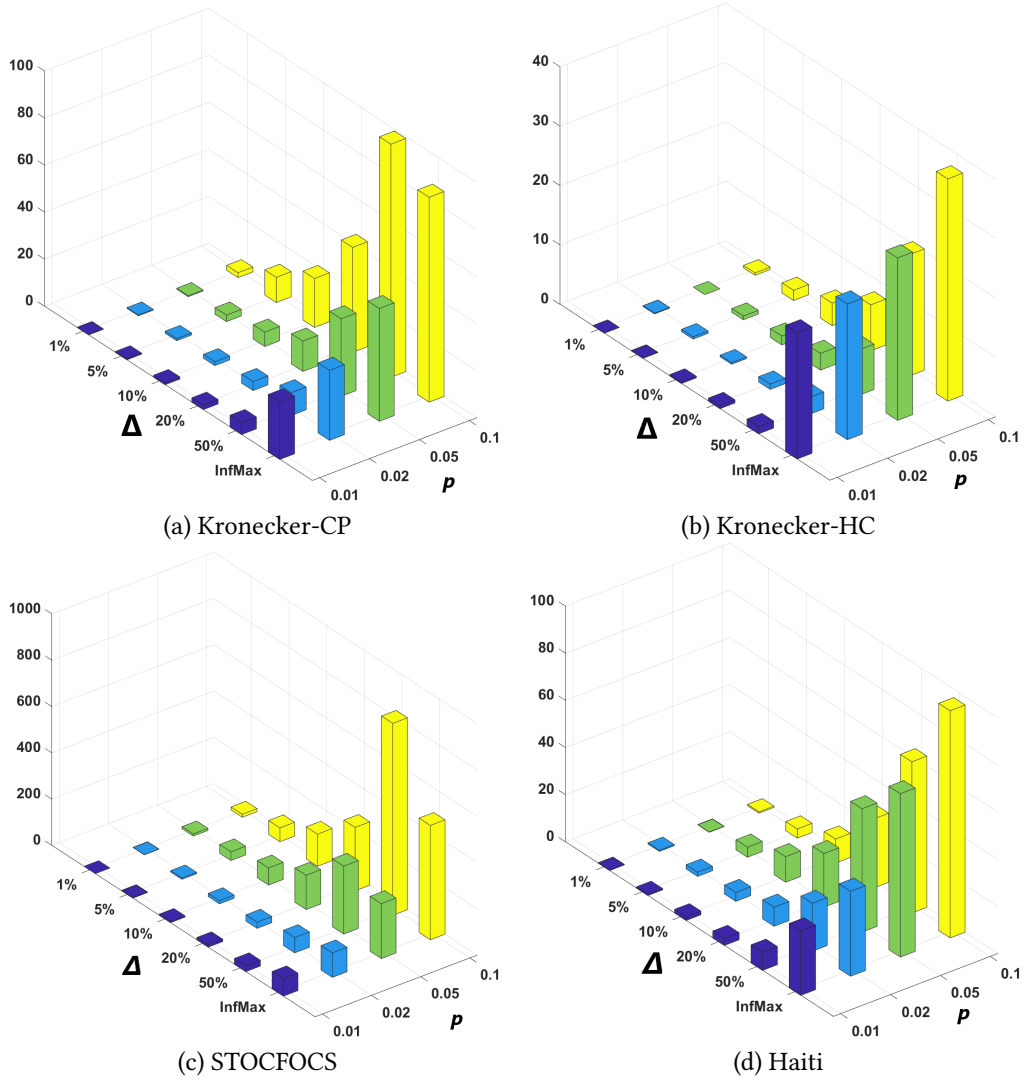
5.3.4 Results. In all our experiments, the results for the Grid and Small-World network were sufficiently similar that we omit the results for grids here. As a first sanity check, we empirically computed $\max_{S:|S|=1} \delta_{\theta^+,\theta^-}(S)$ for the *complete graph* on 200 nodes with $I_e = [1/200 \cdot (1-\Delta), 1/200 \cdot (1+\Delta)]$ and $k = 1$. According to the analysis in Section 5.1, we would expect extremely high instability. The results, shown in Table 3, confirm this expectation.

Table 3. Instability for the clique K_{200}.

Δ	σ_{θ^+}	σ_{θ^-}
50%	66.529	1.955
20%	23.961	4.253
10%	15.071	6.204

Next, Figure 1 shows the (approximately) computed values $\max_{S:|S|=k} \delta_{\theta^+,\theta^-}(S)$, and — for calibration purposes — $\max_{S:|S|=k} \sigma_\theta(S)$ for all networks and parameter settings. Notice that the result is obtained by running the Random Greedy algorithm without any approximation guarantee. However, as the algorithm's output provides a lower bound on the maximum influence difference, a large value suggests that Influence Maximization could be unstable. On the other hand, small values *do not* guarantee that the instance is stable.

While individual networks vary somewhat in their susceptibility, the overall trend is that larger estimates of baseline probabilities θ make the instance more susceptible to noise, as do (obviously) larger uncertainty parameters Δ. In particular, for $\Delta \geq 20\%$, the noise (after scaling) dominates



(a) Kronecker-CP (b) Kronecker-HC

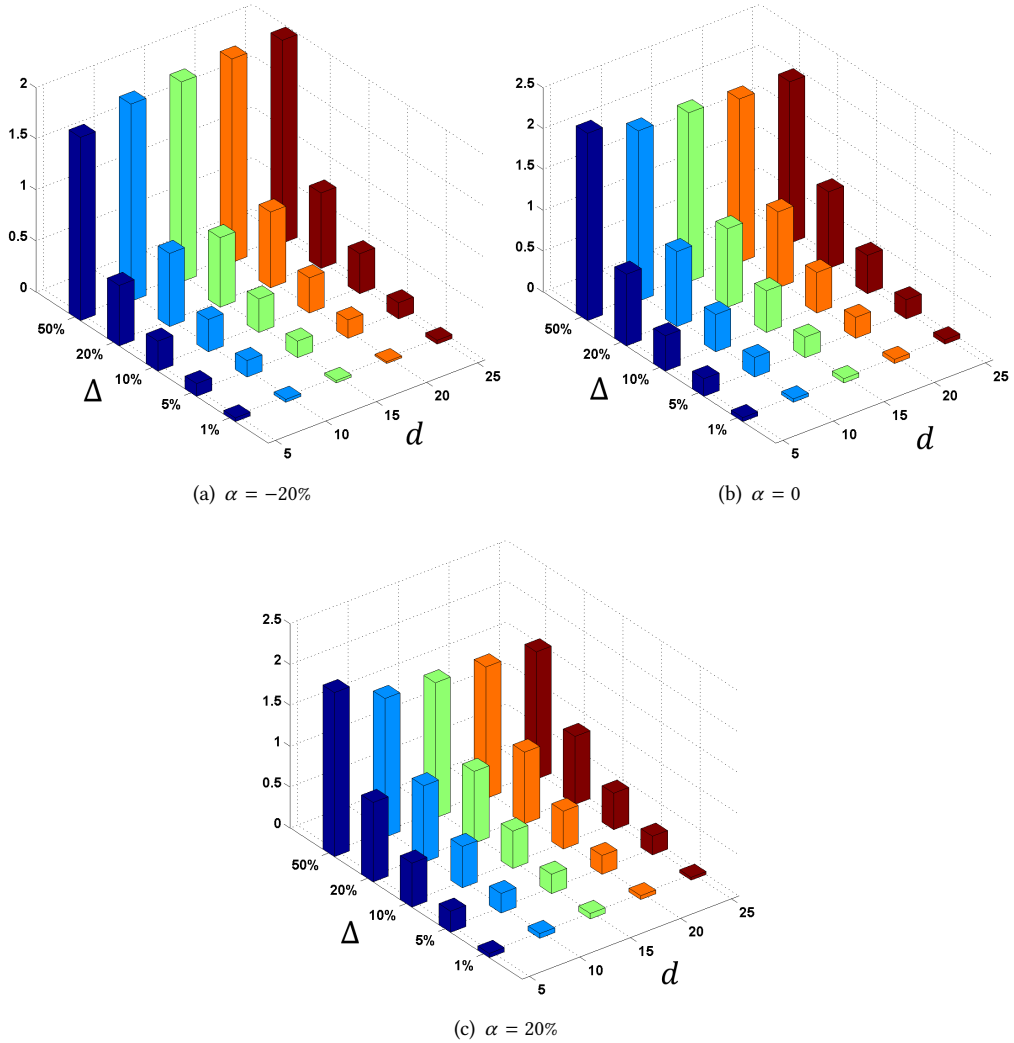
(c) STOCFOCS (d) Haiti

Fig. 1. Comparison between Influence Difference Maximization and Influence Maximization results for four different networks. (The result for the STOCFOCS network is obtained with CELF optimization.)

the Influence Maximization objective function value, meaning that optimization results should be used with care. The only exception are the *Kronecker-HC* networks, which are overall much less susceptible to uncertainty. The reason is that the networks tend to be much sparser than the other networks; for example, the size of cascades is roughly half of that in the *Kronecker-CP* network.

Next, we evaluate the dependence of the noise tolerance on the degrees of the graph, by experimenting with random d-regular graphs whose degrees vary from 5 to 25. It is known that such graphs are expanders with high probability, and hence have percolation thresholds of $1/d$ [3]. Accordingly, we set the base probability to $(1+\alpha)/d$ with $\alpha \in \{-20\%, 0, 20\%\}$. We use the same setting for uncertainty intervals as in the previous experiments. Figure 2 shows the ratio

between Influence Difference Maximization and Influence Maximization, i.e., $\frac{\max_S \delta_{\theta^+,\theta^-}(S)}{\max_S \sigma_\theta(S)}$, with $\alpha \in \{-20\%, 0, 20\%\}$. It indicates that for random regular graphs, the degree does not appear to significantly affect stability, and that again, noise around 20% begins to pose a significant challenge.



(a) $\alpha = -20\%$

(b) $\alpha = 0$

(c) $\alpha = 20\%$

Fig. 2. Ratio between the computed values of Influence Difference Maximization and Influence Maximization under random regular graphs with different degree.

Moreover, we observe that the ratio reaches its minimum when the edge activation probability is exactly at the percolation threshold $1/d$. This result is in line with percolation theory and also the analysis of Adiga et al. [2].

As a general takeaway message, for larger amounts of noise (even just a relative error of 20%) — which may well occur in practice — a lot of caution is advised in using the results of algorithmic Influence Maximization.

6 ROBUST INFLUENCE MAXIMIZATION

The fact that our main theorem for diagnosing instability is negative (i.e., a strong approximation hardness result) is somewhat disappointing, in that it rules out reliably categorizing data sets as stable or unstable. This motivates our investigation of the third question: setting up a *robust* Influence Maximization framework wherein an algorithm is presented with a set of influence functions derived from different influence models or different parameter settings for the same model. The main motivation for our work is that often, σ is not precisely known to the algorithm trying to maximize influence. There may be a (possibly infinite as under the PIM in Section 4.1) number of candidate functions σ, resulting from different diffusion models or parameter settings.

Since the algorithm does not know which $\hat{\sigma} \in \Sigma$ is the ground truth influence function, in the Robust Influence Maximization problem, it must "simultaneously optimize" for all objective functions in Σ, in the sense of maximizing $\rho(S) = \min_{\hat{\sigma} \in \Sigma} \frac{\hat{\sigma}(S)}{\hat{\sigma}(\hat{S})}$, where $\hat{S} \in \operatorname{argmax}_S \hat{\sigma}(S)$ is an optimal solution knowing which function $\hat{\sigma}$ is to be optimized. In other words, the selected set should simultaneously get as close as possible to the optimal solutions for all possible objective functions. That is, the algorithm's goal is to identify a set of k nodes who are simultaneously influential for all influence functions, compared to the (function-specific) optimum solutions.

To be more specific, our study is guided by the following overarching questions:

(1) How well can the objective ρ be optimized in principle?
(2) How well do simple heuristics perform in theory?
(3) How well do simple heuristics perform in practice?
(4) How do robustly and non-robustly optimized solutions differ qualitatively?

We address these questions as follows. First, we show (in Section 6.2.2) that unless the algorithm gets to exceed the number of seeds k by at least a factor $\ln|\Sigma|$, approximating the objective ρ to within a factor $O(n^{1-\epsilon})$ is NP-hard for all $\epsilon > 0$.

However, when the algorithm does get to exceed the seed set target k by a factor of $\ln|\Sigma|$ (times a constant), much better bicriteria approximation guarantees can be obtained. Specifically, we show that a modification of an algorithm of Krause et al. [37] uses $O(k \ln|\Sigma|)$ seeds and finds a seed set whose influence is within a factor $(1 - 1/e)$ of optimal.

We also investigate two straightforward heuristics:

(1) Run a greedy algorithm to optimize ρ directly, picking one node at a time.
(2) For each objective function $\sigma \in \Sigma$, find a set S_σ (approximately) maximizing $\sigma(S_\sigma)$. Evaluate each of these sets under $\rho(S_\sigma)$, and keep the best one.

We first exhibit instances on which both of the heuristics perform very poorly. Next (in Section 6.3), we focus on more realistic instances, exemplifying the types of scenarios under which robust optimization becomes necessary. In the first set of experiments, we infer influence networks on a fixed node set from Twitter cascades on different *topics*. Individuals' influence can vary significantly based on the topic, and for a previously unseen topic, it is not clear which inferred influence network to use. In additional sets of experiments, we derive data sets from the same MemeTracker data [39], but use different *time slices*, different *inference algorithms and parametrizations*, and different samples from confidence intervals.

The main outcome of the experiments is that while the algorithm with robustness as a design goal typically (though not even always) outperforms the heuristics, the margin is often quite small. Hence, heuristics may be viable in practice, when the influence functions are reasonably similar. A

visual inspection of the nodes chosen by different algorithms reveals how the robust algorithm "hedges its bets" across models, while the non-robust heuristic tends to cluster selected nodes in one part of the network.

6.1 Problem Definition

For concreteness, we focus on two diffusion models: the DIC model and the CIC model introduced in Section 3.1. Our framework applies to most other diffusion models; in particular, most of the concrete results carry over to the discrete and continuous-time Linear Threshold models [34, 49].

A specific instance is described by the class of its influence model (such as DIC, CIC, or others not discussed here in detail) and the setting of the model's parameters; in the DIC and CIC models above, the parameters $\theta_{u,v}$ are the activation probabilities and the parameters of the delay distributions, respectively.

We now formally define the *Robust Influence Maximization* problem.

Definition 6.1 (Robust Influence Maximization). Given a set Σ of influence functions, maximize the objective

$$\rho(S) = \min_{\sigma \in \Sigma} \frac{\sigma(S)}{\sigma(S^*_\sigma)},$$

subject to a cardinality constraint $|S| \leq k$. Here S^*_σ is a seed set with $|S^*_\sigma| \leq k$ maximizing $\sigma(S^*_\sigma)$.

A solution to the Robust Influence Maximization problem achieves a large fraction of the maximum possible influence (compared to the optimal seed set) under all diffusion settings simultaneously. Alternatively, the solution can be interpreted as solving the Influence Maximization problem when the function σ is chosen from Σ by an adversary.

While Definition 6.1 per se does not require the $\sigma \in \Sigma$ to be submodular and monotone, these properties are necessary to obtain positive results. Hence, we will assume here that all $\sigma \in \Sigma$ are monotone and submodular, as they are for standard diffusion models. Notice that even then, ρ is the minimum of submodular functions, and as such not necessarily submodular itself [37].

A particularly natural and important special case of Definition 6.1 is the Perturbation Interval model we considered in Section 4.1. Here, the influence model is known (for concreteness, DIC), but there is uncertainty about its parameters. For each edge e, we have an interval $I_e = [\ell_e, r_e]$, and the algorithm only knows that the parameter (say, θ_e) lies in I_e; the exact value is chosen by an adversary. Notice that Σ is (uncountably) infinite under this model. While this may seem worrisome, the following lemma shows that we only need to consider finitely (though exponentially) many functions:

Lemma 6.2. *Under the Perturbation Interval model for DIC[4], the worst case for the ratio in ρ for any seed set S is achieved by making each θ_e equal to ℓ_e or r_e.*

Proof. Fix one edge $\hat{e}$, and consider an assignment (fixed for now) $\theta_e \in I_e$ of activation probabilities to all edges $e \neq \hat{e}$. Let $x \in I_{\hat{e}}$ denote the (variable) activation probability for edge $\hat{e}$. First, fix any seed set S, and define $f_S(x)$ to be the expected number of nodes activated by S when the activation probabilities of all edges $e \neq \hat{e}$ are θ_e and the activation probability of $\hat{e}$ is x.

We express $f_S(x)$ using the triggering set approach introduced in [34]. Let $\mathcal{G}$ be the set of all possible directed graphs on the given node set V. For any graph G, let $R_G(S)$ be the number of nodes reachable from S in G via a directed path, and let $P(G)$ be the probability that graph G is

[4]The result carries over with a nearly identical proof to the Linear Threshold model. We currently do not know if it also extends to the CIC model.

obtained when each edge e is present in G independently with probability θ_e (or x, if $e = \hat{e}$). By the triggering set technique [34, Proof of Theorem 4.5], we get that

$$f_S(x) = \sum_{G \in \mathcal{G}} P(G) \cdot R_G(S).$$

The probabilities $P(G)$ for obtaining a graph G are:

$$P(G) = (1-x) \cdot \prod_{e \in G} \theta_e \cdot \prod_{e \notin G, e \neq \hat{e}} (1-\theta_e) \qquad \text{when } \hat{e} \notin G;$$

$$P(G) = x \cdot \prod_{e \in G, e \neq \hat{e}} \theta_e \cdot \prod_{e \notin G} (1-\theta_e) \qquad \text{when } \hat{e} \in G.$$

In either case, we obtain a linear function of x, so that $f_S(x)$, being a sum of linear functions, is also linear in x.

Therefore, the function $g(x) := \max_S f_S(x)$, being a maximum of linear functions of x, is convex and piecewise linear. Consider any fixed seed set S, and the ratio $h(x) := \frac{f_S(x)}{g(x)}$. Its α-level set $\{x \mid h(x) \geq \alpha\}$ is equal to $\{x \mid g(x) - 1/\alpha \cdot f_S(x) \leq 0\}$. Because $g(x) - 1/\alpha \cdot f_S(x)$, a convex function minus a linear function, is convex, its 0-level set is convex. Hence, all α-level sets of h are convex, and h is quasi-concave.

Because h is quasi-concave, it is unimodal, and thus minimized at one of the endpoints of the interval. Hence, we can minimize the ratio $h(x)$ — and thus the performance of the seed set S— by making x either as small or as large as possible. By repeating this argument for all edges $\hat{e}$ one by one, we arrive at an influence setting minimizing the performance of S, and in which all influence probabilities are equal to the left or right endpoint of the respective interval $I_{\hat{e}}$. ■

Lowalekar et al. [42] have proved a similar result to Lemma 6.2. The difference lies in the objective of Robust Influence Maximization. We model the problem as maximizing the minimum ratio $\rho(S)$ while Lowalekar et al. consider minimizing the maximum regret $\max_{\sigma \in \Sigma} \sigma(S^*_\sigma) - \sigma(S)$. They prove that the maximum regret is achieved by making each θ_e equal to ℓ_e or r_e similarly by showing that the maximum regret is also quasi-concave.

6.2 Algorithm and Hardness

Even when Σ contains just a single function σ, Robust Influence Maximization is exactly the traditional Influence Maximization problem, and is thus NP-hard. This issue also appears in a more subtle way: *evaluating* $\rho(S)$ (for a given S) involves taking the minimum of $\frac{\sigma(S)}{\sigma(S^*_\sigma)}$ over all $\sigma \in \Sigma$. It is not clear how to calculate the ratio $\frac{\sigma(S)}{\sigma(S^*_\sigma)}$ even for one of the σ, since the scaling constant $\sigma(S^*_\sigma)$ (which is independent of the chosen S) is exactly the solution to the original Influence Maximization problem, and thus NP-hard to compute.

This problem, however, is fairly easy to overcome: instead of using the true optimum solutions S^*_σ for the scaling constants, we can compute $(1-1/e)$-approximations S^g_σ using the greedy algorithm, because the σ are monotone and submodular. Then, because $(1-1/e) \cdot \sigma(S^*_\sigma) \leq \sigma(S^g_\sigma) \leq \sigma(S^*_\sigma)$ for all $\sigma \in \Sigma$, we obtain that the "greedy objective function"

$$\rho^g(S) = \min_{\sigma \in \Sigma} \frac{\sigma(S)}{\sigma(S^g_\sigma)},$$

satisfies the following property for all sets S:

$$(1-1/e) \cdot \rho^g(S) \leq \rho(S) \;\leq\; \rho^g(S). \tag{2}$$

Hence, optimizing $\rho^g(S)$ in place of $\rho(S)$ comes at a cost of only a factor $(1-1/e)$ in the approximation guarantee. We will therefore focus on solving the problem of (approximately) optimizing $\rho^g(S)$.

Because each σ is monotone and submodular, and the $\sigma(S^g_\sigma)$, just like the $\sigma(S^*_\sigma)$, are just scaling constants, $\rho^g(S)$ is a minimum of monotone submodular functions. However, we show that even in the context of Influence Maximization, this minimum is impossible to approximate to within any polynomial factor. This holds even in a bicriteria sense, i.e., the algorithm's solution is allowed to pick $(1-\delta)\ln|\Sigma|\cdot k$ nodes, but is compared only to solutions using k nodes. The result also extends to the seemingly more restricted Perturbation Interval model, giving an almost equally strong bicriteria approximation hardness result there.

Theorem 6.3. *Let $\delta, \epsilon > 0$ be any constants, and assume that $P \neq NP$. There are no polynomial-time algorithms for the following problems:*

(1) *Given n nodes and a set Σ of influence functions on these nodes (derived from the DIC or CIC models), as well as a target size k. Find a set S of $|S| \leq (1-\delta)\ln|\Sigma|\cdot k$ nodes, such that $\rho(S) \geq \rho(S^*)\cdot\Omega(1/n^{1-\epsilon})$, where S^* is the optimum solution of size k.*
(2) *Given a graph G on n nodes and intervals I_e for edge activation probabilities under the DIC model (or intervals I_e for edge delay parameters under the CIC model), as well as a target size k. Find a set S of cardinality $|S| \leq \epsilon\cdot c\cdot\ln n\cdot k$ (for a sufficiently small fixed constant c) such that $\rho(S) \geq \rho(S^*)\cdot\Omega(1/n^{1-\epsilon})$, where S^* is the optimum solution of size k.*

Proof. We prove the two parts of the theorem by (slightly different) reductions from the gap version of Set Cover. A Set Cover instance consists of a universe $U = \{a_1, \ldots, a_N\}$, a collection $\mathcal{T}$ of M subsets of U, and an integer k. A set cover is a collection $C \subseteq \mathcal{T}$ such that $\bigcup_{T\in C} T = U$. Without loss of generality, we assume that each element is contained in at least one set — otherwise, there trivially is no set cover. Also, without loss of generality, we assume that $k \leq \min(M, N)$, as otherwise, one can trivially pick all sets or one designated set per element.

The gap version of Set Cover then asks us to decide whether there is a set cover C of size $|C| \leq k$ or whether each set cover has size at least $(1-\delta)\ln N\cdot k$. (The algorithm is promised that the minimum size will never lie between these two values.) Dinur and Steurer [17, Corollary 1.5] showed that the gap version of Set Cover is NP-hard.

Part 1. Based on the Set Cover instance, we construct the following instance of Robust Influence Maximization under the DIC model. Let $m := (\max(N, M))^{3/\epsilon}$. The instance consists of N bipartite graphs on a shared vertex set $V = X \cup Y$. X contains one node x_T for each set $T \in \mathcal{T}$; Y contains m nodes $y_{a,1}, \ldots, y_{a,m}$ for each element $a \in U$. Hence, the number of nodes in the constructed graph is $n = M + mN = \Theta(mN) \leq \Theta(m^{1+\epsilon/3})$; in particular, it is polynomial, and the reduction takes polynomial time.

In the i^{th} influence function, all nodes x_T with $T \ni a_i$ have a directed edge with activation probability 1 (or exponential delay distribution with delay parameter 1) to all of the $y_{a_i,j}$ (for all j); no other edges are present. Hence, $|\Sigma| = N$, and $\ln N = \ln|\Sigma|$. For the CIC model, the time window has size $T_s = NM$.

First, consider the case when there is a set cover C of size k. Choose the corresponding $x_T, T \in C$ as seed nodes, and call the resulting seed set S. Because C is a set cover, in the i^{th} instance, all of the $y_{a_i,j}$ are activated, for a total of at least $m + k$ nodes. (Under the CIC model, all of these $y_{a_i,j}$ are activated with high probability, not deterministically, within T steps.) Because none of the nodes in X and none of the $y_{a_{i'},j}, i' \neq i$ have incoming edges in the i^{th} instance, the optimum solution for that instance can activate at most all of the m nodes $y_{a_i,j}$ and its k selected nodes, for a total of $m + k$. Thus, the objective function value will be 1 (or arbitrarily close to 1 w.h.p. for the CIC model).

Now assume that there is no set cover of size $(1-\delta)\ln N\cdot k$, and consider any seed set S. Let $k' = |S \cap X| \leq (1-\delta)\ln N\cdot k$ be the number of nodes from X selected as seeds. Because the set

$\mathcal{S} := \{T \in \mathcal{T} \mid x_T \in S\}$ cannot be a set cover by assumption, there must be some $a_i \notin \bigcup_{T\in\mathcal{S}} T$. Therefore, under the i^{th} influence function, none of the $y_{a_i,j}$ can be ever activated, except those selected directly in S. Hence, the number of nodes activated under the i^{th} influence function is at most $|S| \leq (1-\delta)\ln N \cdot k$. On the other hand, by selecting just one node x_T corresponding to any set $T \ni a_i$, one could have activated all of the $y_{a_i,j}$ (with high probability under the CIC model), for a total of m. Thus, the objective function value is at most $\rho(S) \leq \frac{(1-\delta)\ln N\cdot k}{m} \leq O(m^{2\epsilon/3-1}) \leq O(n^{\frac{2\epsilon-3}{3+\epsilon}}) = o(n^{-(1-\epsilon)})$, where we crudely bounded both $\ln N$ and k by $N \leq O(m^{\epsilon/3})$.

Hence, a $((1-\delta)\ln N, O(n^{1-\epsilon}))$ bicriteria approximation algorithm could distinguish the two cases, and thus solve the gap version of Set Cover.

Part 2. For the second part, we just consider the gap version with a fixed δ, say, $\delta = \frac{1}{2}$. Then, in the hard instances, M and N are polynomially related, which we assume here, i.e., $M \leq N^q$ for some constant q which is independent of ϵ or N.

Based on the Set Cover instance, we construct a different Robust Influence Maximization instance, consisting of a directed graph with three layers $V = X \cup Y \cup Z$. The first layer again contains one node x_T for each set $T \in \mathcal{T}$; the second layer now contains just one node y_a for each element $a \in U$. There is an edge (with known influence probability 1, or exponential delay distribution with parameter 1) from x_T to y_a if and only if $a \in T$. The third layer Z contains $m = (\max(N,M))^{2/\epsilon}$ nodes. For each $a \in U$ and $z \in Z$, there is a directed edge (y_a, z) with complete uncertainty about its parameter: under the DIC model, the probability is in the interval $I_{(y_a,z)} = [0,1]$, and under the CIC model, the edge delay is exponentially distributed with parameter in the interval $I_{(y_a,z)} = (0,1]$. In total, the graph has $n = N + M + m = \Theta(m)$ nodes (in particular, polynomially many), and the reduction takes polynomial time. Because N is at most polynomially smaller than M, we have $N = \Omega(n^{\epsilon/2q})$, and thus $\ln N = \Omega(\frac{\epsilon}{q} \cdot \ln(n))$. For the CIC model, we set the time horizon to $T_s = NM$.

First, consider the case when there is a set cover C of size k. Consider choosing the corresponding $x_T, T \in C$ as seed nodes; call the resulting seed set S. S will definitely activate all nodes in Y, for a total of $k + N$. Now, consider any assignment of probabilities $\theta_{y_a,z}$ or edge delays $d_{y_a,z}$ to the edges from Y to Z, and an optimal seed set S^* of size k. Let $Z^* = Z \cap S^*$ be the set of seed nodes chosen from Z, of size k'. Then, S^* definitely activates all of Z^*, and at most all N nodes from Y as well as $k - k'$ nodes from X, for a total (so far) of $N + k$. For any node $z \in Z \setminus Z^*$, the probability that it is activated by S is at least as large as under S^*, because for any values of the individual activation probabilities or delays between Y and Z, the fact that S activates all of Y ensures that any node in Z activated under S^* is also activated under S (by time T, in the case of the CIC model). Because the expected number of nodes activated from $Z \setminus Z^*$ is at least as large under S as under S^* (for all settings of the activation probabilities or edge delay parameters), we get that $\rho(S) \geq 1$.

Now assume that there is no set cover of size $\frac{1}{2}\ln N \cdot k$, and consider any seed set S. If S contained any node y_a, we could replace it with any node x_T such that $a \in T$ and activate at least as many nodes as before, so assume without loss of generality that $S \cap Y = \emptyset$. Because $|S \cap X| \leq |S| \leq \frac{1}{2}\ln N \cdot k$, the gap guarantee implies that there is at least one node $y_a \in Y$ that is never activated by S. Now consider the probability assignment $\theta_{y_a,z} = 1$ for all $z \in Z$, and $\theta_{y,z} = 0$ for all $y \neq y_a, z \in Z$. (Under the CIC model, set $d_{y_a,z} = 1$ for all $z \in Z$, and $d_{y,z} = 1/(NM)^2$ for all $y \neq y_a, z \in Z$.) Then, the seed set S cannot activate any nodes in Z (except those it may have selected), and will activate a total of at most $N + k = O(N) = O(n^{\epsilon/2})$ nodes. (Under the CIC model, this statement holds with high probability.) On the other hand, the seed set $\{y_a\}$ (just a single node) would have activated all of Z (with high probability, under the CIC model), for a total of $m + 1 = \Omega(n)$ nodes. Hence, the ratio is at most $O(n^{\epsilon/2}/n) = O(1/n^{1-\epsilon/2})$, implying that $\rho(S) \leq O(1/n^{1-\epsilon/2})$.

If there were an $(\epsilon \cdot c \cdot \ln(n), O(n^{1-\epsilon}))$ bicriteria approximation algorithm for a sufficiently small constant c, it could distinguish which of the two cases $(\rho(S) = 1, \rho(S) \leq O(1/n^{1-\epsilon/2}))$ applied, thus solving the gap version of Set Cover. ■

The hardness results naturally apply to any diffusion model that subsumes the DIC or CIC models. However, an extension to the DLT model is not immediate: the construction relies crucially on having many edges of probability 1 into a single node, which is not allowed under the DLT model.

6.2.1 Bicriteria Approximation Algorithm. Theorem 6.3 implies that to obtain any non-trivial approximation guarantee, one needs to allow the algorithm to exceed the seed set size by at least a factor of $\ln|\Sigma|$. In this section, we therefore focus on such bicriteria approximation results, by slightly modifying an algorithm of Krause et al. [37]. The difference is that we use the Greedy Mintss algorithm [28] (with pseudo code in Algorithm 3) to solve the submodular coverage subproblem.

After we proposed the bicriteria approximation algorithm [33], [11] designed an alternative algorithm with similar guarantees. Their algorithm first generates a distribution over many solution sets by reducing robust optimization to a Bayesian optimization problem with an iterative weighting scheme. Then, by sampling a logarithmic number of solutions and taking their union, their algorithm implies a similar bicriteria result to ours.

The high-level idea of the algorithm is as follows. Fix a real value c, and define $h_\sigma^{(c)}(S) := \min(c, \frac{\sigma(S)}{\sigma(S_\sigma^g)})$ and $H^{(c)}(S) := \sum_{\sigma\in\Sigma} h_\sigma^{(c)}(S)$. Then, $\rho^g(S) \geq c$ if and only if $\frac{\sigma(S)}{\sigma(S_\sigma^g)} \geq c$ for all $\sigma \in \Sigma$. But because by definition, $h_\sigma^{(c)}(S) \leq c$ for all σ, the latter is equivalent to $H^{(c)}(S) \geq |\Sigma| \cdot c$. (If any term in the sum is less than c, no other term can ever compensate for it, because they are capped at c.)

Because $H^{(c)}(S)$ is a non-negative linear combination of the monotone submodular functions $h_\sigma^{(c)}$, it is itself a monotone and submodular function. This enables the use of a greedy $\ln|\Sigma|$-approximation algorithm to find an (approximately) smallest set S with $H^{(c)}(S) \geq c|\Sigma|$. If S has size at most $k\ln|\Sigma|$, this constitutes a satisfactory solution, and we move on to larger values of c. If S has size more than $k\ln|\Sigma|$, then the greedy algorithm's approximation guarantee ensures that there is no satisfactory set S of size at most k. Hence, we move on to smaller values of c. For efficiency, the search for the right value of c is done with binary search and a specified precision parameter.

A slight subtlety in the greedy algorithm is that $H^{(c)}$ could take on fractional values. Thus, instead of trying to meet the bound $c|\Sigma|$ precisely, we aim for a value of $c|\Sigma| - \epsilon$. Then, the analysis of the Greedy Mintss algorithm of Goyal et al. [28] (of which our algorithm is an unweighted special case) applies. The resulting algorithm Saturate Greedy is given as Algorithm 2. The simple greedy subroutine — a special case of the Greedy Mintss algorithm — is given as Algorithm 3.

The slight difference between our algorithm and the algorithm of Krause et al. [37] lies in how the submodular coverage subproblem is solved. Both [37] and the Greedy Mintss algorithm [28] greedily add elements. However, the Greedy Mintss algorithm adds elements until the desired submodular objective is attained up to an additive ε term, while [37] requires exact coverage. That is, the while loop in line 3 in Algorithm 3 exits only if $f(S) \geq \eta$ in [37]. Moreover, directly considering real-valued submodular functions instead of going through fractional values leads to a more direct analysis of the Greedy Mintss algorithm [28].

By combining the discussion at the beginning of this section (about optimizing ρ vs. ρ^g) with the analysis of Krause et al. [37] and Goyal et al. [28] (the following theorem), we obtain the following approximation guarantee.

ALGORITHM 2: SATURATE GREEDY (Σ, k, precision γ)

Initialize $c_{\min} \leftarrow 0$, $c_{\max} \leftarrow 1$.
while $(c_{\max} - c_{\min}) \geq \gamma$ **do**
  $c \leftarrow (c_{\max} + c_{\min})/2$.
  Define $H^{(c)}(S) \leftarrow \sum_{\sigma \in \Sigma} \min(c, \frac{\sigma(S)}{\sigma(S^g_\sigma)})$.
  $S \leftarrow$ GREEDY MINTSS$(H^{(c)}, k, c \cdot |\Sigma|, c \cdot \gamma/3)$.
  if $|S| > \beta \cdot k$ **then**
    $c_{\max} \leftarrow c$.
  else
    $c_{\min} \leftarrow c \cdot (1 - \gamma/3)$, $S^* \leftarrow S$.
  end if
end while
return S^*.

ALGORITHM 3: GREEDY MINTSS (f, k, threshold η, error ε)

Initialize $S \leftarrow \emptyset$.
while $f(S) < \eta - \varepsilon$ **do**
  $u \leftarrow \text{argmax}_{v \notin S} f(S \cup \{v\})$.
  $S \leftarrow S \cup \{u\}$.
end while
return S.

THEOREM 6.4 (THEOREM 1 IN [28]). *Let $\varepsilon > 0$ be any shortfall and let S be the solution of GREEDY MINTSS with chosen threshold $\eta - \varepsilon$. Then $|S| \leq |S^*| \cdot (1 + \ln \frac{\eta}{\varepsilon})$ where S^* is the optimal solution of the submodular set cover problem.*

THEOREM 6.5. *Let $\gamma > 0$ be arbitrary and $\beta = 1 + \ln |\Sigma| + \ln \frac{3}{\gamma}$. SATURATE GREEDY finds a seed set $\hat{S}$ of size $|\hat{S}| \leq \beta k$ with*

$$\rho(\hat{S}) \geq (1 - 1/e) \cdot \rho(S^*) - \gamma,$$

where $S^* \in argmax_{S:|S| \leq k} \rho(S)$ is an optimal robust seed set of size k.

Proof. Algorithm 2 uses Algorithm 3 (GREEDY MINTSS) as a subroutine to find[5] a solution S such that $f(S) \geq \eta - \varepsilon$ and $|S| \leq |S^*| \cdot (1 + \ln \frac{\eta}{\varepsilon})$, where S^* is a smallest solution guaranteeing $f(S^*) \geq \eta$.

In light of the general outline and motivation for the SATURATE GREEDY algorithm given above, it mostly remains to verify how the guarantees for GREEDY MINTSS and the balancing of the parameters carry through.

We will show that throughout the algorithm (or more precisely: the binary search), $c_{\min}$ always remains a lower bound on the solution for the problem with the relaxed cardinality constraint, while $c_{\max}$ remains an upper bound on the solution for the original problem. In other words, there

[5]Technically, the guarantees on GREEDY MINTSS depend on being able to evaluate f precisely [28, Theorem 1]. However, Theorem 2 of [28] states that by obtaining $(1 \pm \delta)$-approximations to f, we can ensure that $|S| \leq (1 + \delta')|S^*| \cdot (1 + \ln \frac{\eta}{\varepsilon})$, where $\delta' \to 0$ as $\delta \to 0$. For influence coverage functions, arbitrarily close approximations to f can be obtained by Monte Carlo simulations. We therefore ignore the issue of sampling accuracy, and perform the analysis as though f could be evaluated precisely. Otherwise, the approximations carry through in a straightforward way, leading to multiplicative factors $(1 + \delta'')$.

is no set S of cardinality at most $|S| \leq k$ with $\rho(S) > c_{\max}$, and there *is* a set S of cardinality at most $|S| \leq \beta k$ with $\rho(S) \geq c_{\min}$.

To show this claim, consider the set S returned by the Greedy Mintss algorithm. If $|S| > \beta k$, the guarantee for Greedy Mintss implies that $|S| \leq \beta |S^*|$, where S^* is the optimal solution for the instance. Because $|S^*| \geq |S|/\beta > k$, the value c is not feasible, and the algorithm is correct in setting $c_{\max}$ to c.

Otherwise, $|S| \leq \beta k$, and the guarantee of Greedy Mintss in Theorem 6.4 implies that $H^{(c)}(S) \geq c \cdot |\Sigma| - c \cdot \gamma/3$. Because each $h_\sigma^{(c)}(S) \leq c$ by definition, we get for all σ,

$$h_\sigma^{(c)}(S) \;\geq\; H^{(c)}(S) - (|\Sigma| - 1) \cdot c \;\geq\; c - c \cdot \gamma/3,$$

and therefore $\rho(S) \geq c - c \cdot \gamma/3$. This confirms the correctness of assigning $c_{\min} = c \cdot (1 - \gamma/3)$.

Since we do not set $c_{\min} = c$, we need to briefly verify termination of the binary search. For any iteration in which we update $c_{\min}$, let $\Delta := c_{\max} - c_{\min} \geq \gamma$. When the new $c'_{\min}$ is set to $c \cdot (1 - \gamma/3) \geq c - \gamma/3$, we get that $c_{\max} - c'_{\min} = (c_{\max} - c) + (c - c'_{\min}) \leq \Delta/2 + \gamma/3 \leq 5\Delta/6$. Hence, the size of the interval keeps decreasing geometrically, and the binary search terminates in $O(\log(1/\gamma))$ iterations.

At the time of termination, we obtain that $|c^* - c_{\min}| \leq \gamma$. Combining this bound with the factor of $(1 - 1/e)$ we lost due to approximating ρ with ρ^g, we obtain the claim of the theorem. ∎

Theorem 6.5 holds very broadly, so long as all influence functions are monotone and submodular. This includes the DIC, DLT, and CIC models, and allows mixing influence functions from different model classes.

Notice the contrast between Theorems 6.5 and 6.3. By allowing the seed set size to be exceeded just a little more (a factor $\ln|\Sigma| + O(1)$ instead of $0.999 \ln|\Sigma|$), we go from $\Omega(n^{1-\epsilon})$ approximation hardness to a $(1 - 1/e)$-approximation algorithm.

6.2.2 Simple Heuristics. In addition to the Saturate Greedy algorithm, our experiments use two natural baselines. The first is a simple greedy algorithm Single Greedy which adds βk elements to S one by one, always choosing the one maximizing $\rho^g(S \cup \{v\})$. While this heuristic has provable guarantees when the objective function is submodular, this is not the case for the minimum of submodular functions.

The second heuristic is to run a greedy algorithm for each objective function $\sigma \in \Sigma$ separately, and choose the best of the resulting solutions. Those solutions are exactly the sets S_σ^g defined earlier in this section. Thus, the algorithm consists of choosing $\operatorname{argmax}_{\sigma \in \Sigma} \rho^g(S_\sigma^g)$. We call the resulting algorithm All Greedy. A variant of this heuristic was also proposed by Chen et al. [13] as the LUGreedy algorithm under the Perturbation Interval Model; in their work, Σ contains only two functions: one with all edge parameters set to the maximum, the other one with all edge parameters set to the minimum.

In the worst case, both Single Greedy and All Greedy can perform arbitrarily badly, as seen by the following class of examples with a given parameter k. The example consists of k instances of the DIC model for the following graph with $3k + m$ nodes (where $m \gg k$). The graph comprises a directed complete bipartite graph $K_{k,m}$ with k nodes $x_1, \ldots, x_k$ on one side and m nodes $y_1, \ldots, y_m$ on the other side, as well as k separate edges $(u_1, v_1), \ldots, (u_k, v_k)$ between $2k$ new nodes $\{u_i\}$ and $\{v_i\}$. The edges (u_i, v_i) have activation probability 1 in all instances. In the bipartite graph, in the i^{th} scenario, only the edges leaving node x_i have probability 1, while all others have 0 activation probability.

The optimal solution for Robust Influence Maximization is to select all nodes x_i, since one of them will succeed in activating the m nodes y_j. The resulting objective value will be close to 1. However, All Greedy only picks one node x_i and the remaining $k - 1$ nodes as u_j. Single Greedy

instead picks all of the u_j. Thus, both All Greedy and Single Greedy will have robust influence close to 0 as m grows large. Empirical experiments confirm this analysis. For example, for $k = 2$ and $m = 100$, Saturate Greedy achieves $\rho = 0.985$, while Single Greedy and All Greedy only achieve 0.038 and 0.029, respectively.

Implementation. The most time-consuming step in all of the algorithms is the estimation of influence coverage, given a seed set S. Naïve estimation by Monte Carlo simulation could lead to a very inefficient implementation. The problem is even more pronounced compared to traditional Influence Maximization as we must estimate the influence in multiple diffusion settings. We use the ConTinEst algorithm of Du et al. [20] for fast influence estimation under the CIC model. For the DIC model, we generalize the approach of Du et al. To accelerate the Greedy Mintss algorithm, we also apply the CELF optimization [41] in all cases. Analytically, one can derive linear running time (in both n and $|\Sigma|$) for all three algorithms, thanks to the fast influence estimation. This is borne out by detailed experiments in Section 6.3.4.

6.3 Experiments

We empirically evaluated the Saturate Greedy algorithm and the Single Greedy and All Greedy heuristics. Our goal is twofold: (1) Evaluate how well Saturate Greedy and the heuristics perform on realistic instances. (2) Qualitatively understand the difference between robustly and non-robustly optimized solutions.

Our experiments were all performed on real-world data sets. The exception was the scalability experiments in Section 6.3.4, which benefits from the controlled environment of synthetic networks. The data sets span the range of different causes for uncertainty, namely: (1) influence functions are learned from cascades for different topics; (2) influence functions are learned with different modeling assumptions; (3) influence functions are only inferred to lie within intervals I_e (the Perturbation Interval model).

The first real-world data we consider is the Twitter dataset introduced in detail in Section 5.3.2. The second real-world cascade dataset we consider is the MemeTracker dataset.

MemeTracker Dataset. The MemeTracker dataset [39] contains memes extracted from the Blogsphere and main-stream media sites between August 2008 and February 2009. It tracks the quotes and phrases that appear most frequently over time across this entire online news spectrum. Overall, Memetracker tracks more than 17 million different phrases; about 54% of the total phrase/quote mentions appear on blogs and 46% in news media. The dataset is available for download at the SNAP Library[6].

MemeTracker Network We constructed a diffusion network from the MemeTracker dataset. We restricted ourselves to data from only August 2008 and the 500 most active users. We inferred a diffusion network among the active users from the cascades during the period, using the ConNIe algorithm [44] The resulting network together with the parameters (activation probabilities) is referred to as the *MemeTracker-DIC* network.

Time-specific Networks Besides the MemeTracker-DIC network, we extracted a set of time-specific networks according to different times of activities. We extracted the 2000/5000 sites with the most posting activity across the six-month period we study (MemeTracker2000/5000). We grouped the cascades by their publishing month, leading to six groups of cascades. We constructed six diffusion networks, one for each month, from the hyperlinks between the blog posts during the same period. That is, if site A published an article in August with a link to site B, we added an edge in the diffusion network of August.

[6]https://snap.stanford.edu/data/memetracker9.html

6.3.1 Different Networks. We first focused on the case in which the diffusion model is kept constant: we used the DIC model, with parameters specified below. Different objective functions were obtained from observing cascades (1) on different topics. We used Twitter retweet networks for different topics (Twitter100/250), discussed in depth in Section 5.3.2. (2) at different times. We used MemeTracker diffusion network snapshots at different times (MemeTracker2000/5000). The parameters of the DIC model used for this set of experiments are summarized in Table 4.

Table 4. Diffusion model settings

Data set	Edge Activation Probability	# Seeds
Twitter100	0.2	10
Twitter250	0.1	20
MemeTracker2000	0.05	50
MemeTracker5000	0.05	100

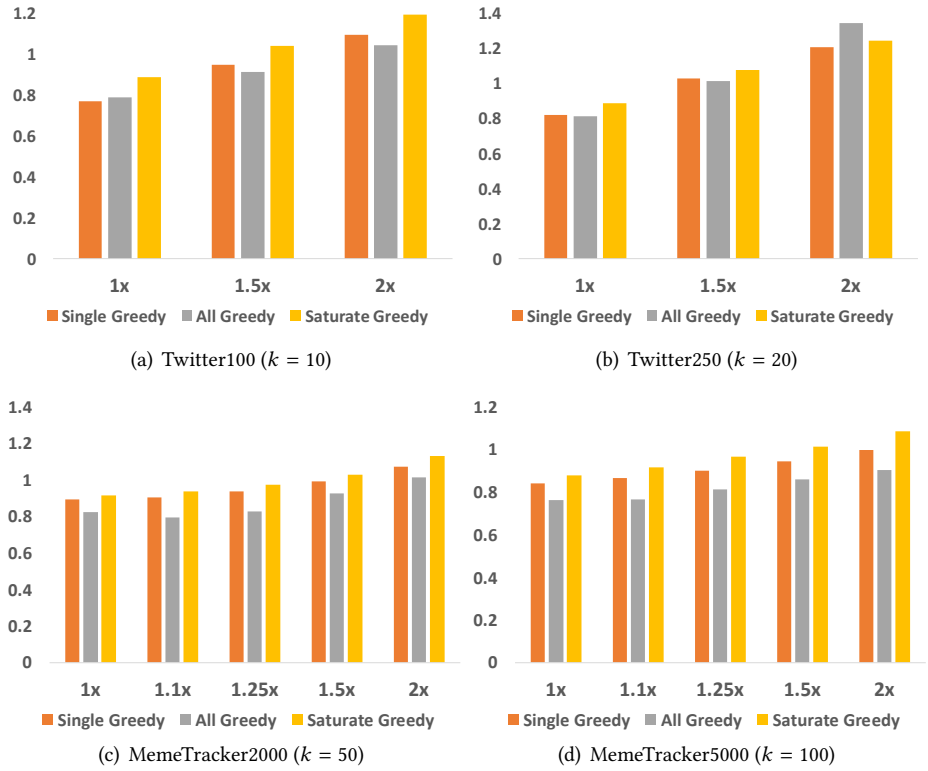
Recalling that in the worst case, a relaxation in the number of seeds is required to obtain robust seed sets, we allow all algorithms to select more seeds than the solution they are compared against. Specifically, for the Twitter100 and Twitter250 datasets, we report results in which the algorithms may select k, $1.5 \cdot k$ and $2 \cdot k$ seeds, respectively. Because the number of seeds selected is larger for the MemeTracker data sets ($k = 50$ for the MemeTracker2000 data set and $k = 100$ for the MemeTracker5000 data set), for the MemeTracker data sets, we also include two additional more fine-grained settings: letting the algorithms select $1.1 \cdot k$ and $1.25 \cdot k$ seeds. The reported results are averaged over three independent runs of each of the algorithms.

Results: Performance. The aggregate performance of the different algorithms on the four data sets is shown in Figure 3.

The first main insight is that (in the instances we studied) getting to over-select seeds by 50%, all three algorithms achieve a robust influence of at least 1.0. In other words, 50% more seeds let the algorithms perform as though they knew exactly which of the (adversarially chosen) diffusion settings was the true one. This suggests that the networks in our data sets share a lot of similarities that make influential nodes in one network also (mostly) influential in the other networks. This interpretation is consistent with the observation that the baseline heuristics performed similarly to (and in one case better than) the Saturate Greedy algorithm. Notice, however, that when selecting just k seeds, Saturate Greedy did perform best (though only by a small margin) among the three algorithms. This suggests that keeping robustness in mind may be more crucial when the algorithm does not get to compensate with a larger number of seeds.

Results: Visualization. To further illustrate the tradeoffs between robust and non-robust optimization, we visualized the seeds selected by Saturate Greedy (robust seeds) compared to seeds selected non-robustly based on only one diffusion setting. For legibility, we focus here only on the Twitter250 data set, and only plot 4 out of the 5 networks. (The fifth network is very sparse, and thus not particularly interesting.)

Figure 4 compares the seeds selected by Saturate Greedy with those (approximately) maximizing the influence for the Iran network. Notice that Saturate Greedy focused mostly (though not exclusively) on the densely connected core of the network (at the center), while the Iran-specific optimization also exploits the dense regions on the left and at the bottom. These regions are much less densely connected in the US politics and Climate networks, while the core remains fairly densely connected, leading the Saturate Greedy solution to be somewhat more robust.



(a) Twitter100 ($k = 10$)

(b) Twitter250 ($k = 20$)

(c) MemeTracker2000 ($k = 50$)

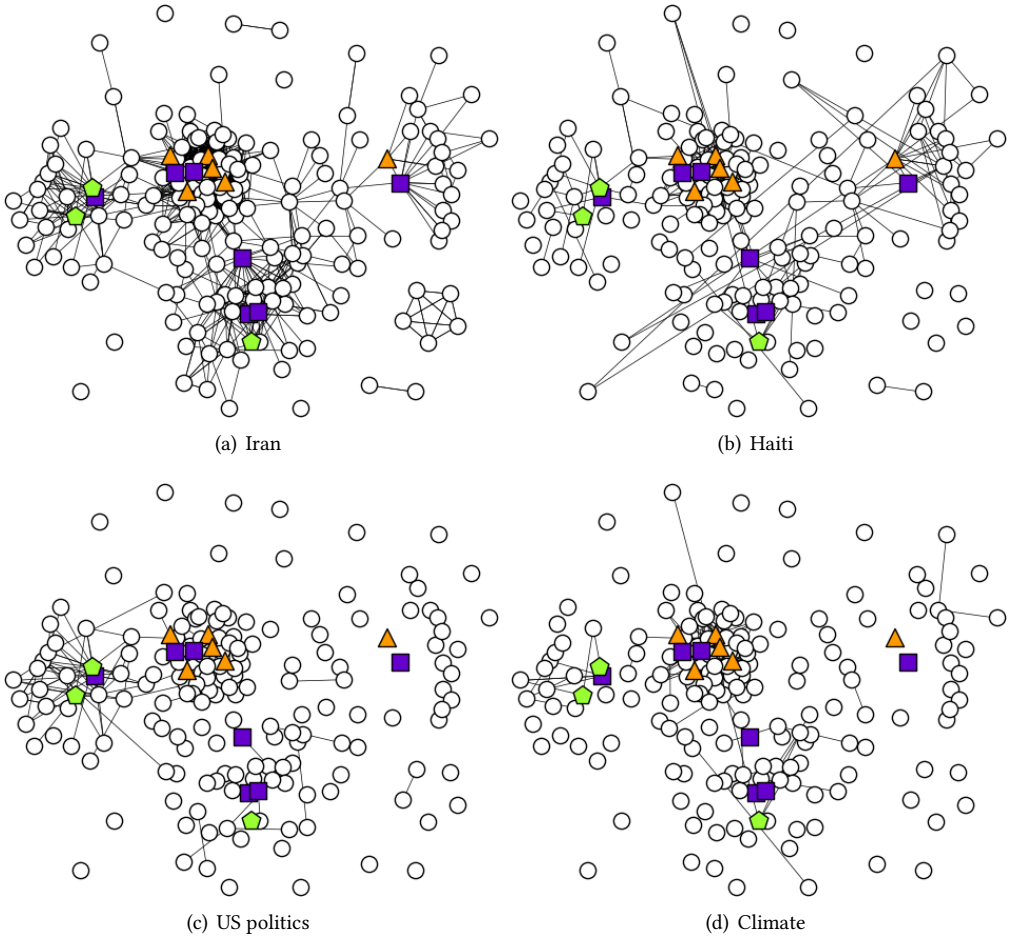
(d) MemeTracker5000 ($k = 100$)

Fig. 3. Performance of the algorithms on the four topical/temporal datasets. The x-axis is the number of seeds selected, and the y-axis the resulting robust influence (compared to seed sets of size k).

Similarly, Figure 5 compares the SATURATE GREEDY seeds (which are the same as in Figure 4) with seeds for the Climate network. The trend here is exactly the opposite. The seeds selected based only on the Climate network are exclusively in the core, because the other parts of the Climate network are barely connected. On the other hand, the robust solution picked a few seeds from the clusters at the bottom, left, and right, which are present in other networks. These seeds led to extra influence in those networks, and thus more robustness.

6.3.2 Different Diffusion Models. In choosing a diffusion model, there is little convincing empirical work guiding the choice of a model class (such as CIC, DIC, or threshold models) or of distributional assumptions for model parameters (such as edge delay). A possible solution is to optimize robustly with respect to these different possible choices.

In this section, we report on an evaluation of such an approach. Specifically, we performed two experiments: (1) learning the CIC influence network under different parametric assumptions about the delay distribution, and (2) learning the influence network under different models of influence (CIC, DIC, DLT). We again used the MemeTracker dataset, restricting ourselves to the data from August 2008 and the 500 most active users. We used the MULTITREE algorithm of Gomez-Rodriguez et al. [26] to infer the diffusion network from the observed cascades. This algorithm requires a parametric assumption for the edge delay distribution. We inferred ten different networks G_i corresponding to the Exponential distribution with parameters 0.05, 0.1, 0.2, 0.5, 1.0, and to the



(a) Iran (b) Haiti

(c) US politics (d) Climate

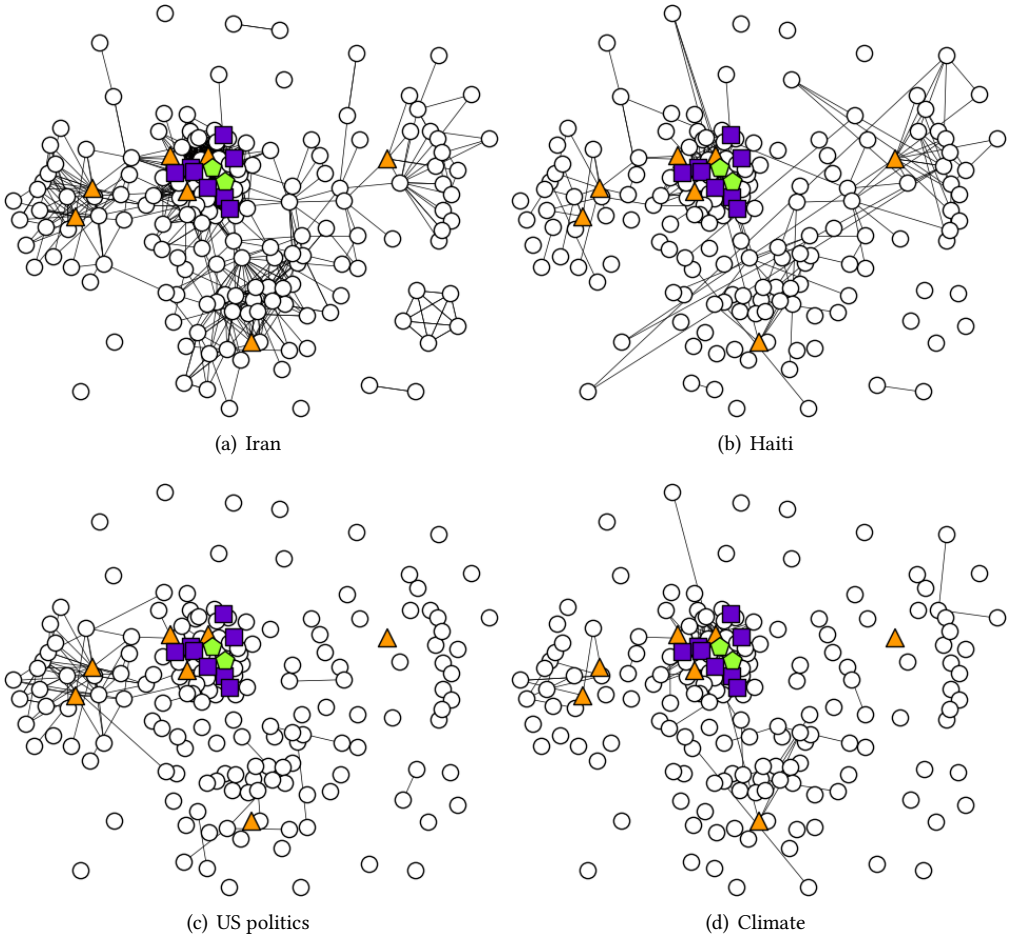
Fig. 4. Saturate Greedy vs. Iran graph seed nodes. Green/pentagon nodes were selected by both; orange/triangle nodes were selected by Saturate Greedy only; purple/square nodes for Iran only.

Rayleigh distribution with parameters 0.5, 1, 2, 3, 4. The length of the observation window was set to 1.0.

We then used the three algorithms to perform robust influence maximization for $k = 10$ seeds, again allowing the algorithms to exceed the target number of vertices. The influence model for each graph is the CIC model with the same parameters that were used to infer the graphs.

The performance of the algorithms is shown in Figure 6(a). All methods achieved satisfactory results in the experiment; this is again due to high similarity between the different diffusion settings inferred with different parameters.

For the second experiment, we investigated the robustness across different classes of diffusion models. We constructed three instances of the DIC, DLT and CIC model from the ground truth diffusion network between the 500 active users. For the DIC model, we set the activation probability uniformly to 0.1. For the DLT model, we followed [34] and set the edge weights to $1/d_v$ where d_v is the in-degree of node v. For the CIC model, we used an exponential distribution with parameter



(a) Iran (b) Haiti

(c) US politics (d) Climate

Fig. 5. Saturate Greedy vs. Climate graph seed nodes. Green/pentagon nodes were selected by both; orange/triangle nodes were selected by Saturate Greedy only; purple/square nodes for Climate only.

0.1 and an observation window of length 1.0. We performed robust influence maximization for $k = 10$ seeds and again allowed the algorithms to exceed the target number of seeds.

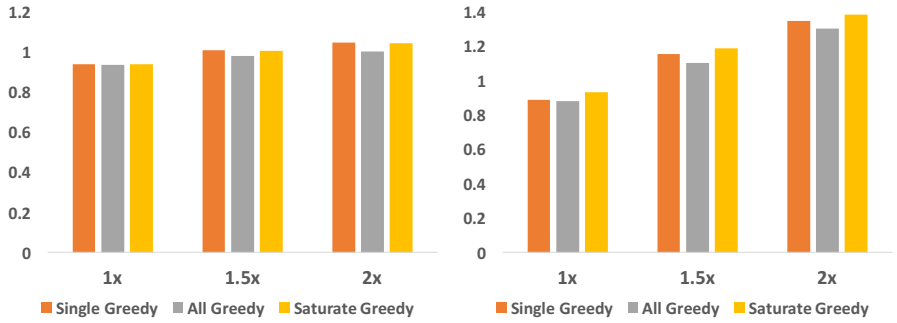
The results are shown in Figure 6(b). Similarly to the case of different estimated parameters, all methods achieved satisfactory results in the experiment due to the high similarity between the diffusion models. Our results raise the intriguing question of which types of networks would be prone to significant differences in algorithmic performance based on which model is used for network estimation.

6.3.3 Networks sampled from the Perturbation Interval model. To investigate the performance when model parameters can only be placed inside "confidence intervals" (i.e., the Perturbation Interval model), we carried out experiments under two networks, MemeTracker-DIC and STOCFOCS (introduced in Section 5.3.2).

Following the approach in Section 5, for both networks, we assigned "confidence intervals" $I_e = [(1-q)\theta_e, (1+q)\theta_e]$, where the θ_e are the inferred activation probabilities. Whenever $(1-$

$q)\theta_e > 1$, we truncated its value to 1. For experiments on the MemeTracker-DIC network, we set $q \in \{10\%, 20\%, 30\%, \ldots, 100\%\}$, while we used a coarser grid for the experiments on the large graph STOCFOCS, with $q \in \{5\%, 10\%, 20\%, 50\%, 100\%\}$.

While Lemma 6.2 guarantees that the worst-case instances have activation probabilities $(1-q)\theta_e$ or $(1+q)\theta_e$, this still leaves $2^{|E|}$ candidate functions, too many to include. We generated an instance for our experiments by sampling 10 of these functions uniformly, i.e., by independently making each edge's activation probability either $(1-q)\theta_e$ or $(1+q)\theta_e$. This collection was augmented by two more instances: one where all edge probabilities are $(1-q)\theta_e$, and one where all probabilities are $(1+q)\theta_e$. Notice that with the inclusion of these two instances, the All Greedy heuristic generalizes the LUGreedy algorithm by Chen et al. [13], but might provide strictly better solutions on the *selected* instances because it explicitly considers those additional instances. The algorithms got to select 20 seed nodes; note that in these experiments, we were not considering a bicriteria approximation.



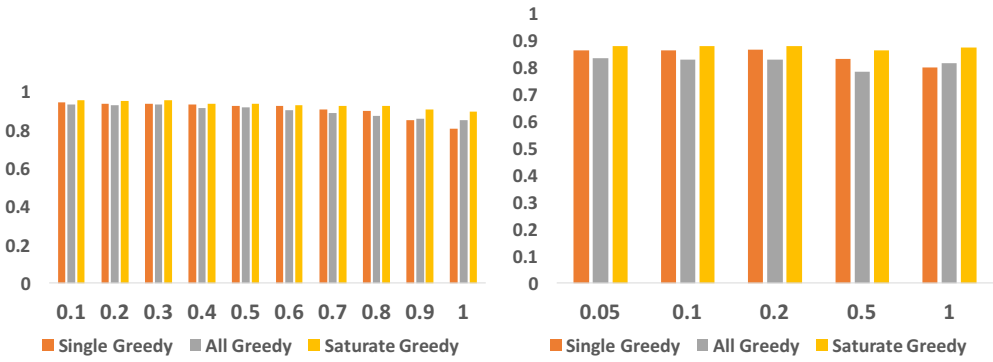
(a) Different Distributions. (b) Different Models

Fig. 6. Performance of the algorithms (a) under different delay distributions following the CIC model, and (b) under different classes of diffusion models. The x-axis shows the number of seeds selected, and $k = 10$.

The results are shown in Figures 7(a) and 7(b). Contrary to the previous results, when there was a lot of uncertainty about the edge parameters (relative interval size 100% in both networks), the Saturate Greedy algorithm more clearly outperformed the Single Greedy and All Greedy heuristics. Thus, robust optimization does appear to become necessary when there is a lot of uncertainty about the model's parameters.

Notice that the evaluation of the algorithms' seed sets was performed only with respect to the *sampled* influence functions, not with respect to all $2^{|E|}$ functions. Whether one can efficiently identify a worst-case parameter setting for a given seed set S is an intriguing open question. Absent this ability, we cannot efficiently guarantee that the solutions are actually good with respect to all parameter settings.

6.3.4 Scalability. To evaluate the scalability of the algorithms, we departed from real-world data sets in order to obtain a controlled environment. We generated networks using the Kronecker graph model [40]: specifically, we generated Erdős-Rényi networks, core-peripheral networks and networks with hierarchical community structure. For each type, we generated a set of 5, 10, 15, 20, 25 networks of sizes 128, 256, ..., 4096. We used the DIC model with activation probability set to 0.1, and selected $k = 50$ nodes. The running times of the three algorithms are shown in Figure 8 and Figure 9. In Figure 8, we fixed the number of networks to five and varied the size of each network;



(a) Perturbation Interval model: MemeTracker ($k = 20$) (b) Perturbation Interval model: STOCFOCS ($k = 50$)

Fig. 7. Performance of the algorithms under networks sampled from the Perturbation Interval model: (a) MemeTracker-DIC network; (b) STOCFOCS network. (The x axis shows the (relative) size of the perturbation interval I_e).

in Figure 9, we fixed the size of the networks to 1024 and varied the number of networks. The graphs show that the heuristics are faster than the Saturate Greedy algorithm by about a factor of ten, but all three algorithms scale linearly both in the size of the graph and the number of networks, due to the fast influence estimation method.

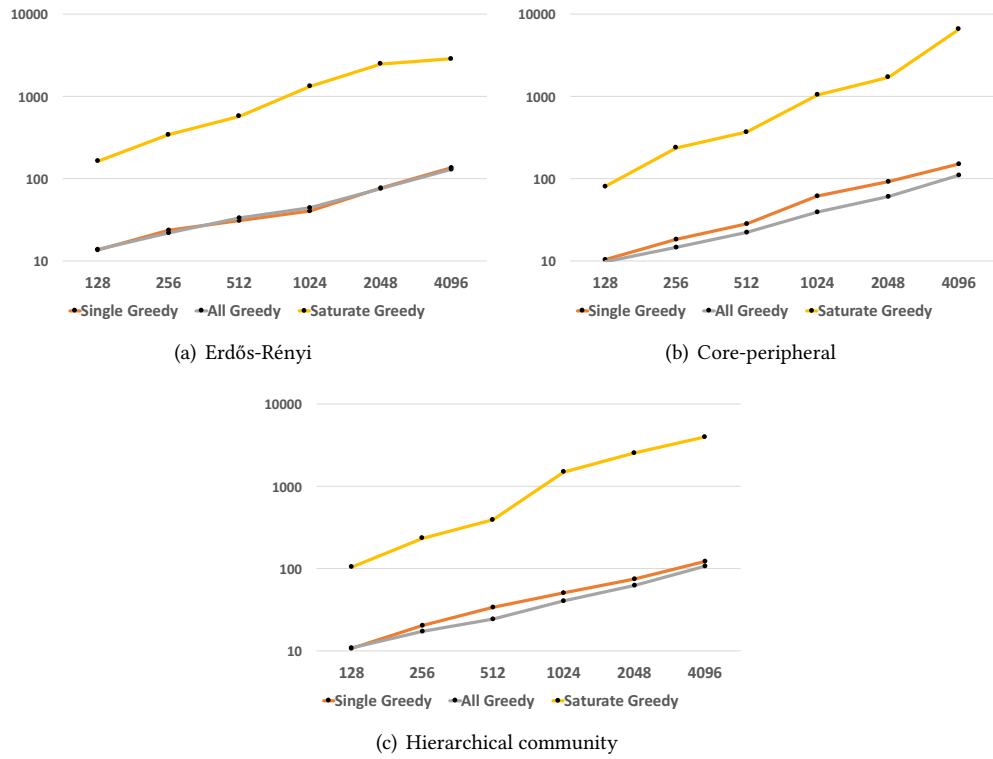
7 CONCLUSION AND FUTURE WORK

In this work, we began a study of the stability of Influence Maximization when the input data are adversarially noisy. We showed that estimating the susceptibility of an instance to perturbations can be cast as an Influence Difference Maximization problem. Unfortunately, the Influence Difference Maximization problem under the Independent Cascade model is as hard to approximate as the Independent Set problem. Thus, we empirically evaluated the stability of Influence Maximization instances on different synthetic and real-world networks under different diffusion models and diffusion settings. As a general result from the experiments, caution is advised in using the algorithmic result of Influence Maximization when there is a large amount of noise (with relative error of 20%).

Given the hardness of the Influence Difference Maximization problem, we then designed an efficient algorithm for Robust Influence Maximization to find influential users robustly across multiple diffusion settings. We carried out a theoretical analysis on the hardness of the Robust Influence Maximization problem and proved an approximation guarantee for our algorithm.

An interesting unresolved question is whether one can efficiently find an (approximately) worst-case influence function in the Perturbation Interval model. This would allow us to empirically evaluate the performance of natural heuristics for the Perturbation Interval model, such as randomly sampling a small number of influence functions. Furthermore, it would allow us to design "constraint generation" style algorithms for the Perturbation Interval model in the style of the algorithm of Chen et al. [11].

In the context of the bigger agenda, one could conceive of other notions of robustness in Influence Maximization, perhaps tracing a finer line between worst-case and Bayesian models. Also, much more research is needed into identifying which influence models best capture the behavior of real-world cascades, and under what circumstances. It is quite likely that different models will perform differently depending on the type of cascade and many other factors, and in-depth evaluations



(a) Erdős-Rényi

(b) Core-peripheral

(c) Hierarchical community

Fig. 8. Running times on Kronecker graph networks with different structures. The x axis represents the number of nodes, and the y-axis is the running time in seconds, both plotted on a log scale.

of the models could give practitioners more guidance on which mathematical models to choose. While our model of robustness allows us to combine instances of different models (e.g., IC and LT), this may come at a cost of decreased performance for each of the models individually. Thus, it remains an important task to identify the influence models that best fit real-world data.

Acknowledgments. We would like to thank Shishir Bharathi, Shaddin Dughmi, and Mahyar Salek for useful discussions and feedback, and anonymous reviewers for useful feedback on conference versions of the present material. In particular, we would like to thank Debmalya Mandal, Jean Pouget-Abadie and Yaron Singer for bringing to our attention a counter-example to an earlier incorrect claim that the objective function for Influence Difference Maximization is submodular.

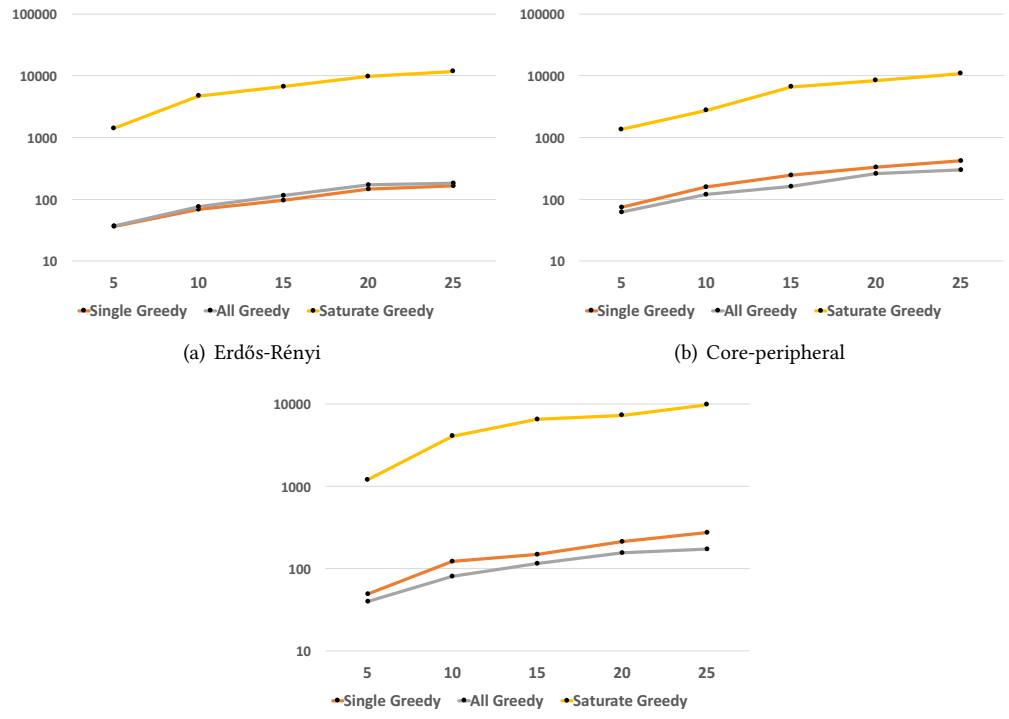
REFERENCES

[1] Bruno Abrahao, Flavio Chierichetti, Robert Kleinberg, and Alessandro Panconesi. 2013. Trace Complexity of Network Inference. In *Proc. 19th Intl. Conf. on Knowledge Discovery and Data Mining*. 491–499.

[2] Abhijin Adiga, Chris J. Kuhlman, Henning S. Mortveit, and Anil Kumar S. Vullikanti. 2013. Sensitivity of Diffusion Dynamics to Network Uncertainty. In *Proc. 28th AAAI Conf. on Artificial Intelligence*.

[3] Noga Alon, Itai Benjamini, and Alan Stacey. 2004. Percolation on finite graphs and isoperimetric inequalities. *Annals of Probability* 32 (2004), 1727–1745.

[4] Eric Balkanski, Nicole Immorlica, and Yaron Singer. 2017. The Importance of Communities for Learning to Influence. In *Proc. 29th Advances in Neural Information Processing Systems*. 5864–5873.



(a) Erdős-Rényi

(b) Core-peripheral

(c) Hierarchical community

Fig. 9. Running times on Kronecker graph networks with different structures. The x axis represents the number of diffusion settings, and the y-axis is the running time in seconds, both plotted on a log scale.

[5] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. 2016. The power of optimization from samples. In *Proc. 28th Advances in Neural Information Processing Systems*. 4017–4025.
[6] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. 2017. The Limitations of Optimization from Samples. In *Proc. 49th ACM Symp. on Theory of Computing*. 1016–1027.
[7] Noam Berger, Béla Bollobás, Christian Borgs, Jennifer Chayes, and Oliver Riordan. 2003. Degree distribution of the FKP network model. In *Proc. 30th Intl. Colloq. on Automata, Languages and Programming*. 725–738.
[8] Christian Borgs, Michael Brautbar, Jennifer T. Chayes, and Brendan Lucier. 2014. Maximizing Social Influence in Nearly Optimal Time. In *Proc. 25th ACM-SIAM Symp. on Discrete Algorithms*. 946–957.
[9] Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. 2014. Submodular Maximization with Cardinality Constraints. In *Proc. 25th ACM-SIAM Symp. on Discrete Algorithms*. 1433–1452.
[10] Karen E Campbell and Barrett A Lee. 1991. Name generators in surveys of personal networks. *Social networks* 13, 3 (1991), 203–221.
[11] Robert S. Chen, Brendan Lucier, Yaron Singer, and Vasilis Syrgkanis. 2017. Robust optimization for non-convex objectives. In *Proc. 29th Advances in Neural Information Processing Systems*. 4708–4717.
[12] Wei Chen, Laks V. S. Lakshmanan, and Carlos Castillo. 2013. *Information and Influence Propagation in Social Networks*. Morgan & Claypool.
[13] Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. 2016. Robust Influence Maximization. In *Proc. 22nd Intl. Conf. on Knowledge Discovery and Data Mining*. 795–804.
[14] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. In *Proc. 15th Intl. Conf. on Knowledge Discovery and Data Mining*. 199–208.
[15] Wei Chen, Yajun Wang, Yang Yuan, and Qinshi Wang. 2016. Combinatorial multi-armed bandit and its extension to probabilistically triggered arms. *Journal of Machine Learning Research* 17, 50 (2016), 1–33.

[16] Wei Chen, Yifei Yuan, and Li Zhang. 2010. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In *Proc. 10th Intl. Conf. on Data Mining*. 88–97.

[17] Irit Dinur and David Steurer. 2014. Analytical approach to parallel repetition. In *Proc. 46th ACM Symp. on Theory of Computing*. 624–633.

[18] Pedro Domingos and Matthew Richardson. 2001. Mining the Network Value of Customers. In *Proc. 7th Intl. Conf. on Knowledge Discovery and Data Mining*. 57–66.

[19] Nan Du, Yingyu Liang, Maria-Florina Balcan, and Le Song. 2014. Influence Function Learning in Information Diffusion Networks. In *Proc. 31st Intl. Conf. on Machine Learning*. 2016–2024.

[20] Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuan Zha. 2013. Scalable Influence Estimation in Continuous-Time Diffusion Networks. In *Proc. 25th Advances in Neural Information Processing Systems*. 3147–3155.

[21] Paul Erdős and Alfréd Rényi. 1960. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci* 5, 1 (1960), 17–60.

[22] Jacob Goldenberg, Barak Libai, and Eitan Muller. 2001. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters* 12, 3 (2001), 211–223.

[23] Jacob Goldenberg, Barak Libai, and Eitan Muller. 2001. Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata. *Academy of Marketing Science Review* 2001 (2001), 1.

[24] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. 2011. Uncovering the temporal dynamics of diffusion networks. In *Proc. 28th Intl. Conf. on Machine Learning*. 561–568.

[25] Manuel Gomez-Rodriguez, Jure Leskovec, and Andrease Krause. 2012. Inferring networks of diffusion and influence. *ACM Transactions on Knowledge Discovery from Data* 5, 4 (2012), 21:1–21:37.

[26] Manuel Gomez-Rodriguez and Bernhard Schölkopf. 2012. Submodular inference of diffusion networks from multiple trees. In *Proc. 29th Intl. Conf. on Machine Learning*.

[27] Manuel Gomez-Rodriguez, Le Song, Hadi Daneshmand, and B. Schölkopf. 2014. Estimating Diffusion Network Structures: Recovery Conditions, Sample Complexity & Soft-thresholding Algorithm. In *Proc. 31st Intl. Conf. on Machine Learning*. 793–801.

[28] Amit Goyal, Francesco Bonchi, Laks VS Lakshmanan, and Suresh Venkatasubramanian. 2013. On minimizing budget and time in influence propagation over social networks. *Social Network Analysis and Mining* 3, 2 (2013), 179–192.

[29] Amit Goyal, Francesco Bonchi, and Laks V. S. Lakshmanan. 2011. A Data-based Approach to Social Influence Maximization. *Proc. VLDB Endow.* 5, 1 (sep 2011), 73–84.

[30] Mark Granovetter. 1978. Threshold Models of Collective Behavior. *Amer. J. Sociology* 83 (1978), 1420–1443.

[31] Avinatan Hassidim and Yaron Singer. 2016. Submodular optimization under noise. In *Proc. 30th Conference on Learning Theory*. 1069–1122.

[32] Johan Håstad. 1999. Clique is hard to approximate within $n^{1-\varepsilon}$. *Acta Mathematica* 182 (1999), 105–142.

[33] Xinran He and David Kempe. 2016. Robust Influence Maximization. In *Proc. 22nd Intl. Conf. on Knowledge Discovery and Data Mining*. 885–894.

[34] David Kempe, Jon Kleinberg, and Éva Tardos. 2015. Maximizing the Spread of Influence through a Social Network. *Theory of Computing* 11, 4 (2015), 105–147.

[35] Harry Kesten. 1990. Asymptotics in High Dimension for Percolation. In *Disorder in Physical System*, Geoffrey Grimmett and Dominic Welsh (Eds.). Oxford University Press, 219–240.

[36] Sanjeev Khanna and Brendan Lucier. 2014. Influence Maximization in Undirected Networks. In *Proc. 25th ACM-SIAM Symp. on Discrete Algorithms*. 1482–1496.

[37] Andreas Krause, H. Brendan McMahan, Carlos Guestrin, and Aunpam Gupta. 2008. Robust Submodular Observation Selection. *Journal of Machine Learning Research* 9 (2008), 2761–2801.

[38] Siyu Lei, Silviu Maniu, Luyi Mo, Reynold Cheng, and Pierre Senellart. 2015. Online influence maximization. In *Proc. 21st Intl. Conf. on Knowledge Discovery and Data Mining*. 645–654.

[39] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. 2009. Meme-tracking and the dynamics of the news cycle. In *Proc. 15th Intl. Conf. on Knowledge Discovery and Data Mining*. 497–506.

[40] Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. 2010. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research* 11 (2010), 985–1042.

[41] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie S. Glance. 2007. Cost-effective Outbreak Detection in Networks. In *Proc. 13th Intl. Conf. on Knowledge Discovery and Data Mining*. 420–429.

[42] Meghna Lowalekar, Pradeep Varakantham, and Akshat Kumar. 2016. Robust influence maximization. In *Proc. 15th Intl. Conf. on Autonomous Agents and Multiagent Systems*. 1395–1396.

[43] Elchanan Mossel and Sebastien Roch. 2010. Submodularity of Influence in Social Networks: From Local to Global. *SICOMP* 39 (2010), 2176–2188.

[44] Seth A. Myers and Jure Leskovec. 2010. On the convexity of latent social network inference. In *Proc. 22nd Advances in Neural Information Processing Systems*. 1741–1749.
[45] Harikrishna Narasimhan, David C Parkes, and Yaron Singer. 2015. Learnability of Influence in Networks. In *Proc. 27th Advances in Neural Information Processing Systems*. 3168–3176.
[46] George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming* 14 (1978), 265–294.
[47] Praneeth Netrapalli and Sujay Sanghavi. 2012. Learning the Graph of Epidemic Cascades. In *Proc. 12th ACM Sigmetrics Intl. Conf. on Measurement and Modeling of Computer Systems*. 211–222.
[48] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. 2009. Learning Continuous-Time Information Diffusion Model for Social Behavioral Data Analysis. In *Proceedings of the 1st Asian Conference on Machine Learning: Advances in Machine Learning*. 322–337.
[49] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. 2010. Generative Models of Information Diffusion with Asynchronous Timedelay.. In *ACML*. 193–208.
[50] Youze Tang, Yanchen Shi, and Xiaokui Xiao. 2015. Influence maximization in near-linear time: A martingale approach. In *Proc. 35th ACM SIGMOD Intl. Conference on Management of Data*. 1539–1554.
[51] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proc. 34th ACM SIGMOD Intl. Conference on Management of Data*. 75–86.
[52] Chi Wang, Wei Chen, and Yajun Wang. 2012. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery* 25 (2012), 545–576.
[53] Yu Wang, Gao Cong, Guojie Song, and Kunqing Xie. 2010. Community-based greedy algorithm for mining top-K influential nodes in mobile social networks. In *Proc. 16th Intl. Conf. on Knowledge Discovery and Data Mining*. 1039–1048.
[54] Amulya Yadav, Bryan Wilder, Eric Rice, Robin Petering, Jaih Craddock, Amanda Yoshioka-Maxwell, Mary Hemler, Laura Onasch-Vera, Milind Tambe, and Darlene Woo. 2017. Influence maximization in the field: The arduous journey from emerging to deployed application. In *Proc. 16th Intl. Conf. on Autonomous Agents and Multiagent Systems*. 150–158.